From Pragmatism to Interaction Design: 
A Socio-Technical Design Space

Anders I. Mørch
InterMedia, University of Oslo, P.O. Box 1161 Blindern, N-0318 Oslo, Norway
anders.morch@intermedia.uio.no

Abstract. This paper presents a ‘translational’ approach to theory-based design cast within a socio-technical framework. The starting point is theory-based design in HCI and CSCW and sociotechnical systems (STS). Key works in theory-based design is presented and analyzed to identify strengths and limitations of this approach. STS is interpreted to mean the integration of social and technical components by the application of social science theories to interaction design. The motivation for this and a model for integration have been found in architecture. The notion of ‘externalized design’ of buildings is adopted as a model for how to incorporate external (non computational) objects in user interfaces. The framework is applied to the retrospective analysis of an interactive system the author has been involved in designing (Janus). The system was stimulated by the ideas of Pragmatism (reflection-in-action). A three-staged process provides gradual steps for translating reflection-in-action into a concrete interaction design: selection, appropriation, and translation. The paper ends by discussing strengths and limitations of the translational approach, and points out some directions for further work.

Keywords: Sociotechnical interaction design, externalized design, theory-based design, applied pragmatism, reflection-in-action, Janus, selection, appropriation, translation.

1 Introduction

This paper is about sociotechnical interaction design, which is formed in the intersection of Human Computer Interaction (HCI) and the socio-cultural approach to research [30]. The goal of the approach is to express specific theories of the latter in the concrete artifacts of the former, and to enhance communication of significant ideas of the past to students of today. This interest of mine was prompted by a retrospective analysis of two systems I have been involved in designing over a number of years (a design environment and a collaboration interface). These systems were inspired by ideas of Pragmatism, a branch of early American social science that began in the late 19th century and is associated with the works of Pierce, James, Dewey, Mead, and Schön, among others. Pragmatism is characterized by interdisciplinary and relevance to contemporary problems phased by society. For instance many of the early contributors were associated with multiple disciplines like
philosophy, psychology, sociology, education, and urban design. Within each
discipline and their intersections, they developed far-reaching insights, many of which
are relevant to contemporary problems studied in these fields. It is beyond the scope
of this paper to do justice to this work here (but I return to one theory later in the
paper). It has been the subject of extensive study over a number of years. This
includes discussing the relevance of Pragmatist ideas to contemporary problems in
society in areas such as education [2, 27b].

The critical reader might object to my use of theoretical ideas that originated in the
pre-web era, and are not directly related to the knowledge-based society we live in
today. At present, there are new forms of communication and information-sharing
emerging, which extends face-to-face (f2f) interaction, as well as new tools to reflect
online behavior. Some of this work will have the effect of rendering communities
more transparent than they have been in the past, which may require new theories and
models to be made. However, it does not mean that social science has to be
reinvented. Human nature is still the main component of social interaction. An
implication of this is that the significant theories developed by the American
Pragmatists are highly relevant today. This is manifest in the application of these
thinkers’ ideas to the problems society is facing today. An example is Dewey, Mead,
and Schön’ ideas on education [2, 27b].

Pragmatism has implications for practical action, collaboration, learning and
design, which are equally valid today as when the ideas were first introduced. These
theories contain insights that are useful to interaction designers in HCI, Computer
Supported Cooperative Work (CSCW), and Computer Supported Collaborative
Learning (CSCL), and there is plenty of opportunity for research in this area.

