Supporting the evolution of workflow patterns for virtual communities

Hans Weigand, Aldo de Moor and Willem-Jan van den Heuvel
Infolab, Tilburg University
PO Box 90153
Tilburg, The Netherlands
email: H.Weigand@kub.nl, ademoor@kub.nl, wjheuvel@kub.nl

Abstract

Virtual communities that make use of network information systems (NIS) have a need for specification support that agrees with their communal character. System specification changes must be acceptable to all members for a community to thrive. We concentrate on the specification of workflow-enabling communication tools rather than information tools that support a single user. In a previous paper, we have defined various levels of workflow specification, from speech acts to scenarios. Once patterns for these levels have been identified, they can be stored in a component library and be (re)-used effectively by communities to speed up their NIS development. To ensure the acceptability of changes in workflow patterns, we propose to apply an existing method for legitimate user-driven specification.

1 Introduction

The internet is embracing more and more social functions, such as commerce (Electronic Commerce), public government and various kinds of community interactions. Some communities owe their existence on the presence of Internet, and can truly be called virtual communities, while in many other communities the Internet is becoming at least an important medium of communication. In this article, no distinction is made between these two groups, since in both instances there is a strong mutual influence between community and IT.

To characterize a virtual community, we first note that it consists of a group of people *bound* together [19]. These bonds can be of many kinds, but include at least some shared interests and norms governing the behaviour of community members. Second, we consider the role of the enabling (Internet) technologies to be more than just passively capturing and representing information. They should also help to unify the community by actively engaging users in defining and integrating the information resources of the community [20]. Third, the members of the community should not be forced to use the technologies. Instead, their participation in the design and development of the required information systems should take place in a democratic dialogue in which their requirements are gradually formalized [3].

The structure of this paper is as follows. Section 2 contains a case description of typical community growing virtual, namely, a dog owner club. In section 3, the notion of community is deepened by associating it to the theory of communicative action introduced by Habermas. This theory is then applied to describe the way in which the norms governing the - operational and specification - behavior in communities evolve. After presenting an overview of a layered component library for workflow processes in section 4, we describe in section 5 how the use of such a library, which amounts to changing workflow patterns, can be supported in such a way that the results are acceptable to the community.

2 Case description: the Dutch sheltie club

The Dutch sheltie club is an association founded in 1934 of private persons interested in shelties, the Shetland sheep dog. It is a national, non-profit club of sheltie fanciers devoted to the betterment of the Shetland sheepdog breed in the Netherlands.

Activities of the club include organizing shows and championships, meeting-days and other activities in the interest of their members. An important role is their mediation of pup breeding and pup sales. People, either a member or not, interested in buying a puppy, can contact the association and receive a list of certified breeders that have puppies on sale. Breeders are certified by the association. When subscribing to the association, a breeder also signs an agreement that specifies certain responsibilities. For example, puppies should not be sold under the age of 8 weeks without proper immunization and health protection. Buyers are encouraged to fill in an evaluation form afterwards and

send it to the association. In this way, malbehavior can be detected. Buyers also have certain responsibilities. Buyers, dog owners that do no adhere to these responsibilities can be refused membership.

The ethical guidelines and procedures advocated by the club are discussed and decided by the member community itself. The guidelines are thus not imposed by a small subgroup but are supported by the whole community.

The Dutch sheltie club does not have a Web-site yet, as for example the American Shetland Sheepdog Association does have (http://www.assa.org/), but it already uses email. We expect that the current non-digital procedures such as for membership application and sales mediation will be (partially) replaced by electronic ones in the near future. However, it does not make a big difference: it is not the communication media that create the community, but the shared interests and the norms that are adhered to. Therefore, any replacement of manual procedures by means of electronic ones should take the community character into account.

The sheltie community shows some characteristics of communities that are worth noting. The first item is that the community has a ethical code. More generally, some norms of behaviour are defined and laid down in documents. These norms do refer to actions of the members at home, but also to what we could call workflow procedures, such as for selling a puppy or managing a championship.

