Ontology-Directed Conceptual Modelling for Interaction Design

Qingyi Hua, Hui Wang, and Matthias Hemmje
Fraunhofer – IPSI
Dolivostr. 15, D-64293 Darmstadt, Germany
{hua, hwang, hemmje}@ipsi.fhg.de

Abstract A usable interactive system provides its users presentation and manipulation of useful concepts for solving their problems at hand without becoming bogged down in accidental features of user interface. It implicates that interaction design is directed by the acquisition and representation of knowledge about the context of use in a way that can be traced back to the users’ problem-solving activity. In this paper we focus on conceptualization of that activity. The conceptualization is characterized by a set of ontological terms that capture a continuum between user tasks and problem domain in a declarative way, and by a framework of conceptual models that describe a system on a very abstract level without being limited to a particular set of design models.

1 Introduction

In HCI a task is defined as an activity performed on computer to achieve a goal [1]. However, there is at least another more or less parallel activity during the performance, that is, the mental processing and representation of the problem to be solved at hand. Based on the beliefs on the problem domain of interest, the problem-solving activity generates the goal and intentions performing the task in terms of Norman’s action theory [2]. Human attention studies point out that human has two control systems: conscious and subconscious [3]. Only a single task requiring working memory can be under conscious control at any one time, whereas the other can control several tasks requiring only skills simultaneously. Users could distract their mental effort from their working situations to manipulation of user interface if a system fails to provide the relevant concepts to the problem-solving context even a quality user interface available. As a result, a usable interactive system provides its users presentation and manipulation of useful concepts for solving their problems at hand without becoming bogged down in accidental features of user interface.

The problem-solving activity relies on the knowledge structures supporting understanding (schemas) and the mechanisms used to organize that knowledge (plans) (see [4] for a cognitive discuss for the structures and mechanisms). In AI terms a problem solver requires two types of knowledge [5]: domain factual knowledge
This paper will appear in the Proceedings of the 10th International HCI Conference

(objects, events, relations, processes, etc) and problem-solving knowledge about how

to achieve various goals, such as problem-solving methods. However, mental

processing and representation differs from a system description in many ways. For

example, human can have much semantic-richer, and more complex schemas than the

system, whereas the amount of knowledge can be handled by human mind at any
given time is limited to 5±2 chunks according to psychologists. As a result, it is

fundamental to develop a shared understanding between the users and designers
according to the context of use [6] such as the intended users, their tasks and

environments.

Contextual development has been recognized to be crucial for meeting the
demands of user-centred systems design [7]. Two categories of approaches can be

found in HCI: task-based and context-oriented. Task-based approaches, e.g. [1],
capture only the hierarchical properties of tasks by sub-tasking and task sequencing

without memory reminding why the tasks are executed or decomposed, whereas
context-oriented ones, e.g. [8], describe rich pictures of real working situations

without mental abstraction. However, the user guides her tasks to be done by making

plans. The properties of plans are anticipation and simplification [4]: a heuristic

nature without a detail analysis of the situation, optimal use of memory with keeping

only critical properties of objects and events, and higher control level without details

of the activity being processed.

In this paper, we present our initial research result on conceptualization of

knowledge about the context of use. The purpose of the conceptualization is to
develop a representation and shared understanding of beliefs, goals, intentions and

plans between the users and designers with a set of conceptual models. One of key
features of our approach is that the conceptualization is captured by a basic set of

ontological terms that act as meta-knowledge to derive concepts in terms meaningful
to the users. Another one is that the description of the envisioned system is done at a
very abstract level, which can be made an analogy to Newell’s knowledge level [9].

On this level, we talk of the users’ mental states instead of system states, of the users’
intentions instead of task decompositions, of task operations instead of interaction and
system features. As a result, this level allow us for representing knowledge about the
context of use in a way that can be traced back to the users’ problem-solving ability.
On the other hand, it makes possible provide opportunities and constraints for
This paper will appear in the Proceedings of the 10th International HCI Conference

interaction (presentation and manipulation) and system design without being limited to a particular set of design models.

