Evaluating Methods for Community IS Development[1]

Aldo de Moor
Infolab, Dept. of Information Management
Tilburg University, The Netherlands
e-mail: ademoor@kub.nl

Abstract

The information systems of virtual professional communities are prone to much complex change. Community information systems development (CISD) methods can facilitate their evolution. We present a framework for CISD method evaluation. Starting point is Preece's user-centered approach to CISD that focuses on achieving usability and sociability. To ground CISD in existing user-centered methods, we introduce an analytical framework based on the development criteria of user control and legitimacy. Using this framework, we argue that CISD requires legitimate user-driven methods. We illustrate the framework by positioning some user-centered methods.

1. INTRODUCTION

As instruments of collaboration, virtual communities are gaining in importance. Many examples exist, for instance in the research, healthcare, and e-business domains. We define the *virtual professional communities* in which this joint work takes place as communities of professionals whose collaboration on activities required to realize shared goals is mostly or completely computer-enabled [1].

Virtual professional communities are typically supported by *community information systems* that are composed of standard information tools like e-mail and web applications. Together, community and tools form a complex socio-technical system in which change plays an important role: it is continuous and requires a co-evolution between the social and information system [2].

Change is *continuous*, as there are many social, political, technical, and organisation forces that act as change drivers [3].The social and technical systems can be studied separately, but are inter-dependent. The social domain is where the requirements originate in the form of business processes and organisational structures. The technical domain consists of the functionality provided by the information tools. Requirements can be mapped to various tool configurations, whereas individual tools may support many different requirements. Changes in either requirements or functionality thus lead to complex dependencies. This results in the need for a *co-evolution* process, in which the effects of changes are carefully explored.

It is clear that community information systems development (CISD) requires well-defined *methods* that allow for these complex change processes to be systematically supported. For traditional waterfall information systems development methods, clear design criteria have been defined [4]. However, design principles for community information systems development are still unclear [5]. In this paper, we consider CISD methods to belong to the class of user-centered IS development methods. Our goal is to classify existing user-centered methods and see how they can contribute to the development of more structured CISD methods. We first clarify the concept of CISD in Sect.2 and then present our framework for user-centered method analysis from a community perspective in Sect.3. To illustrate, we position several existing methods in this framework in Sect.4. We end the paper with some conclusions.

[1] Proc. of the Seventh CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD-2002), Toronto, May 27-28, 2002, pp.152-159

2. COMMUNITY IS DEVELOPMENT

Although community is such a core concept, there is much confusion about what it actually means. To better understand the essence of a community, it helps to look at the etymology of the word. *Community* stems from the root ‘communis’, which itself is composed of two groups of Latin words: ‘cum’ (together) and ‘munis’ (obligation), or ‘cum’ and ‘unus’ (one) [6]. Many definitions exist, but they are all based on these roots. An example is the definition of Talbott, who says that a community is “a group of people *bound* together by certain mutual concerns, interests, activities, and institutions” [7].

2.1 Towards a Method

One of the most comprehensive attempts at a CISD method is the work done by Preece on community-centered development [8]. She starts with two main design criteria for successful community information systems: *sociability*, which focuses on social interaction, and *usability*, aimed at human-computer interaction. More precisely, sociability is concerned with the extent to which the social policies incorporated by the information system support the purpose of the community and are understandable and acceptable to its members. Usability is to ensure that community members can perform their purpose-related tasks effectively, intuitively, and easily.

Usability is necessary for all information systems, and focuses on individual human-computer interaction. Sociability, however, is the key concept that distinguishes community information systems from other types of information systems. Sociability is not permanent, however, but will change as the community evolves [8].

The *process* of community-centered development, according to Preece, should focus on the community's needs prior to making decisions about the technology and social planning. The process consists of two main parts: software design by tool selection and tailoring; and sociability planning. Development is to proceed iteratively, with a strong focus on user-centered design, contextual inquiry, and participatory design. User-centered design focuses on activating users in the design process, contextual inquiry stresses the importance of understanding the user context, and participatory design advocates strong user and community participation in the design process.