A second remark is that the documents formalizing (some of) the norms of this community are created and updated in a democratic process. It is important that all members can raise new points to be considered and that the discussion and decision about these norms is made in a transparant way that is acceptable to the community. In this respect, the sheltie club is really a community rather than a hierarchical organization.

To identify the characteristics of such democratic discussion, we turn to the work of Habermas.

3 Communicative action and practical discourse

The German philosopher Jurgen Habermas ([9] [13] [6] [25]) is well-known for his critical theory of society which over the years he has based on a theory of communicative action. In this section, we start by introducing Habermas' theory of rationalization, and then discuss his ideas on practical discourse and their relevance for communities.

3.1 Rationalization processes

Rationalization is a key concept in the philosophical work of Habermas. Rationalization refers here in the first place to a particular development in Western society in which the reasons for actions are no longer primarily implicitly determined by traditions, but have to be listed more explicitly. What Habermas (in the line of Weber) means is that modern culture has made available a "rationalized lifeworld" - one in which actors consistently carry the expectation that the validity claims raised in speech are opened for discussion and cognitively distinguished [25]. As such a rationalized lifeworld emerges, an increasing number of spheres of social interaction are removed from the guidance by unquestioned tradition and opened to coordination through consciously achieved agreement. In other words, in the lifeworld we can notice an increasing reliance on *communicative action*, also called "action toward understanding" (Verstandigung). Communicative action achieves coordination by means of shared knowledge and norms that are not imposed but voluntarily accepted by the participants in an open discussion. According to Habermas, rationalization also means that *different* validity claims are distinguished. This means that every communicative action simultaneously raises a claim to truth, a claim to normative rightness, and a claim to truthfulness. These claims refer to three different worlds (the object world, the social world, and the subject world, respectively), and hence should not be mixed up, as they often are in premodern societies. Nor should they be reduced to one, as in modern positivist thinking, where only the claim to truth is recognized. Such a reduction means in effect that everything is considered an object, including the human subject and normative grounds.

However, there is also a second process of rationalization that has been described by Max Weber as well, but which Habermas distinguishes sharply from the first process. Simultaneously with the advance in communicative rationalization, there also occurs an advance in the rationality of the society as measured from a functionalist or systems perspective. This means that there is an expansion of social subsystems that coordinate action through other means, namely, through the media of money (the market) and administrative power (the bureaucracy, or the centralized state). This rationalization process is ambivalent. It is beneficial to the extent that it releases the (growing) pressure on communicative action. Communicative action is rational, but also costly; it typically takes a lot of time to reach agreement in a group. The other coordination mechanisms are much more efficient. But the problem that Habermas notices is that these other coordination mechanisms increasingly invade *all* the areas of social life. This is called the "colonization of the lifeworld" that brings in its wake a growing sense of meaninglessness and dwindling freedom.

To a large extent, our lifeworld is formed by the communities we live and work in. In the line of Habermas, we suggest to use communicative action as the basic coordination mechanism in communities. In this way, we hope to promote rationalization in the first sense while avoiding the

undesirable effects of purely functionalist rationalization.

3.2 Rules of discourse

Communicative action, also described as "negotiation of new situation definitions" is the process through which the social validity of knowledge is reproduced. Several types of discourse can be distinguished, but of particular relevance to the functioning of communities is the discourse about norms. A socially valid norm, that is, a norm that is recognized by a community, cannot claim to be right simply on the grounds that it is in fact recognized. It must be made clear that behind every valid norm stands a good reason, and this requires a community to have a constant, rational, and dependable method to validate moral norms reflectively. Practical discourse is this formal, universal and ideal form of communication. In the following, we will take the discussion of practical discourse in [1] as our starting-point.