On the other hand, socio-technical systems (or STS) originated in an effort to study
organization design by recognizing the importance of interaction between people and
technology in the workplace. STS researchers study the various forms of
interrelatedness of social and technical sub-systems. The term STS was coined in the
1960s by Fred Emery and Eric Trist, who were working as consultants at the
Tavistock Institute in London. They have evolved the approach over many years, and
together with colleagues made important contributions to multidisciplinary studies in
organizations, e.g., [4], [20], [29] and participatory design, e.g. [23]. For example,
Bostrom and Heinen identified what conditions caused information systems design of
its time to fail and suggested to reframe the conditions in terms of STS to create a
better fit between the technical and the social sub-systems (Bostrom & Heinen, 1977)
[4]. STS was also influential (albeit in a different way) on the early work in
participatory design (Schuler & Namioka 1993)[25b], in particular on Kristen
Nygaard and his colleagues in Scandinavia. He contributed to STS by developing a
conceptual framework for collaborative design of information systems for developers
and users to work together [23].

In the context of this work, socio-technical systems are interpreted in the tradition
of participatory design and extended to mean the evolutionary creation of web-based
collaboration environments with a focus on the interaction between social and
technical components [33], which is interpreted to mean the integration of social and
technical components by the application of social science theories to interaction
design or in other words how theories from the applied social sciences (American
pragmatism) can inform the design of technical tools [21]. Furthermore, this paper
understands STS as a “design space,” bounded by source (theory) domain and target (technical systems) domain, within which artifacts are “transformed” by appropriation and translation. Appropriation is the adoption of ideas from the source domain into the target domain [7], and translation is the step-by-step work to “move” an object through the space, from the inception of a theoretical idea in the theory domain to a concrete instantiation in the technical domain in the form of Graphical User Interface (GUI) objects.

In sum, by socio-technical systems design we mean the appropriation of ideas from the applied social sciences to stimulate and guide (with affordances and constraints) interaction design and to communicate the ideas through interaction with the resulting designs.

The following research questions guide the work reported in this paper:

• What methodological support (guidance, model) is needed during the early (creative) stages of design before designers start to think in terms of technical (software) objects? What other fields of design can we learn from in this regard?

• How can we trace the development of abstract objects (theoretical ideas) into concrete (interaction) objects in terms of evolutionary design and user participation?

The rest of the paper is organized as follows. It starts by giving a survey of theory-based design in HCI and CSCW. Next, it presents the design process behind an award-winning chair in the Nordic design tradition to provide an example of creative appropriation in the early phases of the design process. This is followed by an example of ‘externalized design,’ a technique from postmodern architecture, which provides a model for how to incorporate external (non computational) objects into user interfaces. Based on this, a socio-technical framework is developed that breaks the first part of the design process into three stages (selection, appropriation, translation). This framework is applied to the analyses of an interactive system (Janus). The analysis reveals strengths and limitations of the approach. Finally, lessons learned and directions for further work are discussed.

2 Theory-Based Design

Theory-based design (TBD) of interactive systems is a design approach first proposed in HCI, e.g. [5]. The basic idea is that a theory, theoretical idea, conceptual model or a set of related concepts provides the starting point for a design process. What makes this controversial (and challenging) is that the theory should originate in a field outside of computer science, most notably cognitive science (the source of inspiration of early HCI), or the social sciences, which is more common today with the advent of new theoretical frameworks like Activity theory [15]. The phrase theory-informed design is sometimes used instead of theory-based design. I do not make a distinction between the two phrases here. The survey below focuses on two seminal contributions in HCI and CSCW: design principles for user interfaces and design process behind the Coordinator system.
2.1 Design Principles for User Interfaces

The term theory-based design in HCI can be traced to work done at IBM and the University of Colorado in the mid 1980s. Polson and Lewis [25] proposed a design based on psychological theories and cognitive models. The goal was to build a formal theory of exploratory learning that could assist in the design of easy-to-use interfaces. From the theory, they derived a list of design principles they could use to design user interfaces [17]. Four of the seven design principles derived from this theory are:

- Make the repertoire of available actions salient.
- Provide an obvious way to undo actions.
- Offer few alternatives (to prevent wrong moves).
- Model tasks that require as few choices as possible [17].

