Activity Context Representation: Techniques and Languages
AAAI Technical Report WS-12-05

Activity-Context Aware Computing for Supporting Knowledge-Works

Arijit Laha[1], **Lokendra Shastri**[2] and **Vikas Agrawal**[1]
C-KDIS, Infosys Labs, Infosys Ltd,
[1]Hyderabad, [2]Bangalore

Abstract

The problem of designing and building effective assistive systems for human agents performing "professional knowledge-intensive activities", or "knowledge-works" is of great interest and has wide implications. In this paper we propose a new approach for solving the problem. The approach is based on activity-context aware computation paradigm that can lead to flexible yet robust systems for holistic support in performing complex knowledge-works. To this end, we also outline here the notion of "activity-context" and the idea of "activity-models" as core artifacts used by such systems embodying the notion.

Introduction

There are mainly two areas of research within which the idea of "context" is studied from computational perspective, (1) Information Science (Saracevic 1999) and (2) Mobile/Ubiquitous Computing (Dey, Abowd, and Salber 2001). In information science, context is viewed as a tool for judging/establishing relevance of retrieved information. This led to development of various types, e.g., explicit, implicit, blind, etc., of relevance-feedback based information retrieval (IR) systems (Ruthven and Lalmas 2003). In the later field, the interest is essentially in delivering context-aware services through hand-held/mobile devices, where context is perceived mainly in the light of the parameters (e.g., location, motion, acceleration, etc.) sensed by the sensors embedded in the devices.

In contrast, we are interested in solving the problem of designing and building computational systems, more specifically, *assistive systems for human agents* performing "professional knowledge-intensive activities", or "knowledge-works". Efficient and high quality performance of knowledge-works (planning, designing, research, analysis, etc.) is crucial for any organization. However, these activities can be extremely complex, often amounting to solving "unstructured" or "ill-defined" problems (Markus, Majchrzak, and Gasser 2002; Laha 2011). Nevertheless, people *must* perform them and any significant improvement in the level of assistance available to them in doing so, in terms of quality and/or efficiency, can be of great value to the workers, to their organizations as well as to the society at large.

A knowledge-work, as a whole, can be perceived as a complex mix of *interdependent* processes, encompassing cognitive as well as system-assisted ones (Dror and Harnad 2008). Of these, IR processes are, though crucial, are far from being only ones. Also, while devices enabling ubiquitous computing can be of great value, they mostly act as means to communication and interaction with a system dealing with a much wider context. In this paper we propose a rich "activity-centric" view of context, called, for brevity, the "activity-context" and "activity-context aware computation" paradigm, leveraging the idea, which can be used for designing as well as implementing a new class of assistive systems.

In the rest of the paper, we start with a discussion on some of the difficult aspects of knowledge-works and their implications to system design. Then we introduce the notion of activity-context as an alternative framework for understanding knowledge-works. Following this, we introduce a formalism for activity-modeling and outline the activity-context aware computational paradigm. In the next section we explain the application of the proposed approach in light of a use-case. Then we conclude the paper with a discussion.

Knowledge-work and activity-context

Performance of a knowledge-intensive activity or knowledge-work in undertaken by human agents in order to find solutions of complex problems. Such activities, especially in professional spheres, appear in myriads of form and nomenclatures, e.g., research, design, analysis, planning, etc. Despite the training and experience of a worker, performing such an activity presents many elements of novelty, which requires the worker to exercise her ingenuity. Further, while performing, a worker needs to build and actively maintain a detailed context of the current problem-at-hand. This context maintained by a worker drives her information consumption (seeking, retrieval, relevance judgment and understanding/internalization in form of new knowledge) and information production/creation (articulation of new/updated knowledge in symbolic and persistent form(s)) behaviors in relation to the performance of the knowledge-work.

Human society, all over the history, always attempted to make best possible use of the available technological means, sometimes known as “cognitive technologies” (Dror and Harnad 2008), for expanding capabilities for performing knowledge-works. Use of computational systems is the newest addition in this repertoire and we need to understand and explore their capabilities further. Attempts in this direction include Ingwersen and Järvelin (Ingwersen and Järvelin 2005) proposing new research directions for bridging information seeking and retrieval, Pédauque (Pédauque 2003) calling for a re-look into organization and presentation of information contained in digital documents, the ASAP system for genomic research (Glasner et al. 2006), the CODEX system for geography/geology research (Pike and Gahegan 2007) and many others.