In normal day-to-day communication, also that within communities, speakers can appeal in their validity claims to common norms. It is when the norm itself is challenged that a break-down occurs and practical discourse starts in order to reestablish a background consensus. Given that common understanding is the end of discourse, certain conditions of discourse suggest themselves. These conditions are intended to ensure that the resulting understanding is indeed genuinely common and that the agreed-upon norms are considered valid by all. In his most formal account of these conditions [10], Habermas proposes three levels of argumentation.

The first set of rules require that we speak the same natural language according to the same general conventions. In short, the discourse must be *meaningful* to all participants.

The second set of rules is drawn from the premise that participants desire to reach agreement and has to do with sincerity and *responsibility*. The participants are expected to be honest in their claims and to respect the intent behind claims of the other participants.

The third set of rules formalizes the process of communication itself, and aims to ensure that only the "force of the better argument" prevails. No one with the competency to speak and act may be excluded from discourse. Everyone is allowed to question or introduce any assertion as well as to express his attitudes, desires and needs. And no one may be prevented, by internal or external coercion, from exercising these rights.

Although each participant enters the discourse with his or her personal interests and needs, it is characteristic of practical discourse that we search for *generalizable interests*. Many of our needs and interests are not generalizable, but some of them are, and practical discourse asks participants to search for such points of commonality to serve as foundations for legitimate norms.

These rules of discourse can be taken as a basis for discussion and decision making in communities. In section 5, we will present the RENISYS specification method in which these rules are applied to support the negotiation about workflow definition changes.

Before showing how such a discourse-based approach can be applied, we first describe how workflow definitions can be structured using a hierarchy of patterns.

4 Workflow Patterns

In a previous paper [24], we have applied the notion of patterns to the analysis of (electronic commerce) communication and conversation policies. Our patterns are partly based on linguistic theories, like speech act theory (Austin, Searle) and Habermas' theory of communicative action, as well as principles of information system design.

4.1 Workflow levels

In order to enhance maximal reusability, we distinguish five abstraction levels of (communicational) analysis patterns (see Fig. 1) from low-level speech acts to high-level scenarios. *Transactions* are units composed of speech acts, for example, a request/commit. Transactions can be grouped in *workflow loops*. A contract or *interaction* represents a reciprocal relationship and typically consists of two workflow loops. Finally, a set of related interactions is called a *scenario*, an instance of a use case, which typically denotes a complete business process.

4.1.1 Speech Acts

Representation languages such as the Formal Language for Business Communication (FLBC - [12]) and methods based on the Language/Action Perspective ([26]) assume that the speech act is the most elementary unit within the communication between subjects.

According to Searle [17], speech acts are constituted of three parts: the propositional contents, the illocutionary point and the illocutionary force. He distinguishes between five different illocutionary points: assertives, directives, commissives, expressives and declaratives. This taxonomy defines what the speaker can do on the basis of an utterance, with a propositional content.

FLBC-II uses only the assertions and directives, leaving out commissives, expressives and declarations. However, they can be added when needed, since the language is not closed. Commissives are used to commit speakers to a future course of action. The expressive point expresses the subjective attitude of the speaker towards the state of affairs. Declarations are used to change the state of the world

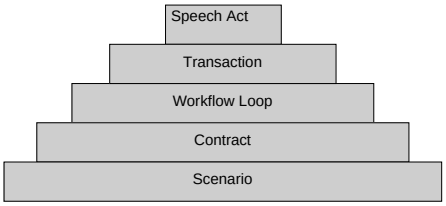

Figure 1. Levels of Meta-Analysis Patterns

according to the proposition uttered. An example of a message type definition is the following:

```
MsgType accept_request_provide_puppy
  (sender($breeder), receiver($member),
  product($provide_puppy), date($date) ==
  (person($member), person($puppy), accept,
  request_provide_puppy($provide_puppy,
  $date))
```

4.1.2 Transaction

Typically, speech acts go in pairs, for example, a request followed by a commit. This reflects the fact that communicative action is a joint activity (cf. section. 3 above). For example, the request itself does not create an obligation as long as the Addressee has not agreed with the validity of the request.