To operationalize this process, Preece gives valuable guidelines and techniques. However, these take more the form of heuristics than a systematic methodology. Furthermore, already many proven user-centered methods exist that could help in making the CISD process more efficient and effective. User-centered design is the aim of a wide range of approaches, such as participative/participatory design [9, 10], cooperative design [11, 12], joint-application design [13], and customer-centered design [14]. In participatory design, for example, end-users, in conjunction with developers, explore the possibilities and limitations of information tools for specific work practices [10].

The question now becomes: how to interpret existing user-centered methods in terms of communities? What role can they play in the CISD process? To answer this question, some way of classifying user-centered methods from a community perspective is needed. The next section presents the outline of a framework that can be used to position, analyze, and improve such methods for the purpose of CISD.

3. A COMMUNITY PERSPECTIVE ON USER-CENTERED METHODS

Preece’s design criteria are necessary, but not sufficient for CIS development. Basically, they belong to the *use dimension* of the information system. However, what is lacking are criteria for the *development dimension*. Whereas the first dimension focuses on guaranteeing certain community qualities of the system in operation, the development dimension should ensure that the *evolutionary process* of the socio-technical system enabled by the CIS matches the interests of the community.

3.1 Development Criteria

Two important *development criteria* are user control and legitimacy. They concern the *way* in which community members are involved in the design process and the *selection* of which members to involve in a particular kind of changes, respectively.

3.1.1 User Control

Despite the use of different terminologies, the philosophy underlying all user-centered approaches promote a more or less active role of users in the analysis and design, in other words in the specification of their own information systems.

The first development criterion for classifying user-centered methods therefore is that of *user control.* It concerns the role that users play as modellers of specification knowledge. The modelling roles can be mostly or completely played either by external analysts (as in traditional methods) or by the users themselves. We call the first category *user-assisted*, and the second category *user-driven* development methods.

In user-assisted methods, users play a rather passive role, mostly being the source of the specification knowledge, not their interpreter. In user-driven systems development, however, users, instead of only being consulted, are considered to be true partners in the design process [15]. First, they have access to the subtle tacit knowledge suffusing the community, often unavailable to outsiders, and, second, they are the ones who face the breakdowns in work that lead to new change processes. Also, it has been shown that approaches in which users are actively involved generate more requests for modification, as users have a greater awareness of the system potential, leading to better perceived systems quality [9]. Finally, studies of end user computing have shown that giving end users more control over the development of their own applications, with IS staff only playing an assisting role, is often successful [16]. This is especially the case in professional communities, with their use of standard information tools, subtle specification changes, and knowledgeable, committed users.

3.1.2 Legitimacy

Sociability was identified as a key use criterion that determines whether the information system operates successfully from a community point of view. However, this concept has to do with whether the IS *in use* supports the social policies of the community. As Preece indicates, the sociability of an IS may change over time [8]. In dealing with this change, handing over control to end users does not automatically mean that the interests of the community are best served. To ensure that the information system *remains* a community system, the role of users as modellers needs to be guided by another development criterion: *legitimacy* [1, 17]. This entails that any specification change must first be *meaningful* in that its semantics are well-understood within the community. Second, this change needs to be *acceptable* to the community, implying that those users to whom the change is relevant must agree before it is implemented.

We call systems development approaches that focus on obtaining specifications from individual users *individualistic* approaches, and those that concentrate on the meaningfulness and acceptability to the community as a whole *legitimate* specification methods.