The success of these and related design principles have been useful for specific platforms. For example, Apple Computer and IBM created guidelines along these lines for supporting user interfaces within their GUI frameworks. These guidelines tended to be either very general and open-ended like those used in IBM’s 1984 Olympic Message System [11], or, like the Macintosh Human Interface Guidelines [1], they gave very detailed specifications for how to select (GUI) components, how components should look on the screen, and what behavior they should exhibit in applications. Critics at that time considered these design principles severely limited in scope and a passing fad in terms of impact, favoring consistency over creativity and exploration [12].

When looking back at the evolution of the field, the design principles movement in HCI gradually turned into research on evaluation criteria. The application of Polson and Lewis’ work is a good example of this. Their frequently cited cognitive walkthrough methodology is an evaluation technique rather than a design method, and it was developed on the basis of the above-mentioned theory and corresponding design principles [17]. Another influential evaluation technique for user interfaces is heuristic evaluation. This can also be traced to the design principles developed in the late 1980s [22].

In sum, strengths and limitations of design principles to support design of user interfaces are as follows:

- **Strength**: Design principles work best for specific platforms and GUI frameworks when the action space is large and there are expectations that user interfaces are consistent with respect to given standards and conventions.
- **Limitation**: The same set of design principles could be derived from multiple sources, each of seemingly legitimate origin, e.g., psychological theories, empirical findings (user needs), and informed guessing based on design experience. The importance of theory to support design was not adequately demonstrated.

Another approach to theory-based design is represented by the design of The Coordinator, a messaging system known for bringing theory-based design to CSCW. It addresses some of the limitations of the design principles approach.
2.2 The Coordinator

One challenge (as well as strength) in theory-based design is to connect two separate domains of discourse (theory domain and technical solutions domain) without “mechanically mapping” elements from one domain into the other [6]. The design space should be kept open to encourage multiple paths to solution. The Coordinator system [32] is arguably the best-known early system in this tradition. It has theoretical roots in speech act theory, which is a theoretical framework for the study of human communication proposed by Searle [26].

Speech acts are operationalized in the Coordinator in the form of templates (sentence openers) for managing the different conversation acts (generating, transmitting, storing, retrieving, and displaying messages). The user interface supports this by providing menus for selecting what actions the user may take at any point in a conversation. In many ways it resembles email structured according to message types. For example, the menus for responding to requests include the items: acknowledge, promise, counter-offer, decline, and report-completion [32].

Ordinary email does not support message categorization in the way the Coordinator does. Some critics say that by requiring explicit structuring of a phenomenon that in the outset is subtle (i.e. transition between illocutionary acts), the Coordinator changes the nature of communication [28]. Proponents reply that Searle’s conversation structure can instead be found inside the body of text messages (i.e. reveal the conversation structure at a different level of abstraction, for example, separated by punctuation and spacing). An advantage of the Coordinator over ordinary email is that it allows message recipients to view the conversation structure when browsing previous messages. Communication situations relying on visualization of complex conversation can benefit from this approach. However, the upfront cost of breaking a complex conversation into conversation chunks may not be worthwhile when considering all factors, because those who benefit are not those who do the work. Previous studies have shown that such a dilemma is likely to cause a system to fail or gradually be subverted [18].

In sum, strengths and limitations of the “conversation model” approach to design of collaboration systems are as follows:

- **Strength**: Several successful collaboration systems in CSCW and Computer Supported Collaborative Learning (CSCL) have been designed based on this model, using templates and sentence openers to organize online conversation.
- **Limitation**: Task-specific collaboration (e.g., student communication in web-based forums) is highly determined by local constraints. Predefined conversation types might be in the way because they do not take the local constraints into account. As a consequence they do not make sense to some students [18].

3 Externalized Design

In the previous work there was no rationale for the approach (i.e. why TBD is important), and the stages passed through when translating theoretical ideas into concrete designs (theory adaptation) was not identified, discussed, or problematized.
Furthermore, local constraints were not brought into the environment. In the remainder of the paper we address these issues by developing a model of theory-based design based on creative practices associated with two design fields that interaction design can be compared with: furniture design and building architecture [8], [14], [22].