We define a transaction as the smallest possible sequence of actions (speech acts) that has an effect in the social world of the participants, in other words an obligation, an authorization or an accomplishment [22]. Deontic logic is the modal logic theory that deals with notions of obligations and permissions and that has been applied in law as well as in computer science. Deontic consequences of (a sequence) of speech acts play an important role during the representation of the electronic commerce transaction, because they define the mutual rights and duties of the two parties, i.e. the *implications* of a message. However, the transaction type definition (see below for an example) describes only the messages that the transaction contains and their relative ordering, while the deontic effects are described at the higher workflow level.

```
TransType request_provide_puppy
  (speaker($member), addressee($breeder),
  product($puppy), date($date) ==
  ([person($member), person($breeder)],
  request_provide_puppy($member, $breeder,
  $date, msg1),
  accept_request_provide_puppy($breeder,
  $member, $date, msg2),
  [before (msg1,msg2)] )
```

Two very general patterns of transactions are the factagenic and the actagenic conversation [4], each constituted of at least two speech acts. The former establishes a mutually agreed fact and the latter a mutually agreed obligation to perform some action.

4.1.3 Workflow

The next level that we distinguish is called ”workflow” in accordance with the use of this term in the Action Workflow approach of [14]. The workflow can follow the model of the basic conversation of action, as defined by Winograd and Flores. It is assumed in the Business Process Modelling approaches based on the Language/Action Perspective (DEMO, Action Workflow) that the business processes are composed of workflow loops. The basic principles underlying this approach are:

- Actions are performed by subjects and *for* subjects. An action specification is not complete without the beneficiary role;
- Actions do have an effect in the object world, but to count as fact in the social world, the action must be reported and accepted. So the action specification is not complete without an evaluative communication afterwards;
- Both the request for action and the acceptance of a fact require a give-and-take, the active involvement of both parties.

The workflow loop ([2]) starts with a proposal, a request from the customer (or initiator) or an offer from the performer (or executor). In the second phase, the customer and the performer come to an agreement. After the executor has executed the promised action, he states/declares that (s)he

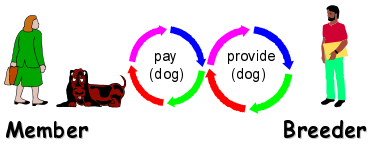

Figure 2. Interaction Level

is finished to the initiator. In the last phase, the satisfaction phase, the initiator can declare to the performer that the transaction was (un)successful.

We have found that this pattern is very general, but it is not always the case that the evaluator is identical with the initiator. The loop idea is based in essence on the agency relationship. It may be that communities work with other kinds of relationships as well.

The following example describes a workflow loop pattern for selling a puppy.

```
WflType sell_puppy_to_member
  (initiator($member), executor($breeder),
  product($provide_puppy), date($date)) ==
  ([person($member), person($breeder)],
  /* Obligation of the breeder to sell a
  /* puppy after the request of the member
  S1: OBL($breeder, provide_puppy)
  in request_provide_puppy($member,
    $breeder, $price, $date)
  goal provide_puppy ($breeder,
    $member, $price)
  exit cancel (request_provide_puppy) -->
    WflType Cancel_Request
```

4.1.4 Interaction

The transaction models that we have just discussed give a rather biased perspective on the transaction. The analyst must either choose the viewpoint of the initiator or that of the executor of the transaction (in our case, the member or the breeder). We follow Goldkuhl who claims that a business transaction must be interpreted as being an 'interchange process between a supplier and a customer' and that it 'involves the creation and sustainment of business relations' [8].