The nature of the problem

Here we argue from a holistic perspective that there are essentially three aspects of performing knowledge-works where a computational system, by virtue of its great capability of dealing with information, can assist a knowledge-worker in significantly improved manner. They are (1) helping the worker to build and maintain the “context” of work; (2) helping in consuming information more efficiently; and (3) help in creating new information in *richly contextualized* form without significant additional overhead. To be able to do so, the system itself needs to be imbibed with a sense of the “context” of the work being performed using it - which we shall call the “activity-context”, to be defined and explained shortly.

Before proceeding further, let us look into the typical level of support one finds today regarding the above-mentioned aspects of knowledge-works. In a modern Information and Communication Technology (ICT) enabled work-environment, while one finds an abundance of tools bringing about a lot of capabilities/functionality, they are seldom integrated around *activities of interest* and managing them itself poses significant cognitive load (Markus, Majchrzak, and Gasser 2002). Similarly, while we have vast repositories of information and powerful tools for accessing them, very often we are inundated with “information overload” (Eppler and Mengis 2004). Also, the tools used for articulation and capture of information are mostly modeled after physical paper-based documentation paradigm (Pédauque 2003), with most of its constraints such as linearity in organization of information, which makes it difficult to capture much contextual details.

The problem can be viewed in even simpler terms - to understand a piece of information, one needs to know the *who, when, where, what, why* and *how* (the famous 5w & h), analyze and understand them in light of the current activity-context. Conventional ICT tools can capture the first four w’s (let us generously accept that the fields like “title” and “keywords” represent the “what” adequately); we call them *metadata*. But the “why” and “how” are neither captured by the system nor the creator typically has wherewithal to articulate them in adequate details.

Activity-context

Clearly, in order to achieve the above-mentioned capabilities up to a significant degree, the system needs to have an adequate understanding of an supported activity. We argue that performance of an activity takes place within an *work-environment* in order to achieve some *objective(s)*. To perform the activity, the *actor* needs to enact a *process*, which, in turn, require finding and utilizing *resources* from the work environment. Naturally, these elements of an activity has complex interrelationships - of which the system needs awareness to a good degree. In Figure 1 we depict a high level view of the important elements of knowledge-work and some of their important relationships in a typical ICT-enabled workplace. The view depicted in Figure 1 is inspired by the notion of “human activity” as perceived in Activity Theory (Kaptelinin and Nardi 2006). Note that, the set of elements (and relationships) indicated in the figure are neither claimed to be *exhaustive* nor *mandatory,* but indicative only. However, for the scope of this paper this will serve adequately.

The view of a knowledge-work as depicted in Figure 1 can readily be used for the purpose of analysis and understanding an activity in order to build a support system. However, here we intend to use it as a basis of creating a “model” of an activity which will embody a system’s understanding about the “supported activity”. In order to do so, the model needs to be deployable in a computational system, which, in turn, requires the model be *formal one*. We shall explore this in the next section. However, at this point, assuming such a model, which we shall call an “activity-model”, exists for an activity, we can define the term “activity-context” as follows:

Activity-context is a set of information, potentially relevant for continuing performance of the current episode of an activity, which includes

- general categorical information about the elements of activity as embodied in the corresponding activity-model;
- instance-specific/episodic information associated with performances, current and past, of activities of same or related types; and
- information about available resources (e.g., information sources) and means for utilizing them in order to find and process more of relevant information.

Note that the above definition is very specific and technical one and more importantly, *biased towards complex knowledge-works*. It is anchored in the idea of an activity-model, explicit or implicit. In a sense, an “activity-model” acts as a filter and classifier of the available information (including information about work-environment) in order to judge their relevance in context of the modeled activity. Any information commensurate with the model, can be part of the “activity-context”.