3.1 Furniture Design

The design process of chairs is characterized by integrating creativity with utility. Utility (usability, usefulness, and domain-specific needs) is as important to furniture designers as it is to interaction designers. For example, a chair that is uncomfortable will not be used, but one that is comfortable will be. More importantly, and as result of the abundance of comfortable chairs in the world, furniture designers have to bring innovation into their designs to succeed in competition with fellow designers. The Nordic designer Olav Eldøy explained the role of creativity as the first and most important step of the following three-step design process (simplified for illustration):

• Find a recognizable idea that can be expressed in physical form.
• Balance creativity against utility.
• Provide a construction that affords production and export.

All phases were essential in the design of his award winning Peel chair. The inspiration for this chair was orange peels falling to the ground (Figure 1). This turned out to be a realizable idea. The result can be judged by the degree of resemblance between idea and physical form and color. On the other hand, finding a recognizable idea that can be expressed in computational form is not commonly associated with interaction design. However, there is no intrinsic reason why it should not also be applicable here. Another example of the same phenomenon is described in the next section by distinguishing notions of inner and outer language.
Fig. 1. Peel chair (2002). Orange peels falling to the ground inspired this design.

### 3.2 Postmodern Architecture

Postmodernism architecture is associated with architects such as Michael Graves [24], [31]. One of the buildings he is known for is the Portland Building in Oregon, USA (Figure 2). This building was one of a few that instantly became an icon for postmodern architecture. It is distinguished from the nearby buildings by external decoration and small cubic windows. The effect of this is a mix of modernism with older styles in an overall modernistic design. Graves contrasted it with modernist architecture in the following way:

“While any architectural language, to be built, will always exist within the technical realm, it is important to keep the technical expression parallel to an equal and complementary expression of ritual and symbol. It could be argued that the Modern Movement did this; it expressed the symbol of the machine, and therefore practiced cultural symbolism. But in this case, the machine is retroactive, for the machine itself is a utility. So this symbol is not an external allusion, but rather a second, internalized reading. A significant architecture must incorporate both internal and external expressions. The external language, which engages inventions of culture at large, is rooted in a figurative, associational and anthropomorphic attitude.” [31, p. 11].

Fig. 2. Portland Building (1982). Reflections of elements in the local surrounding are embodied in the façade.

The Portland building makes a distinction between external (artistic, symbolic) and internal (technical) expression. The “external language” he refers to consists of reflections of elements in the local surroundings, which are literally embodied in the building’s façade. It consists of symbols professionals can relate to (e.g., small cubic
windows on a light-colored background makes one think of the Bauhaus, whereas the blue ribbons have an artistic or non-functional association).

The combination of expressions from a non-technical (external) language with the concrete material of the building forms the hallmark of the postmodern approach to architectural design. The two kinds of languages (internal and external) have direct implications for user interface design. A user interface is defined by an internal language (programming) as well as an external language (interaction design), although the latter is often not thought of as such. However, from the point of view of theory-based design, one can think of this in terms of two domains of discourse— one associated with the technical system and the other with socio-cultural theories. It is the expression of the latter into the former that can be seen as equivalent to the externalized design of buildings.

4 Translational Approach to Theory-Based Design

In order to adopt externalized design for interactive systems, the following working hypothesis is made. In the same way as nature, local surroundings and symbolic association have inspired architects and materialized in the built environment (physical objects); theories, concepts, and abstract notions ought to provide the same kind of inspiration for interaction designers (computational objects). This claim is based on the following differences between computational and physical objects:
- **Computational objects** like software components are abstract compared to physical objects like chairs [16]. For example, software components are defined in terms of program code in addition to being interactive objects on a computer screen.
- **Theoretical ideas** and related conceptual models have been important to the success of many innovative user interfaces (see Section on TBD).