An interaction involves at least two parties, but in practice may involve several (trusted) third parties. In commerce, the most obvious ones are the bank (for the money transfer) and the transporter (for the product transfer), whereas in the sheltie case, we can think of the club itself or the vet. In the simplest case, the interaction is a parallel execution of two workflows; the synchronization can be described by means of temporal constraints. In Fig. 2. the interaction is modelled as composed of two loops, each consisting of two transactions.

```
InteractionType Member/Breeder
  (customer($member),
  supplier($breeder),product
    ($provide_puppy), date($date)) ==
  ([person($member), person($breeder)],
  request_provide_puppy($member,
    $breeder, $provide_puppy, $date),
  [person($breeder), person($member)],
  request_provide_payment($breeder,
    $member, $transfer_puppy,
  $date, request_provide_puppy.request_
    provide_puppy
  BEFORE request_provide_payment.
    Request_provide_payment)
```

4.1.5 Scenario

The scenario is the highest level of communication pattern that we distinguish. The scenario consists of a coherent collection of interactions, workflow loops and transactions. Whereas the previous levels focus on the communication between two agents (possibly with the aid of mediating parties), the scenario shows how these parts are interconnected.

An example scenario for the sheltie community is given in Fig. 3. It is not intended to be complete, but contains some essential processes. In the first place, it describes the contracts between members, either normal or breeder, and the Sheltie Club. The payment part in both contracts is straight-forward; the subscribe part is a workflow that consists of an application (request to subscribe), a formal evaluation procedure by the Sheltie Club board, and notification. The formal evaluation includes an action from the board to publish the aspiring members in the club news magazine.

We have found it necessary to extend our earlier scenario model to include document references (represented as dotted lines in Fig. 3). The Sheltie Club has defined certain rules of conduct. We have modelled them here as a contract between the Club (represented by the board) and the Community, because it describes obligations for both. When a new member subscribes, it not only means that he or she is registered, but also that he or she subscribes to the rules of conduct. This relationship between the membership contract and the community contract is modelled in the scenario, but we have not worked out the logical-formal consequences of such a reference yet.

```
ScenarioType shipment(customer($member),
  supplier($breeder),
  community($sheltie_club),
  product($product), date($date) ==
  ([person($member_admin), person($breeder)],
  [identification($sheltie_club,$breeder)])
  ..
  ([person($sheltie_club), person($breeder)],
  [ma/br($sheltie_club, $breeder)],
  ..
```

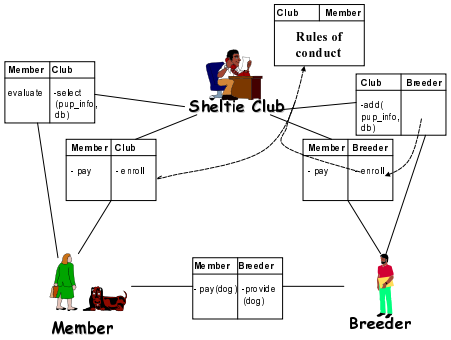

Figure 3. Scenario Level

```
  ([person($sheltie_club), person($member)],
  [termination_relation($sheltie_club,
    $breeder)]
  ..
  Club/member refers_to Rules_of_Conduct
)
```

As has been argued in [11], an important (and first in time) part of the scenario is the *identification* of the communicating actors. Identification in Cyberspace typically requires a Domain Administrator who provides identities to new members and can be asked to check the identity of an agent. The identification process comes back in the sheltie case in the form of the enrollment procedure.

4.2 Conversations and texts

Language/Action-based methods focus on conversation patterns, such as the basic conversation for action pattern defined in [26]. However, as Taylor has argued [7], organizations do not only have conversations but also texts. Texts are persistent representations which are in principle accessible to many subjects for reading and updating. They do not need to exist in paper of course and can be unstructured or structured; a database system is also a text. The member register maintained by the Domain Administrator is an example of a text in the sheltie club.