Activity-context aware systems

Now we are in a position to define an “activity-context aware” assistive system as one that has access to the models

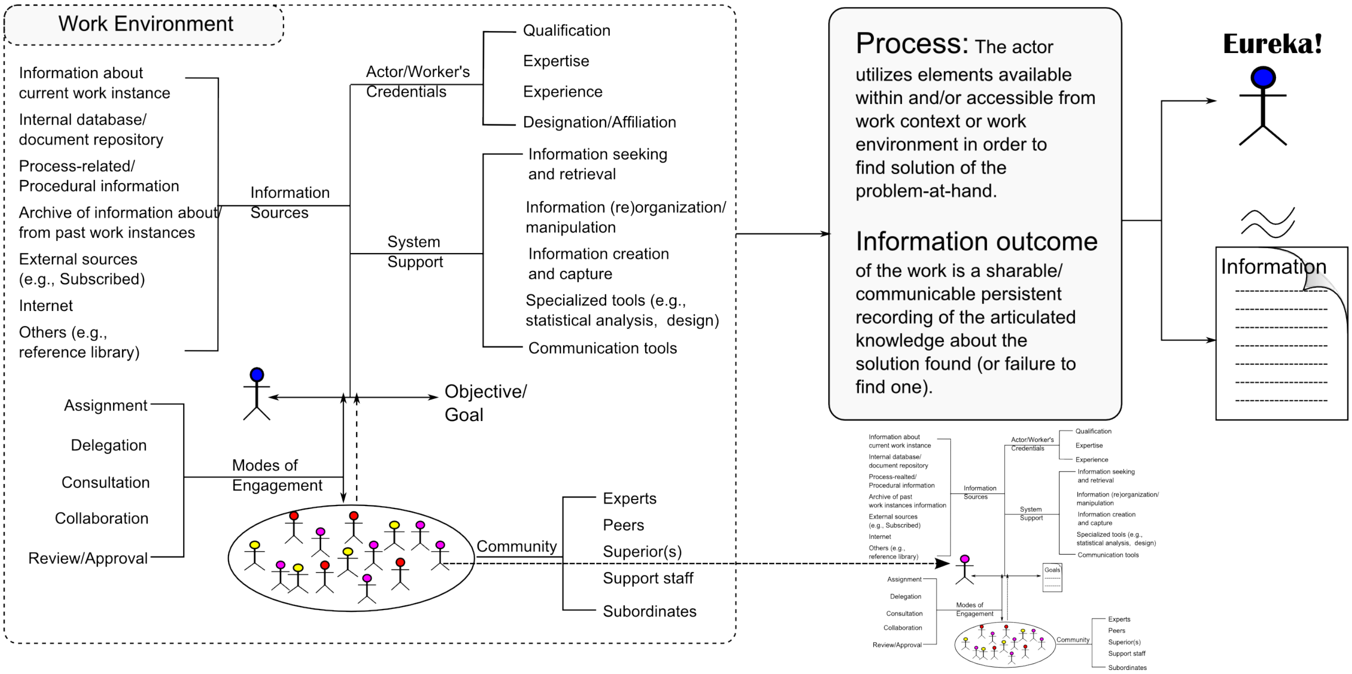

Figure 1: Elements of a knowledge-work in an ICT-enabled work environment.

of the supported activities and has means to leverage it in order to accomplish the following:

1. maintain the activity-context during performance of an activity, present relevant aspect of it to the human actor in order to help her building and maintaining cognitive "work-context;
2. analyze and understand user's requirements during the performance of the activities and attempts to make available the resources relevant for fulfilling those requirements; and
3. capture relevant part of activity-context associated with new information created by the actor during the performance of an activity as well as identify/tag them with appropriate elements of the activity-model.

In the following we develop a formalism for representing activity-models in a machine-deployable form. Then we shall outline a computational paradigm that can be followed for achieving the above capabilities.

Formal activity-model

The activity-model proposed above can be formalized by representing the model of an activity a_i as a 4-tuple,

$$a_i =< I_i, P_i, O_i, S_i >,$$

where, I_i is a set of items, called the *itemset*, relevant in context of a_i. It includes the set En_i, representing (classes of) entities (including, for computational convenience, O_i) and the set of relationships $Rel_i = \{r_{jk} | en_j, en_k \in En_i\}$ among them, *potentially important or useful* for the purpose of performing the activity a_i successfully. P_i represents the process, in form of the sub-activity (if any) structure of a_i, that needs to be enacted in order to perform a_i. Formally, P_i is a directed acyclic graph (DAG), whose nodes are again activity-tuples and the directed edges connecting them connote dependency relationships. P_i contains one (and only one) node a_f, which has the same outcome as a_i, the super-activity and its concluding its performance denote concluding that of the super-activity. O_i is the set of items constituting the outcome of performing a_i. Finally, $S_i = \{Ins(*)\}$ is a set of *instantiation strategies* for objects relevant in activity-context.