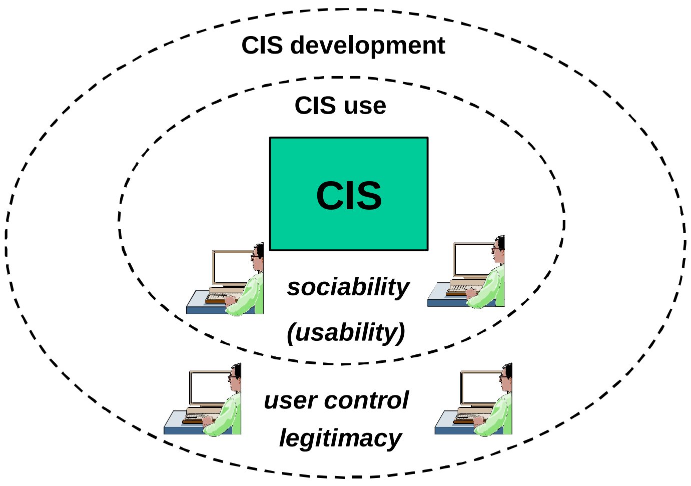

Fig..1 Use and development criteria of CIS

Summarizing, sociability and usability are *use criteria* that capture the fact that the information system in operation is to serve the community, while user control and legitimacy are *development criteria* that are to ensure that the evolution of the information system is grounded in the community (Fig. 1).

3.2 The Development Process: Grounded in the Neo-Humanist Paradigm

The question becomes how to theoretically ground a legitimate user-driven CISD method *process* so that it ensures that the development criteria are met. One appropriate starting point is the neo-humanist paradigm, which holds that knowledge is socially constructed in a process of human interaction, and that there is a natural tendency towards change and conflict [18, 19]. This paradigm is especially suited for modelling CISD, because in virtual professional communities, stakeholders with many conflicting interests need to work together to construct their own models of their work processes and supporting information technologies.

The core idea of neo-humanism is its focus on *emancipation*, which is the process in which pseudo-natural constraints on the realization of human needs and potentials are removed by conscious attempts of human reason. Pseudo-natural constraints are those that seem natural, but in fact are caused by communication distortions. Some of the most common causes of these distortions are authority and illegitimate power; peer opinion pressure; time, space, and resource limitations; social differentiation between actors; and bias and limitations of language use [18].

In neo-humanist system development approaches, the removal of such communication distortions is to be achieved by enabling what is called *rational discourse*, in which claims made throughout the system development process are critically evaluated. Some form of rational discourse should thus be incorporated in any CISD method.

4. CLASSIFYING EXISTING USER-CENTERED METHODS

We are now able to present the outline of a framework for the analysis of user-centered methods from a community perspective. Fig.2 combines the two development criteria in a matrix that can be used to classify these methods. In the limited space available, we can only give a flavour of how to analyze and classify user-centered methods, a more in-depth analysis is given in [20]. In this section, we will illustrate the framework by briefly analyzing four (categories of) methods: prototyping, tailorable tools, socio-technical methods and RENISYS.

legitimacy

		individualistic	*legitimate*
user control	*user-assisted*	**prototyping**	**socio-technical methods**
	user-driven	**tailorable tools**	**RENISYS**

Fig.2 Analyzing User-Centered Methods from a Community Perspective

4.1 Prototyping

A *prototype* contains some of the core functionality that the information system should have, although this functionality has not yet been fully specified and implemented. Prototyping is done because most users find it hard to state their requirements without being able to evaluate an actual piece of software [21]. Based on informal discourse with a user, the analyst refines the specifications, which form the basis for a new prototype. This process is repeated until the user is satisfied.

This is an example of a *user-assisted/individualistic* development method. The key person controlling the development process is the analyst/programmer, who often works with a single or small group of users, but does not take into account whether the prototype is meaningful and acceptable to the community as a whole.

4.2 Tailorable Tools

Tailorable tools are tools of which end users themselves can modify considerable parts of the functionality in its context of use. A typical example of such a tool is Oval, in which users can create a wide range of integrated information management and collaboration applications by combining a small set of functionality primitives like objects, agents, views, and links [22].

Such tailorable tools are examples of *user-driven/individualistic* development methods. The focus is still the individual user, but she has much more control over the development process than with prototyping.

4.3 Socio-Technical Development Methods

Socio-technical development methods pay much attention to users in their organisational context and adopt a socio-technical perspective on information system development. From such a perspective, as discussed before, an organization is considered to consist of an interdependent social and technical system. An important objective of socio-technical approaches is to elicit *variations* in perspectives of users on roles and objectives, rather than forcing them to accept an artificially uniform reality . Two of the most widely used approaches are Soft Systems Methodology and Mumford's ETHICS. Related approaches have been developed in the so-called Scandinavian School, which strongly emphasizes the participation of workers in system design, thus democratizing the whole organization [23].