Normally, conversations draw upon a shared context, and mutually accessible texts are very useful for providing part of this context. When members, or breeders, enroll in the club, they are asked to subscribe to the rules of conduct. This is also an example of a text. The rules of conduct text represents a contract between the club and its members. Since it is a text, it is possible for the community to discuss and amend it in a conversation. As we have seen in the previous section, it is also possible to make references to a text from within another text or conversation. Assuch it can serve as a common ground for the communicative action.

4.3 Enabling change in communication patterns

Communication patterns need to be adapted to incorporate evolution in communication flows. The evolution can be triggered not only from the inside but also from the outside. For example, the board needs to change the Rules of Conduct on the basis of changed national regulations. We discern several types of change. In the first place, there can be changes in the relationships between patterns at the same abstraction level (i.e. between workflows or between interactions). Secondly, there can be changes in the contents of these patterns, such as the instantiation or addition of pattern parameters (e.g. under governmental pressure we may need to change the identification process of members of the community and check their identity; hence we are forced to introduce an additional parameter to the identification interaction pattern).

In short, if we want to react to change, we must be able to adapt the patterns. However, a straight-forward engineering approach is too simple since it abstracts from the legitimacy of the change processes. In the following section, we will explain how we can change patterns in a legitimate way as we store meta-data about the parameters (e.g., parameters in the workflow signature should be of *type* executor or initiator.) It also implies that the change processes themselves should be modelled as conversational processes.

5 Making Legitimate Specification Changes

In Sect. 5.1., the RENISYS system specification method is outlined. In Sect. 5.2., we show how this method can be used to ensure that pattern changes are legitimate.

5.1 The RENISYS Method

The RENISYS (**RE**search **N**etwork **I**nformation **SY**stem **S**pecification) method facilitates the legitimate user-driven specification process. It supports the handling of breakdowns in the collaborative work of virtual professional communities. The method allows individual users who have become aware of a problem with 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*.

The method then determines which other users are to be involved in the resolution of these definitions. To this purpose, the *composition norms* that regulate the acceptable specification behaviour of actors in the community play an important role. An example of such a norm would be that the board of the Sheltie Club is permitted to modify definitions of the puppy selling (workflow) process. The method calculates the resultant deontic effect of the set of composition norms that apply to a particular user and the speci-

fication process required to change the definition. In this way, it knows which users to involve in the *conversation for specification* in which the problematic knowledge definition can be legitimately changed. Additionally, or alternatively, a *discourse process* can be started in which users can critically examine background assumptions that determine the meaning of the various knowledge definitions making up the system specifications.

In the method, knowledge definitions are represented and reasoned about using conceptual graph theory [18]. One of the useful properties of this theory is that it creates implicit generalization hierarchies of graphs. This has the great advantage that properties of different sets of definitions can be concisely represented. Another advantage is that conceptual graphs can be easily mapped to (pseudo)-natural language constructs, thus allowing for more effective interactions between method and users. Dynamic deontic logic [23] is used to handle composition norm conflicts and calculate the authorizations of users involved in a particular conversation for specification. To model the moves that users can make within a conversation for specification, a Specification Process Model was developed, which is a variation of Van Reijswoud's Transaction Process Model [21]. This model links the speech acts that lead to a successful transaction with the speech acts necessary for the discussion of validity claims, and with those acts required for the critical discourse of background assumptions in the sense of Habermas's theory of communicative action.

5.1.1 Knowledge Representation

In RENISYS, four types of knowledge definitions are distinguished[1]. *Composition norms* are meta-norms that determine the acceptable specification behaviour of community members. They include permitted, required, and forbidden compositions. The composition norm example mentioned above, which concerned a permitted composition, is represented as follows:

```
[Perm_Comp : [Club_Board] ← (Agnt) ← [Control] → (Obj) −
   [Modify_Type] → (Rslt) −
      [Type : [Sell_Puppy_To_Member]]].
```

In this example of a graph, we can distinguish the nodes for Club-Board, Modify-Type (an activity) and Type:Sell-Puppy-to Member. The agent link between Club-Board and Modify-Type indicates that it is the Club-Board who executes the activity. Similar for Rslt (result). The whole graph itself is labelled as a Perm_Comp, that is, a permitted composition.