The activity-tuple as described above can be viewed as the general typological or categorical model of the activity, where the elements of the model refer to instance-independent "classes" of entities and relationships (and possibly some ancillary information about them). Such models for supported activities makes up the core part of an activity-context aware system. For every instance or episode of the activity performed using the system, a typological model acts as an template for creating the instance-specific or episodic model, $a_i^e =< I_i^e, P_i^e, O_i^e, S_i^e >$, where the superscript denotes a particular (e^{th})episode of performance.

As argued earlier, in order to discover and capture the activity-context, the system needs to follow, leverage and capture the dynamics/trajectory of performance of an activity. To this end we define the episodic itemset I_i^e to include the triple $< En_i^e, Rel_i^e, Ob_i^e >$, consisting of a set of entity-classes $En_i^e \supseteq En_i$, a set relevant relationships among them $Rel_i^e \supseteq Rel_i$ and a set objects Ob_i^e, where $ob_j \in Ob_i^e$ is discovered/evaluated/instantiated in course of performance of a_i^e and classified as an instance of $en_i \in En_i^e$. In an episodic model the strategy set S_i^e is consisted of the strategies of the form $\left\{Ins^e_{k|j}(I^e_i)\right\}$, where $ob_j \in Ob^e_i, ob_k \notin Ob^e_i, \exists r_{jk} \in Rel^e_i$. However, a detailed discussion of the encoding of these strategies is outside the scope of the current paper.

Note that, with respect to the entity and relationship sets, we have defined the episodic versions as supersets of the typological ones. This is very important, since we do not assume that the typological model is complete, in general. In fact, such an assumption can be severely restricting in case of complex knowledge-intensive activities. So, a human worker is allowed to include entities and relationships to the environment if she chooses so.

Computing with activity-model

Now we can outline a simple computation paradigm based on above ideas. Consider a knowledge-worker performing an activity instance a^e_i using a system equipped with the model a_i.

1. At the beginning (i.e., $t = 0$) the set $I^e_i(0) =< En^e_i = En_i, Rel^e_i = Rel_i, Ob^e_i(0) >$, i.e., episodic the entity and relationship sets are identical as the typological ones, $Ob^e_i(0)$ is consisted of some of the entities evaluated - these are the initial or input entities;
   - Now, the challenge for the support system and human agent together is to instantiate/evaluate rest of the entities in En^e_i and finally O^e_i,
2. The system and user uses the relationships in $I^e_i(t)$, in which the already "instantiated" entities participate, in order instantiate the other entities in the other end of the relationships;
   - The system and user interacts/collaborates in this process and gradually step-by-step transform the $I^e_i(t') \longrightarrow I^e_i(t'+1)$ by evaluating yet unevaluated entities;
   - Also, at this stage, if user needs so, she may include new entities and/or relationships in the episodic itemset.
3. The process continues till the outcome entities $O^e_i \in En_i$ are evaluated - to the acceptable value(s) for success, to failure otherwise.

The simple description above can accommodate enormous range of variations in the ways they can be designed and implemented. The model a_i can be implemented directly into the application programmatically or it can be derived form a knowledge-base in form of a ontology/semantic network. At the extreme end we can think of making the system adaptive so that the I_i and P_i can start from an initial knowledge-base and enriched thereafter by learning from the users' behavior across episodes performed.

The next step (step 2) is the at the heart of the matter and we can consider a vast number of alternatives with varying degrees of sophistication. Two aspects of activity-models, (1) the granularity level up to which the activities are modeled; and (2) richness of the itemsets I_is for the modeled activities will be crucial factors here. Beyond that it is about systems ability to choose appropriate instantiation strategies for unevaluated entities, given the current activity-context and, of course, the richness of the strategies themselves.

An instantiation strategy $Ins^e(*)$ represents access to various resources as well as means to marshal and compose their services in order to enable the identification of the desired object(s) - automatically, if possible or based on human actor's inputs/assertions or a (hopefully optimal) blend of both. For example, we shall observe the use of qualifier "convenient" in the use-case description. The constraints defining the notion of "convenience" are not likely to be fully objective, and thus amenable to formalization, in nature and it is the human agent who needs to make the judgment whether a choice qualifies as such or not. So, here the challenge is to design the strategy that enables the user to make a correct choice with minimum effort.