A socio-technical conceptual framework is presented below. It adopts the “architectural model” presented above and was further inspired by the retrospective analysis Carroll and Kellogg [6] performed to identify the “myriad of claims and their interrelations” embodied in the Training Wheels and HyperCard interfaces to determine how the claims were given coherence by being codified in software. They used the term “psychological claim,” instead of a theoretical idea. A socio-technical approach puts more emphasis on cultural tools [30] and conceptual artifacts [3] than on cognitive artifacts [22], [6]. In spite of this, cognitive artifacts have been important in understanding the design of interactive systems, and re-conceptualizing them in terms of cultural artifacts will bring a social dimension to that line of work.

The following stages define gradual translation and serve as intermediate abstractions for talking about the early (creative appropriation and theory adaptation) stages of interaction design:
- **Selection:** Any source of inspiration one wishes to explore for the purpose of realization into physical (computational) form. A criterion for selection is to be able to communicate the idea to others (designers and users). However, there are no intrinsic reasons for prohibiting certain ideas. In the work presented here, selection is associated with theories, chosen from a domain outside computation and cognition (Pragmatism).
• **Appropriation:** From a socio-cultural point of view, appropriation has been defined as “the process of taking something that belongs to others and making it one’s own” [30, p. 53]. According to Wertsch the path to appropriation is not always straight and smooth, but sometimes involves tension between what we appropriate and the way we use it within a particular context. Appropriation is the act of breaking down an idea so that it will stand out in a contemporary design context, and restructuring the elements to make a concrete design possible without distorting the conceptual integrity.

• **Translation:** This is arguably the most critical step, as it is the most revealing. It involves turning appropriated material into concrete design. An example is the translation of a conceptual artifact (e.g., key words) into a physical or computational artifact (e.g., GUI objects). There are multiple ways of accomplishing this, some better than others. To avoid ‘Kitsch design’ (poor translation, misunderstanding of original idea), evaluation criteria for assessing the quality of a translated idea are necessary. In the remainder of this paper this conceptual framework is applied to analyze the theory adaptation stages of the design process of a computer application in order to illustrate the translational approach with a concrete example.

4 **Retrospective Analysis**

The system presented below has developed over a number of years and have been written about in the literature, but it has not been profiled in terms of sociotechnical interaction design. The theoretical idea that inspired Janus was reflection-in-action [27].

4.1 **Reflection-in-action and Janus**

The Janus system [9], [10], [19] is a design environment for kitchen design. It was inspired by Donald Schön’s theory of reflection-in-action [27]. The end result was the integration of two sub-systems (for construction and argumentation support) with a design critiquing system.

**Selection.** Reflection-in-action [27] can be described as “thinking on our feet.” Schön has examined this largely unarticulated, improvisational process in a study of practitioners in a variety of professional domains to identify how they explore design spaces and “communicate” with the domain materials for inspiration. It involves looking to our experiences, connecting with our feelings, and attending to our working theories. It entails building new understandings based on these things to inform our actions in new situations. From this perspective, the knowledge inherent in practice is to be understood as artful doing. In Schön’s own words:

“In a good process of design, this conversation with the situation is reflective. In answer to the situation’s back talk, the designer reflects-in-action on the construction
of the problem, the strategies of action, or the model of the phenomena, which have been implicit in his moves.” [27, p. 79]

**Appropriation.** Schön was primarily interested in developing a descriptive account of design activities, illustrating and explaining what designers do, identifying the importance of human collaboration in this process, and arguing for educational changes on this basis [27b]. Therefore, his ideas do not lend themselves to operationalization in an interactive system design. His concepts must be further interpreted and broken down into manageable chunks before they can be experimented with in terms of computer support. We considered “action” and “reflection” as the basic activities of a reflective practitioner and to form the basic components of computer support for reflection-in-action. For reflection support to be part of reflection-in-action, however, it needs to be brought to the designers’ attention during the “action present,” and to provide answers to the situation’s “back talk.” In this respect, Schön’s notions of “action present” and “back talk,” intimately connected to reflection-in-action, have been interpreted to mean automated feedback from the work area the designer is interacting with immediately after an operation on a design has occurred. This is analogous to how a human design critic stands behind the shoulder of a student in a design studio and gives feedback on work in progress.