[1]Knowledge definitions are in conceptual graph notation, which we assume to be familiar to the reader. The syntax of the knowledge definition categories is explained in [16], and is not repeated here.

Besides composition norms, there are *action norms* which are workflow-level norms that regulate the acceptable operational behaviour of network actors. Action norms comprise permitted, required, and forbidden actions. An example is that a buyer is required to send in an evaluation form:

```
[Req_Act : [Buyer] ← (Agnt) ← [Exec] −
   (Obj) → [Send_Evaluation]].
```

Type definitions define the meaning of network concepts. For example, a workflow process can be defined as a transformation of some input object into an output object.

```
[Type : [Workflow : *x] → (Def) → [Transformation : *x] −
   (Matr) → [Object]
   (Rslt) → [Object]].
```

Finally, *state definitions* indicate properties of concrete entities in the universe of discourse. Such a definition could state that Jeroen is the chair of the club board.

```
[State : [Chair : #Jeroen] → (Poss) −
   [Club_Board : #Sheltie]].
```

This is a very brief introduction of the RENISYS method. Space is lacking to describe it in more depth. Further details are introduced in the next section where necessary.

5.2 Making Acceptable Workflow Pattern Changes

In Sect.4, the patterns were presented that can be (re)used in the specification of network information systems. However, typical of virtual professional communities is that they are prone to extensive change. To ensure that pattern changes are not only meaningful but also acceptable to the community as a whole, we now show how RENISYS can be used to ensure that only legitimate such changes can be made.

As an example, we describe how the sell_puppy_to_member-workflow (See Sect.4.1.3) could have been legitimately defined out of an existing, more generic workflow pattern. In the example, we do not include all attributes distinguished in the pattern, to conserve space. Similar illustrations could be given for the other patterns.

The (simplified) workflow pattern to be created is:

WflType sell_puppy_to_member (**initiator**($member), **executor**($breeder),
product ($provide_puppy)) ==
([**person**($member),**person**($breeder)],
S1: OBL($breeder, provide_puppy))

This pattern says that selling a puppy to a member is a workflow that can be initiated by a member, is mandatorily executed by a breeder, and results in a provided puppy.

This workflow is a *specialization* of the generic workflow pattern. However, the workflow pattern change process in which this specialization is produced cannot be directly mapped to a single RENISYS *specification process*. This because a workflow pattern is a complex construct, whereas RENISYS only distinguished a limited number of primitive knowledge categories. In terms of the specification method, the workflow pattern consists of several type and action norm components. Therefore, any workflow pattern specialization needs to be decomposed into (for instance) the following three RENISYS specification processes:

- A creation of a workflow type definition (selling a puppy is a workflow that results in a provided puppy).
- A creation of an action norm indicating who is the initiator of the workflow (a member is permitted to initiate the sell puppy workflow).
- A creation of an action norm indicating who is the executor of the workflow (a breeder is required to execute the sell puppy workflow).

Assume the following (partial) RENISYS type hierarchy has already been defined[2]

Entity >
  Actor >
    Person >
      Board_Member
      Club_Member
      Breeder
  Object >
    Provided_Puppy
  Process
    Control
      Init
      Exec
      Eval

Formally, the mapping between a workflow pattern change and the RENISYS specification processes can now be represented as follows:

ΔPattern$_{(\text{Workflow, new})}$:=
  [Create_Type : [Type : [Workflow]]] +
  [Create_AN : [Perm_Act : [Actor] ← (Agnt) ← [Init] −
    (Obj) → [Workflow]]] +
  [Create_AN : [Req_Act : [Actor] ← (Agnt) ← [Exec] −
    (Obj) → [Workflow]]]

[2]A detailed description of the core process ontology underlying the RENISYS type hierarchy is given in [15].