These methods are examples of *user-assisted/legitimate* development methods. Users play relatively modest modelling roles, as the overall control of the - often lengthy - development process is still in the hands of external analysts. However, the users are not consulted in isolation, but they are considered to be stakeholders representing different interests. There is a strong focus on ensuring that requirement specifications are understandable and acceptable to the community as a whole.

4.4 RENISYS

Previously, we argued that CISD methods should be legitimate and user-driven and that the systems development process should take place in the form of a rational discourse. One approach that incorporates these principles is the RENISYS method.

The RENISYS (**RE**search **N**etwork **I**nformation SYstem **S**pecification) method supports virtual professional communities in the handling of breakdowns in their collaborative work [1, 20]. The method allows individual users who have become aware of a problem with either the way their work is organized, or with the support provided by the enabling technologies, to formulate their problems in terms of problematic knowledge definitions, using a knowledge base of ontological and state knowledge. The method then determines which other users are to be involved in the resolution of these definitions. To this purpose, the *composition norms* that define the acceptable specification behaviour of community members (or stakeholders) play crucial important role. An example of such a norm would be that editors are permitted to initiate the modification process of editorial workflows. The method calculates *the resultant deontic effect* of the set of composition norms that apply to a particular user and the specification process stage required to change the definition. In this way, it knows which users to legitimately invite for the *conversation for specification* (i.e. the discussion to modify the editorial workflow) in which the problematic knowledge definition can be changed.

Fig. 3 shows the specification process as a situated conversation for specification. This particular conversation concerns the modification of the (ontological) type definition of the editorial process. Each conversation consists of an initiation stage, an execution stage in which a knowledge definition proposal is made, and an evaluation stage in which the proposed definition is approved. For each community member and conversation stage, a set of *applicable norms* is calculated, which is a subset of the total set of legitimate composition norms. One such set, for example, contains the norms that apply to user John and the execution of the editorial type modification process. Since applicable norms may have different deontic effects (indicating whether a user is permitted, required, or prohibited to be involved), the *resultant deontic effect* is calculated. For John, this could mean that he is prohibited, i.e. may not, take part in this process, according to the community norms. In this way, the *relevant user group* is determined, defining which community members to legitimately involve in a particular change process. In this case, only Jane would be in the group of members who have to participate, for example because she is an editor.

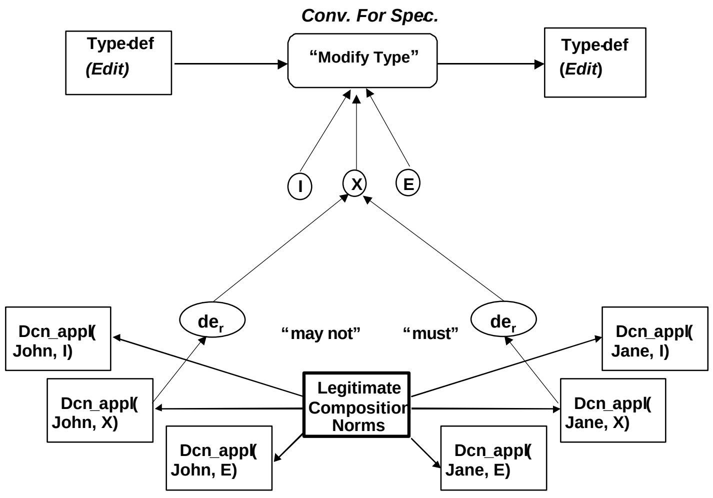

Fig.3 The RENISYS Conversation for Specification

RENISYS is an example of a *user-driven/legitimate* development method. It is user-driven in the sense that users have complete control over the specification process, which they can initiate, execute, and evaluate. The specification process is made *meaningful* by the use of community-defined ontologies to define its specification knowledge, and *acceptable*, by using composition norms to determine which community members to involve in the specification process. This process itself takes place in a form of rational discourse, in that conversational roles and moves are well-defined and that knowledge definitions proposed or made can always be challenged in a discourse process to clarify and defend them. To our knowledge, RENISYS is the only user-centered method for rational specification discourse so far that combines this strong focus on user-drivenness and legitimacy.