It is defining and implementing $Ins^e(*)$, where the design choices need to be made. In one end, one can go for purely programmatic/procedural approach, while other choices include knowledge-based approaches of various degrees of sophistication, including on-the-fly approximate reasoning in order to address uncertainties. Also, because the system is assistive one, there is always a human agent in the loop. Thus it may not be crucial to zero on only one alternative in face of uncertainty, even presenting the user a small number of them along with enough wherewithal for easily acting upon them will serve greatly. Here we have no recommendation to offer for the choice of any of the possible approaches or refinement levels, they can be chosen with pragmatism on problem-to-problem basis. However, we expect that the analysis and design of AFHM ("away from home" management) module in the next section may provide some idea on these issues to the readers.

An use-case

The methodology developed above can be used to build systems for supporting very complex knowledge-works, e.g., "patient-care" in a medical care institution (Patel, Arocha, and Zhang 2005). However, such systems pose highly complex design challenges which cannot be properly dealt with within the scope of this paper. Instead, here we shall work with a use-case of moderate complexity, an "assisted living" service. We shall consider a plausible scenario and attempt to do a high level design of selected part the system as an "activity-context aware" system, to be used by the service provider in order to effectively resolve the problems in the described scenario.

The scenario

The following are the relevant aspects of the service from the (service) subscriber's perspective:

- The service is targeted at aged persons who reside at their own house and are willing to take help in form of reminders to take medication, exercise, etc., and suitable interventions in case of health-related emergencies and contingencies;
- The service is graceful one, i.e., it attempts to cause minimal disruptions/restrictions to an assisted person's normal life;

$$I_{AH} = \left[\begin{array}{c} En_{AH} = \left\{ \begin{array}{c} Id/profile\ (including\ means\ to\ contact)\ of\ assisted\ person,\ Accessibility\ to\ medication(?), \\ Medication\ Id/Name,\ Medication\ schedule,\ Medicine\ Store(?),\ Delivery\ service(?) \\ Current\ Location,\ Target\ location(i.e.,\ Expected\ location\ at\ medication\ time)(?) \\ Tools\ available:\ Location\ server,\ Directories,\ Communication\ tools \\ Information\ sources:\ Past\ transactions\ databases,\ Publicly\ availanle\ information \end{array} \right\} \\ Rel_{AH} \left\{ \begin{array}{c} Distances: < Target\ location,\ Medicine\ Store,\ Delivery\ service > \\ Availability: < Medication\ Id/Name,\ Medicine\ Store > \\ Cost: < Medication\ Id/Name,\ Medicine\ Store,\ Delivery\ service > \end{array} \right\} \end{array} \right]$$