The paper is structured as follows. Section 2 introduces the set of ontological terms. The conceptual models for interaction and system design are described in section 3. Section 4 demonstrates a case study with a simple, well-known hotel reservation example in order to articulate the rationale and concepts. Finally the conclusions are presented in section 5.

2 Ontological assumptions of problem-solving activity

In this section we propose a basic set of ontological terms for representing knowledge about human’s problem-solving activity. The terms we define are only our assumptions about content of that knowledge (i.e. types and their relationships) based on approximate theories developed in cognitive science and AI. Our purpose here is to provide a guideline for the elicitation of user’s task knowledge, rather than to develop a new type of mental model. In next section we will propose a framework of conceptual modelling by making use of these assumptions.

The notion of ontology is frequently used for representing knowledge that we share and reuse in the process of modelling some domain of interest. The notion stemmed originally in AI from the belief of a large body of knowledge in that domain to be the same in various applications. The essence of the notion can be found in [10]: an ontology is an explicit specification of a conceptualization. A conceptualization makes some assumptions about concepts and their relationships in the domain to be modelled. Such ontological assumptions determine the built-in terms offered by a conceptual model, and therefore its range of applicability [11]. In AI and software engineering, for instance, objects, events, processes, and relations are general terms for modelling domain knowledge.

Recently ontological assumptions have been proposed for modelling problem-solving methods in AI, e.g. [5, 12]. Although divergence on their terms, the rationale is the same: to establish assumptions of the knowledge structures used by a problem solver and of the control mechanisms to use those structures. For example, [5] defines ontology of problem solver and ontology of method. The solver ontology specifies problem-solving and domain states and their relations. A transition of problem-solving state under certain conditions on domain states defines a basic unit of problem-solving knowledge. The method ontology defines the structures of a collection of basic units for the control of problem-solving processes.
As mentioned earlier, however, the difference of human mind from the system lies on the capability of knowledge processing and representation. To make the system useful, domain knowledge would be derived from real scenarios of working situations because the system resources are limited. To make the system usable, however, the phenomenon of knowledge chunking has to be taken into account. Chunks are general and do not refer to the information content of knowledge. This implicates that they are a measure of the unrelated knowledge that can be processed naturally [4]. The phenomenon illustrates that the problem-solving knowledge processed by human mind is organized on a more abstract level than the level of domain factual knowledge. For the mediation of the two different levels, the system has responsibility for organizing contextual information about task-performing states, or domain states (how to present the information naturally is the task of interaction design) to meet the users’ needs. In the remainder of this section, we define a set of primitive terms in order to match the representation of the needs.

**Term 2.1. Domain object/action/state.** We apply the standard OOA definition (e.g. OMT [13]) for the three terms in general. In particular, we define that a (problem) domain state is a set of values of state variables representing objects in the problem domain.

**Term 2.2. Task object.** A task object is supposed to match a mental representation describing a problem domain state that a user believes or pursues. Task objects can have the same abstract mechanism and form as domain objects. For example, a task object can be specialized as an entity, relationship, or event dependent on whether it is autonomous, subordinate, or instantaneous, respectively. Task objects are essential for the problem-solving activity because they are a way of representing the user’s knowledge about the problem domain from a view of task. In other words, they represent the content of tasks to be undertaken. In the domain of a hotel, for example, domain objects can be room, guest list and so on, whereas a guest may have particular concepts about the domain states, such as availability, preference and so on, when she wants to make a reservation. It is also desirable to anticipate possibilities, to determine the current situation, and to remember working history by use of task objects. As a result, task objects provide information about how to represent knowledge about problem domain on the mental level.