![Diagram of Janus](image)

**Fig. 3.** Janus provides computer support for action and reflection with “back talk” triggered by critic messages.

**Translation.** Janus consists of two separate interfaces, one supporting action (construction) and the other supporting reflection (argumentation), as seen in Figure 3. During construction, designers select “design units” from the palette and place them into the “work area.” The critiquing component links construction and
argumentation. Critics provide automated feedback, “critique messages,” as shown in the lower right part of Figure 3 (Janus Construction), and they operationalize Schön’s notion of “back talk.” The “back talk” of the situation depicted in the screen image in Figure 3 tells the designer that the ‘work triangle’ is greater than 23 feet. This may trigger reflection on how to incorporate a design recommendation into the design currently under development. The left screen image of the Figure shows the argumentation interface. It is an early hypertext system based on the IBIS design methodology [18b]. It is structured as issues, answers, and arguments, and represents the design rationale behind kitchen planning principles, and in this case shows a discussion of the various pros and cons of the work-triangle concept.

Table 1 summarizes the appropriation and translation of the theoretical idea reflection-in-action [27] into a concrete interaction design (Janus interface).

<table>
<thead>
<tr>
<th>Steps</th>
<th>Janus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection (motivating idea)</td>
<td>Reflection-in-action (Schön, 1983)</td>
</tr>
<tr>
<td>Appropriation (design context)</td>
<td>Action, reflection, action-present, back-talk</td>
</tr>
<tr>
<td>Translation (interaction objects; user interface configuration)</td>
<td>Work area, design units, critic messages, argumentative hypertext (see Fig. 3)</td>
</tr>
</tbody>
</table>

In sum, strengths and limitations of the “translational approach” to interaction design are as follows:

- **Strength:** Identifies “appropriation-time” as the first stage of the lifecycle of an interactive system before conventional design-time and use-time activities begin. This relationship is displayed in Figure 4. This is another stage where end-user developers (amateur computer scientist) can participate (they can also participate at use-time). Requires knowledge of two domains of discourse (theory domain and technical systems domain).

- **Limitation:** Externalized design in architecture have in some cases been criticized for incorporating external expressions that are not considered successful design elements, leaning towards Kitsch (poor translation, misunderstanding of original idea), which for theory-informed interaction designs might mean unrecognizable ideas and little impact on design. Successful results require belief in the approach and proficiency in two domains of discourse, which is demanding.

Fig. 4. “Appropriation time” as the first stage in the design process of a theoretically inspired interactive system.
5 General Discussion and Directions for Further Work

The research questions raised in the beginning of this paper were:

- What methodological support (guidance, model) is needed during the early (creative) stages of design before designers start to think in terms of technical (software) objects? What other fields of design can we learn from in this regard?
- How can we trace the development of abstract objects (theoretical ideas) into concrete (interaction) objects in terms of evolutionary design and user participation?

Related to the first question is the open issue as to what extent there should be selection criteria for choosing theory/idea to be appropriated in the first place. I have not advocated selection criteria in this paper, only to strive for the goal of being able to instantiate and communicate the idea to others (designers and users). Another criterion is motivation: To select an idea that you believe is important to understand and put into a concrete form (other than writing) for others to learn about. A third is criterion is quality of outcome. How can we identify measures that can help to assure the end product is of a certain quality, and if any of these can be associated with a selected theory domain. The notion of critiquing system can be useful here. Critics can be associated with various source domains and can provide feedback on the application of the theoretical ideas to a technical systems domain.