$$O_{AH} = Status:\ Medication\ in\ hand\ of\ assisted\ person(?)$$

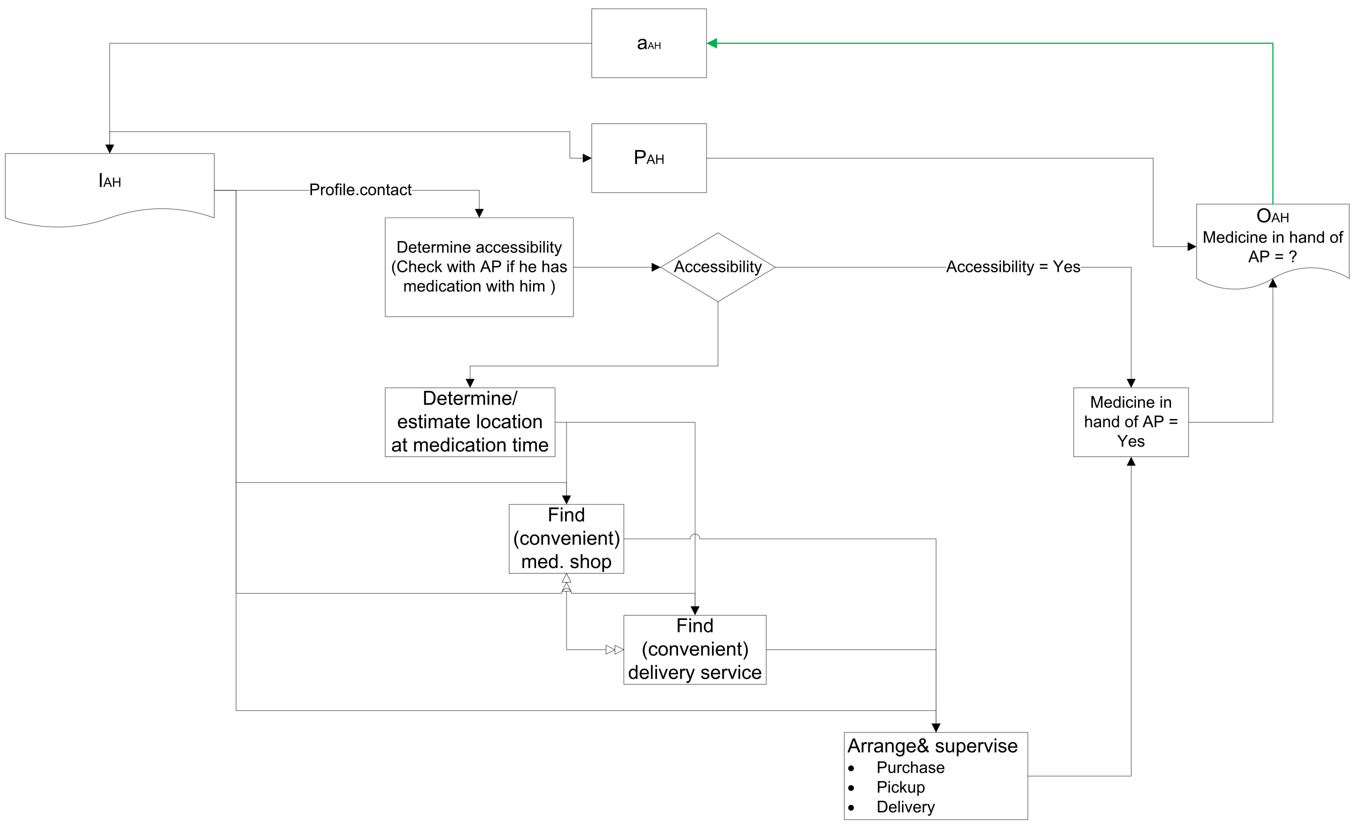

Figure 2: Elements of a_{AH}

- Under the service, the subscriber/assisted person is nominally provided with an application running on a mobile (say, a smartphone) or wearable (say, an enhanced wristwatch) computing and communication device with some standard (e.g., GPS location, accelerometer, etc.) sensors;
- The subscriber is also provided with some other equipments, including a "medicine dispenser" installed in the house (in the kitchen, perhaps), which is serviced (e.g., filling the dispenser) as required.

The nominal/standard service includes the following:

- The subscriber receives timely (say, 2-3 minutes ahead) reminders from the service provider for taking medicine;
- He goes to the dispenser and pushes a button - the dispenser (let us go high-tech here and let the dispenser be programmable one, remotely or otherwise, based on the prescription) dispenses the *appropriate* medicine(s).

At the service provider's end, there are "service supervisors", each of them has a number subscribers assigned to her. In her computer screen, the *status* of every assignee vis-à-vis the service obligations/commitments is displayed. As long as a subscriber's behavior is within the nominal envelope, the system takes care of the tasks (issuing reminder, scheduling a visit to subscriber's home to refill the dispenser, etc.) more or less automatically. It is when it detects some situation of emergency e.g., the subscriber had a fall (detected based on the analysis of accelerometer data) or some contingency, say, the subscriber is *away from home* - has traveled to another part of the city to pay a visit to a friend - the system kicks into a different mode where the supervisor's judgment and direction is needed to respond properly.

Let us take up the "away from home" or AFH, for brevity, scenario. Let us also assume that the service commitment extends to "enabling the subscriber to keep medication schedule", i.e., not merely issuing routine reminders but ensuring accessibility, with maximal quality (correctness, timeliness, etc.) and minimal cost. Thus, once the system detects the AFH scenario and invokes the AFHM (AFH management) module, the supervisor uses it to perform the following:

- Instead of the usual time, the subscriber is given a different reminder ahead (say 15-20 mins.) of time - "In 't' time you need to take medicine 'x'. do you have it with you?"
- If the subscriber answers yes, the service says "great, shall remind you at usual time";
- If no, the service says, "Don't worry. Let us take care of it.";
- and does the following:
  - locates a *convenient* medicine store, buys the medicine (online);
  - hires a *convenient* "pickup and delivery" service to pick-up the medicine from the shop and deliver to the assisted person;
  - follows-up the execution of the arrangement to ensure that it goes smoothly, i.e., the medicine reaches to the hand of the subscriber in proper time;
- gets back to the nominal mode.