**Term 2.3. Task operation.** A task operation is supposed to match a mental operation representing an input-output relation over task objects. Task operation applications
determine state transitions in the domain of task objects. Like a domain action, task operations can be characterized by pre/post, and trigger conditions that capture the elementary state transitions. A task operation can be specialized as required or requested dependent on if it changes domain states of task objects and generates an event or only queries the task domain.

Term 2.4. Problem state. The process of problem solving uses and creates and changes a number of task objects referring the states of the affairs. A problem state is a set of values of state variables (e.g. knowledge chunks) representing these task objects. Problem states include information about current goals. Problem states also include all information generated during the process of problem solving, such as beliefs, desires and so on.

Term 2.5. Goal. A goal is supposed to match a mental representation that the user has an attitude describing an expected problem state (e.g. ‘Make reservation’ is the goal in the above example). Goals are realized by intentions and evaluations. The important point is that a goal is some desired end state to be wished by the user, rather than a state to be reached after successful execution of a task in traditional task analysis. As a result, traditional analysis cannot answer if a goal can be achieved, and how failures can be recovered.

Term 2.6. Intention/Evaluation. An intention (or evaluation) is supposed to match a mental thread of problem solving and of maintaining problem states. A mental thread performs a set of required or requested task operations dependent on whether it is an intention realizing a current goal or an evaluation establishing a belief of current domain state, respectively. In general, mental threads change problem states, which in turn, invoke a set of domain actions to complete intentions or evaluations. Intentions (or evaluation) are characterized by trigger and stop conditions. As shown in Fig. 1a, intention/evaluation is significant in that it represents a basic unit of problem-solving knowledge, and establishes a continuum between the different views of task and domain.

Jarke [14] identifies four worlds that need to be understood and modelled, three of which are shown in Fig. 1b. In requirements engineering a number of researchers focus on the relationships between the three worlds, because they consider the users in the usage world as perceptual agents without cognition. For the sake of usability, we argue that the joint part of the three worlds has to be taken into account, that is, the states of the affairs among the worlds (Fig. 1c). We have identified the joint part
This paper will appear in the Proceedings of the 10th International HCI Conference

between the usage world and the subject world by the definition of the term *problem state*. We also argue that the proposed ontology could be used for a number of applications in various domains, such as knowledge management, information retrieval and so on. The crucial problem in those domains, in our vision, is how to capture the knowledge about what the users intend to do for the effective and efficient management and/or retrieval, rather than about what the users are doing.

3 Conceptual models on the knowledge level

So far we have defined a set of terms for the problem-solving activity. In this section, we propose a framework of conceptual models serving as a bridge from business models to architectural models (Fig. 2).

A conceptual model is a collection of concepts and their relationships, which embodies the view captured by a set of ontological terms with respect to some domain of interest, such as object-, process-, or goal-dominated. The essential features of any conceptual models are representation and understanding [15], although they have slightly different meaning in usage.

The proposed framework is intention-dominated, that is, it is concerned with what goals to achieve instead of how to achieve these goals. In AI terms, intentions are just a way of representing current goals. It is also different from the BDI paradigm [16] in that the framework has the capability of cognitive processing and representation.

As shown in Fig. 1, the business and architectural models (the dashed-line boxes) do not belong to the framework, but they are indispensable as the both ends of the bridge. The business process model (e.g. the business use-case model [17]) provides information about what to conceptualize, that is, it identifies context of usage in which the system will function. The architecture model, on the other hand, explicates why we need to conceptualize on the mental level because a user goal is a problem.
This paper will appear in the Proceedings of the 10th International HCI Conference

state, rather than a system state representing a system goal. Because of the limited space, in the remainder we will not mention them (see [18] for an task-level architectural model). The models in the framework are described as follows.