Case: Electronic Law Journal

We will clarify this approach with an example drawn from a case on the development of an electronic law journal. We will only give an informal description to get the gist of the approach, more details can be found in [20]. In the case, the method was applied in a manual way, by explicating composition norms and organizing live meetings with participants. The RENISYS web server implementing the method consisted of a prototype only. With a robust version, however, virtual conversations for specification could have been organized in a similar way, necessary in the case of geographically distributed working groups, for instance.

Workflows defined by the community in the electronic journal case included the paper submission, review, and editorial process. The available software consisted of such diverse information tools as BSCW (advanced file management system), FTP, mailers, mailing lists, and, of course, a web server and web browsers. At its initiation in 1997, the system was in a continuous state of flux. Community members, such as the editors and test authors, and implementors (librarians, computer centre staff) needed to define the workflows and map them to the available tools. As it was unclear who was responsible for defining which workflows and mappings, the RENISYS method was applied to systematically manage the handling of the specification proposals emerging from the user community.

The project team had given a technical committee the responsibility to define any workflow – tool mapping processes (via a *composition norm*). The technological committee then proposed to use an FTP server for the support of the paper submission process (creating a *mapping knowledge definition*). However, the project coordinator was permitted (via a *composition norm*) to evaluate any proposed mapping definition. He had problems with the proposed FTP server for paper submission, as he favoured the use of the BSCW-web server, which has been optimized for advanced and secure file transfer. In *a conversation for specification*, they discussed this issue. However, the technological committee had its own objections to this proposal, as, certainly in those early Web days, they considered the BSCW tool too complex to learn for the average user. No agreement could be reached. Still, previously another *composition norm* had been defined by the community, which stated that yet another actor, the project team, was permitted, although not obliged, to take part in any specification change. So far, the project team had not actively participated in the conversation for specification. However, since the *specification discourse* that had emerged could not be resolved, it stepped in and played a mediating role. It was agreed that the BSCW server would be used for paper submission, but great care would be taken to facilitate the learning process by FAQs and personal instruction sessions for submitting authors.

5. CONCLUSIONS

In this paper, we investigated methods for community information systems development. We saw that virtual professional communities are prone to change and have co-evolving social and information systems. To support the complex process of community information system evolution, sophisticated development methods are required.

One promising step towards advanced community-centered development methods is made by Preece. She has shown that usability and especially sociability are key dimensions of successful community information systems, and that their development methods should be user-centered with a high degree of user participation. However, there are numerous user-centered methods already in existence and it is not clear how to identify which ones should be used for community information systems development. We thus presented a framework for the analysis of user-centered IS development methods from a community perspective. To illustrate its use, we positioned some existing user-centered methods in our framework.

Like Preece, we consider usability and especially sociability to be key use dimensions of community information systems. In addition, we introduce the development dimensions of user control and legitimacy. Based on these dimensions, we argued that methods appropriate for community IS development need to be both user-driven and legitimate. We do not claim that user-drivenness or legitimacy are equally important in every community, or for every type of specification change within one community. For instance, within an academic e-journal community, some design decisions involve less community members than others. Obviously, acceptable changes to the editorial process may involve only the editors, whereas in changes to the submission procedures also the authors may have a legitimate say. However, if a method claims to be a true CISD method, it should at least *allow for* such development criteria distinctions to be made.

This work is not complete, of course. There are many more user-centered methods than the ones mentioned here. Furthermore, in this small space we can only address the most basic aspects of user-centered methods and community dimensions, cannot zoom in on particular methods, nor do a more detailed comparison of their strengths and weaknesses. Other design criteria, such as community size and participants' backgrounds also may have to be included in successful development methods for community IS. However, we feel that the proposed criteria are at least *necessary,* if not sufficient. We are confident that by further refining and applying our framework, and using it to develop robust specification tools, we can make a contribution to grounding the much-needed research into advanced CISD methods in existing, proven methodological work. This could be another step to help develop more thriving virtual professional communities.