Now the service, confident that the accessibility is ensured, can issue a regular reminder to the subscriber for taking the medicine at designated time.

Just to get some sense of the scale of the problem, let us assume that the supervisor supervises 100-150 subscribers, out of which, at a given time, approximately 10%-15% of them are in some or other "out of nominal" state (such as AFH). The supervisor needs to respond to and resolve these situation within admissible time limits varying from 5 minutes (say, for getting an ambulance to the subscriber who had a fall and not moving since - inferred from the accelerometer and motion sensor data) to 30 minutes (for one who is possibly oversleeping or has forgot to wear the device after waking up - inferred from the motion sensor data). And on the top of these factors, there is quality and cost consideration.

The system

Let us denote "Away-from-Home" Management activity to be performed by a supervisor as a_{AH}. Now, let us try to see its various elements. For brevity, we shall assume that there is a "Assisted Living Service Ontology (ALSO)" from which these elements are drawn and their semantics (i.e., relationships and entities which may be required but not included in activity-model explicitly) can be inferred. Some (naturally, even for this use-case with moderate complexity, it is not possible to include all, but, when required, they will be assumed to be available from ALSO) of the major elements of a_{AH} are depicted in Figure 2 (unevaluated entities are marked with '?', with exception of S_{AH}. Here, since a_{AH} is a composite activity, there is only one strategy, that exposes to the user the sub-activity structure P_{AH} (shown in lower part of Figure 2), enables performance of sub-activities commensurate with the dependency constraints and waits till the final sub-activity is over.

Now, let us study the activity of "finding a convenient medicine shop", a_{FM}. The itemset I_{FM} shares (let us assume for computational convenience) elements of I_{AH} as well as include other elements of importance. For example, the "outcome entity" $Med\,Store\,to\,Purchase\,from(?) \in O_{FM}, En_{FM}$, while many on the criteria for "convenience" are subjective, the criterion of "nearness" can be encoded objectively as a relation,

$$isNearBy < Target\,Loc,\ Medicine\,Store > \in Rel_{FM}.$$

Similarly, before selecting the shop, it must be ensured that the medicine is actually available at the shop. So,

$$isAvailableAt < Medicine\,Store,\ MedicineID > \in Rel_{FM}.$$

There are several other elements specific to I_{FM}, which we shall not try to list here. We just want to convey the spirit of activity-context aware computing for this activity. At this point (1) it is ascertained that the subscriber is not carrying the medication; and (2) the location (i.e., the target location) the subscriber will be at the time of medication is known. So, now the system have all the prerequisites for operationalizing a strategy for instantiating a "medical store" object from which required medication can be bought.

First step in the strategy is to identify a candidate set of "medical stores" using a directory/database and applying the "nearness" criterion on them. Next step is to discover relevant information about them, including contact information (from directory), past transactions, if any (from internal database). Also, if any of the candidate stores has a web application which can be queried about availability of a medicine, the strategy (may be by operationalizing another (sub)strategy) can include ascertaining the availability and modify the candidate list (i.e., drop the store from the list if the medicine is not available there).

At this point, the strategy cannot refine the list further without human intervention - only the human actor can apply other subjective criteria for judging "convenience". So the strategy switches to the goal of creating a user interface that will present a consolidated view of the solution space to the actor and marshal various system functionality in order to accept user's choices and fulfill the requirements associated with the choices.

For example, one possibility is (without going totally "over-the-top") the strategy invokes a "mash-up" engine, that presents to the user, with a geographic map as background, (1) the target location; (2) locations of candidate stores annotated with their respective distances from the target location; and (3) against each store an information box. The information box, among others, contain the contact information - phone number(s), email address, etc.

Now, once the user (in accord with her own judgment of convenience) decides on a store, in order to confirm the availability she clicks on a phone number (say). The system recognized this and rings up the number (if found busy, automatically tries alternative numbers - it classifies the user's action with richer semantics, which is an expression of intent to contact the store over phone, which particular number is used is immaterial) and gets the user in touch with the shop.

Once the supervisor verifies that a convenient store has the required medicine available for sell, she can select it as the outcome object and conclude the activity.