**Model 3.1. Intention model.** An intentional model consist of a goal the model intends to achieve and three interrelated parts:

- Intention specification is a cluster of intentions/evaluations without sequencing, in which intentions and evaluations realize the goal, and validate whether the goal is achieved, respectively. Computationally, a single intention or evaluation specifies an executing thread of performing a set of task and domain operations. It can be appropriately invoked and interrupted upon receiving feedbacks from the world. It can starts the sequence of its own, return the control, and remember problem states between activations. For interaction design, it provides information how to organize dialogue and presentation of user interface towards the current goal it realized.

- Task requirements. Task requirements are a collection of task operations invoked by the intention specification without sequencing. Task sequencing is determined by intention specification. A single operation as entirety cannot be interrupted once it is invoked. For interaction design, task requirements determine the actions that a user may initiate.

- Domain requirements. Domain requirements are a collection of actions invoked by the intention specification without sequencing. Domain requirements in general are independent of task requirements, although they are indispensable for system design.

**Model 3.2. Task model.** A task model is a collection of task objects that represent the content of tasks, and that represent all information that needs to be presented.
Computationally, the model represents a working memory that can be conceived of as the extension of the users’ working memory.

**Model 3.3. Domain mode.** A domain model is a collection of domain objects that represent domain factual knowledge, and it is therefore independent of the task model. In our framework, the model represents only the underlying information that a system should maintain for the purpose of functionality, and it is therefore, usually unperceivable by the users.

### 4 A case study

In this section we demonstrate a case study of hotel reservation. For simplicity, we assume that the users’ needs are to make a room reservation.

The modelling process starts from the artefacts of business process modelling. In general, the artefacts include a domain process model and a domain model. For the sake of the limited space, only the domain model in our example is show in Fig. 3a. It is notable that they only represent how a business actually works. However, they provide necessary domain knowledge for the process of conceptual modelling.

The exact sequence of steps of the modelling process is as follows:

1. Initial intention/evaluation identification

2. Intention/evaluation analysis

3. Task knowledge elicitation through intention/evaluation analysis

4. Elicitation of relationships with domain knowledge

*Until* all consistent intentions/evaluations have been established.

It can be seen that intention/evaluation analysis and task elicitation are complementary steps by analysis of the artefacts of business modelling and by asking the users for following questions repeatedly:

1. *What do you intend to do* when doing this thing?

2. *What do you expect to get* from doing this thing?

in which the *what* in general means the content of task (i.e. task objects) and the *do* (and the *get*) means task operations. The users can usually answer such questions because these questions are on the same level as their plans of guiding their tasks to be done. For example, a user may answer that ‘I want to know if a room is available in this period as I am making a call to this hotel’. From this answer we can find a task object ‘availability’ with properties ‘room’, ‘hotel’ and ‘period’, and an operation ‘query-availability’. The properties represent a set of state variables relevant to the
domain knowledge in Fig. 3a. It is notable that we aim at eliciting which intentions should be realized, rather than exploring how these intentions are realized by control knowledge as it does in traditional task analysis.

Fig. 3b depicts all the task objects and their relationships by repeating the steps of the process. It is not too difficult to establish the relationships between the identified intentions/evaluations (and the operations), and domain actions because each task object, more precisely, its properties make explicit references to domain objects, or their properties. For example, fig. 3c shows a specification for the intention of knowing availability and the evaluation of checking availability.

5 Conclusions

In conclusion, we are not developing a brand-new method, but reusing the traditional OOA technology on the knowledge level. This facilitates the reuse of designers’ expertise. The ontological terms we defined in this paper emphasize acquisition of knowledge about the content of task domain, rather than about control mechanisms in that domain. The continuum defined by these terms represents a declarative specification to mediate activities of problem solving and task performing. On the knowledge level, the proposed conceptual models can represent the requirements for a system in the same state space as the task knowledge, which makes possible build the system to delegate user tasks on the same level. The declarative representation of task knowledge also facilitates the reuse of that knowledge. Task knowledge can be in general applied for various problem domains.

References

This paper will appear in the Proceedings of the 10th International HCI Conference


13. OMT


16. Multi-agent