References

1. de Moor, A. and M.A. Jeusfeld, *Making Workflow Change Acceptable.* Requirements Engineering, 2001. **6**(2): p. 75-96.
2. de Moor, A., *Language/Action meets organisational semiotics: situating conversations with norms.* Information Systems Frontiers, in press.
3. Peterson, R.R., M. Smits, and R. Spanjers. *Exploring IT-enabled network organizations in healthcare: emerging practices and phases of development.* in *8th European Conference on Information Systems*. 2000. Vienna.
4. Brooks, F.P., *The mythical man-month : essays on software engineering*. Anniversary ed. 1995, Reading, Mass.: Addison-Wesley Pub. Co. xii, 322.
5. Kollock, P., *Design principles for online communities.* PC Update, 1998. **15**(5): p. 58-60.
6. Pickard, M., *Under construction: (re)defining culture and community in cyberspace*. 1998, University of Manchester.
7. Talbott, S., *The future does not compute : transcending the machines in our midst.* 1st ed. 1995, Sebastopol, CA: O'Reilly & Associates. xix, 481.
8. Preece, J., *Online communities : designing usability, supporting sociability*. 2000, Chichester ; New York: John Wiley. xxiv, 439.
9. Hirschheim, R., *User participation in practice: Experiences with participative systems design*, in *Participation in systems development*, K. Knight, Editor. 1989, Kogan Page.
10. Bannon, L. *From requirements as texts to requirements as constructions.* in *CAiSE96: Requirements Engineering in a Changing World Workshop, May 20-21*. 1996. Crete.
11. Boedker, S., J. Greenbaum, and M. Kyng, *Setting the stage for design as action*, in *Design at work: Cooperative design of computer systems*, J. Greenbaum and M. Kyng, Editors. 1991, Lawrence Erlbaum Ass.: Hillsdale, NJ.
12. Kyng, M., *Designing for cooperation: cooperating in design.* Communications of the ACM, 1991. **34**(12): p. 65-73.
13. Jackson, R.B. and D.W. Embley, *Using joint application design to develop readable formal specifications.* Information and Software Technology, 1996. **38**(10): p. 615-631.
14. Holtzblatt, K. and H. Beyer, *Making customer-centered design work for teams.* Communications of the ACM, 1993. **36**(10): p. 92-103.
15. Koh, I. and M. Heng, *Users and designers as partners: design method and tools for user participation and designer accountability within the design process.* Information Systems Journal, 1996. **6**: p. 283-300.
16. Salchenberger, L., *Structured development techniques for user-developed systems.* Information & Management, 1993. **24**(1): p. 41-50.
17. Whitworth, B. and A. de Moor. *Legitimate by design: towards trusted virtual community environments.* in *35th Hawaii International Conference on System Sciences, January 7-10*. 2002. Hawaii: IEEE Computer Society Press.
18. Hirschheim, R. and H. Klein, *Realizing emancipatory principles in information systems development: The case for ETHICS.* MISQ, 1994. **18**(1): p. 83-109.
19. Hirschheim, R., H. Klein, and K. Lyytinen, *Information Systems Development and Data Modeling: Conceptual and Philosophical Foundations*. 1995: Cambridge University Press.
20. de Moor, A., *Empowering Communities: a Method for the Legitimate User-Driven Specification of Network Information Systems*. 1999, Tilburg University: Tilburg, The Netherlands.
21. Holt, P., *User-centred design and writing tools: Designing with writers, not for writers.* Intelligent Tutoring Media, 1992. **3**(2/3): p. 53-63.
22. Malone, T.W., K.-Y. Lai, and C. Fry, *Experiments with Oval: a radically tailorable tool for cooperative work.* ACM Transactions on Information Systems, 1995. **13**(2): p. 177-205.
23. Kensing, F. and T. Winograd, *The Language/Action approach to design of computer-support for cooperative work: A preliminary study in work mapping*, in *Collaborative work, social communications and information systems*, R.K. Stamper, et al., Editors. 1991, IFIP. p. 311-331.