Discussions

The above discussion on the use-case is illustrative one. We do not claim that the above cannot be realized without activity-context aware computing. However, what we argue that using the proposed approach, it may be easier, especially in case of more complex activities, frequently found in the domains of business (Vicente 2000), healthcare, legal research, scientific and technological research (Holyoak and Morrison 2005). Here the main idea is not to specify everything programmatically up to the minute details, but to allow the system to reason about the user's activity, situation, goals, etc. and attempt to take, whenever possible, appropriate actions by itself or enable the user, in best possible manner (within the capability of the system), to take appropriate actions in order to make progress towards achieving the goal. The activity-models and the corresponding activity-context are the means to be used by the system as well as user in order to constrain the possible search space while seeking the solution of a problem.

Most crucial aspect of the proposed approach is its increased focus on the *activities* themselves, while all other elements comes into picture in context of the activities, i.e., through the activity-context. The main advantage of this lies in possible flexibility and resilience of the goal-directed behavior of the system. It can be designed to gracefully fall back to a lower level of support in face of uncertainty, and allow the user to resolve the uncertainty or switching and/or combining strategies based on the semantics of the activity-context. Further, these strategies can be designed to dynamically compose the services of available computational capabilities, thus promoting re-usability of the resources. The approach is also easily extensible in a number of ways. For example, it can be extended to incorporate incremental learning about new entities and their semantics as well as discover broader semantics of known entities by analyzing episodic information.

We are certainly a long way from *fully realizing* the possibilities offered by activity-context aware computing. However, with the recent advances in various fields of relevant computational research and technologies, to name a few, knowledge-based computing, natural language processing, semantic web based standards, semantic computing, large data handling, etc., we are already in a position to build systems with significant value-addition.

References

Dey, A.; Abowd, G.; and Salber, D. 2001. A conceptual framework and a toolkit for supporting the rapid prototyping of contextaware applications. *Human-Computer Interaction* 16(2-4):97–166.

Dror, I. E., and Harnad, S. 2008. *Cognition Distributed: How Cognitive Technology Extends Our Minds*. John Benjamins Publishing Company. chapter Offloading cognition onto cognitive technology, 1–23.

Eppler, M. J., and Mengis, J. 2004. The concept of information overload. a review of literature from organization science, marketing, accounting, mis, and related disciplines. *The Information Society: An International Journal* 20(5):1–20.

Glasner, J.; Rusch, M.; Liss, P.; III, G. P.; Cabot, E. L.; Darling, A.; Anderson, B. D.; Infield-Harm, P.; Gilson, M.; and Perna, N. T. 2006. Asap: a resource for annotating, curating, comparing, and disseminating genomic data. *Nucleic Acids Research* 34(Database issue):D41–D45.

Holyoak, K. J., and Morrison, R. G., eds. 2005. *Cambridge Handbook of Thinking and Reasoning*. Cambridge, UK: Cambridge University Press.

Ingwersen, P., and Järvelin, K. 2005. *The turn: Integration of information seeking and retrieval in context*. Dordrecht: Springer.

Kaptelinin, V., and Nardi, B. A. 2006. *Acting with Technology*. Cambridge, Mass.: MIT Press.

Laha, A. 2011. On challenges and opportunities of designing integrated it platforms for supporting knowledge works in organizations. *Vikalpa* 36(3):21 – 46.

Markus, M. L.; Majchrzak, A.; and Gasser, L. 2002. A design theory for systems that support emergent knowledge processes. *MIS Quarterly* 26(3):179–212.

Patel, V.; Arocha, J.; and Zhang, J. 2005. *Cambridge handbook of thinking and reasoning*. Cambridge: Cambridge University Press. chapter Thinking and reasoning in medicine, 727–750.

Pédauque, R. 2003. Document: Form, sign and medium, as reformulated for electronic documents. Version 3, STIC-CNRS, [Available from: http://hal.archives-ouvertes.fr/docs/00/06/22/28/PDF/sic_00000594.pdf, Visited on 10-th June 2008].

Pike, W., and Gahegan, M. 2007. Beyond ontologies: Toward situated representations of scientific knowledge. *International Journal of Human-Computer Studies* 65(7):674–688.

Ruthven, I., and Lalmas, M. 2003. A survey on the use of relevance feedback for information access systems. *The Knowledge Engineering Review* 18(2):95–145.

Saracevic, T. 1999. Information science. *Journal of the American Society for Information Science* 50(12):1051–1063.

Vicente, K. J. 2000. Hci in the global knowledge-based economy: Designing to support worker adaptation. *ACM Transactions on Computer-Human Interaction* 7(2):263–280.