Related to the second question in terms of evolutionary design, I have explored the integration of two kinds of artifacts that belong in disparate worlds. In terms of the philosopher Karl Popper, they are World 3 (conceptual artifacts) and World 1 (physical and computational artifacts). I have shown how one can move from World 3 to World 1, expressing the abstract ideas of the former world in the material of the latter. This should be thought of as a form of externalization [15]. Another of our long-term goals is to understand the transition from World 1 (external expressions) to World 2 (mental representations), which can provide a model for internalization and learning. Whether or not there is a connection from the work presented here (W3 -> W1) toward that end (W1 -> W2) is an open issue outside the scope of this work.

When the second question is addressed from the point of user participation, the approach proposed can be compared with related (socio-technical) approaches in HCI and participatory design (PD), including contextual design [13] and user participation in system development [23]. Whereas many researchers in HCI have taken inspiration from methods originating in the social science in order to improve the design (usability and usefulness) of interactive systems they have had varying degree of success when it comes to integrating technical and social components. Many times the process will stop halfway and not be able to reach to software engineering without starting the process all over, or it has a too ambitious goal at the outset regarding the expectations of user participation. For example, Holtzblatt [13] makes use of a range of techniques to involve users in the design process. These techniques have many features in common with PD techniques (paper prototyping, informal work models, story boards, diagrams, mock-ups, use cases, etc.). On the other hand, it is difficult to trace the influence of these techniques when analyzing the final results, and often the design process of PD ends with diagrams and conceptual models, but uncompleted systems. Nygaard’s [23] approach to integrate PD and system development attempted
to create a comprehensive organization-inspired model of object-oriented systems together with end users. That puts high demands on voluntary user participation. In this work I have a much less ambitious goal because I do not do not address the entire system, only the user interface, since it is the component that will have the biggest impact on end users. The ultimate goal of the approach, which is different from the past work, is to make the GUI resemble or at least be traceable (accountable) to a set of theoretical ideas originating in the social sciences. Whether or not this has been achieved in the current work is up to the users (and readers) to judge. As a final note it should be mentioned that I do not imply that other components of an interactive system (other than user interface) should be excluded from a sociotechnical approach to design. For examples many researchers in HCI, CSCW, and education consider use contexts and activities as key to understanding what should be designed. On the other end, software engineers are often more concerned with the internal components of a system. To the extent these components (external and internal) define links to the human computer interface component (i.e. can be reached by end users) they may also benefit from theory informed design along the way proposed in this article.

Open issues for further investigation based on the current work is as follows:

- The translational approach and the design principles approach have many similarities. For example, educational technology developers have treated a concept like “scaffolding” as a design principle in instructional design, which has evolved from a theory in educational psychology. Does that mean that the translational approach should provide intermediate stages in the form of design principles as a normative adaptation of an idea rather than starting directly with the original idea (e.g., reflection in action)?

- Complex ideas like reflection-in-action offer affordances and constraints for design (as well as motivation), and the analyses show that the final design of Janus (user interface components, configuration and functionality) could be traced back to the theory of reflection-in-action. Can similar or better results be achieved if other developers were asked to do the same exercise, i.e. starting with the same basic idea and express it in different interaction designs? My tentative answer is that these designs will be variations of a common theme already established.

- Retrospectively analyze other interactive systems and concrete designs in general to identify how they have elements that can be traced back to ideas originating in domains outside of design.

- Use theoretical ideas like reflection-in-action (e.g. as developed by the scholars mentioned in this paper) to design interactive systems and other forms of instantiations that can be evaluated for usability, utility, and education.

Acknowledgments. The author thanks Gerhard Fischer and Raymond McCall at University of Colorado, Boulder who contributed to the design of Janus and suggested Donald Schön’s concept of reflection-in-action as inspiration. Colleagues and students at InterMedia, University of Oslo provided constructive feedback on previous versions of this paper. The writing of the paper has been supported financially by the Knowledge Practices Laboratory (KP-Lab) project (WP3), European Information Society Technologies (IST) program, under contract no. 27490.
References