Thus, to create a new workflow (pattern), first, the properties of the new workflow type need to be defined, such as what are its input and output objects (type creation). Second, it must be determined who is permitted to initiate the workflow (action norm creation), and, third, who is required to make or execute the actual definition (action norm creation).

Each RENISYS specification process (i.e. type creation, action norm creation) is considered to consist of three *compositions*: the *initiation*, *execution*, and *evaluation* of the knowledge definition change process that is the objective of the specification process. For all three compositions of each of the specification processes that are determined by the workflow pattern specialization, RENISYS calculates, for all users in the network, who is permitted or required to participate in the composition.

To illustrate, let us take the first specification process, which concerns the creation of a new workflow type, as the *active specification process*. To calculate which users to invite in each of the three *active compositions*, RENISYS uses two functions, of which the semantics have been described in [16].

Say that we want to know if Mary (who is both a club member and a board member) is permitted to execute the creation of a new workflow type (which we must know to determine if she can legitimately make a sell puppy workflow type definition)

The function $\mathbf{D}_{\mathbf{CN\_APPL}}$(*user,comp*) calculates which composition norms apply to user *user* for active composition *comp*.

Applied to the example, this function could provide the following results:

$\mathbf{D}_{\mathbf{CN\_APPL}}$(Mary,
  [Exec] → (Obj) → [Create_Type : [Type : [Workflow]]]) =
  {[Perm_Comp : [Club_Member] ← (Agnt) ← [Exec] → (Obj) −
    [Create_Type] → (Rslt) → [Type : [Workflow]]],
  [Req_Comp : [Board_Member] ← (Agnt) ← [Control] → (Obj) −
    [Create_Type] → (Rslt) → [Type : [Workflow]]]}

Thus, in this case, two composition norms are retrieved. The first one says that any club member is permitted to be involved in the actual definition of new workflow types. The second says that board members have a responsibility to control (i.e. initiate, execute, and evaluate) new workflow types.

Based on this set of applicable norms, the *resultant deontic effect* is calculated by the function $\mathbf{de_r}$ This function deals with norm conflicts by applying norm priorities to the norms in the set. If applied to the example, this function would return the following result:

$\mathbf{de_r}$($\mathbf{D}_{\mathbf{CN\_APPL}}$(Mary, [Exec] −
    (Obj) → [Create_Type : [Type : [Workflow]]])
  = Req

Thus, we know now that Mary is required to be involved in the (execution of the) definition of new workflows, thus also in the creation of the sell puppy workflow type.

We have given an example of the decomposition of a (macro) pattern change process into a set of (micro) specification process changes. We have shown how the RENISYS method can be applied to determine who can be legitimately involved in the compositions that make up these specification processes. The decomposition of a workflow pattern creation into one type creation and two action norm creations is only one way in which this process can be decomposed. In future research, we plan to develop more elaborate mappings from pattern changes to RENISYS specification processes, for all types of patterns in the library.

6 Conclusion

In this paper, we have argued that when communities use network information systems, the way such a system is be designed should be in accordance with the unique character of communities. We have taken Habermas' theory of communicative action as *a way of thinking* about communication processes in communities. This theory argues, among others, that coordination is achieved through the common ground of accepted norms, but these norms evolve over time and rational discussion about the norm changes should be supported. We have proposed *a way of modelling* communication processes based on the use of patterns, and have described *a way of working* that takes advantage of the pattern approach and is contextualized in the community. The way of working is derived from the more general theory of legitimate user-driven specification developed in [16].

We have used the case of the Dutch Sheltie Association as an example. Although the communication processes are rather simple, we have the feeling that the case is illustrative of many existing communities. The primary concern for such communities is not the complexity of the specification, but how to ensure the validity.

In this paper, we have focused on the communication processes within a community. One way of extending the model is to take interactions between communities into account as well. We also have focused on the specification of norms; one topic for future research is the specification of *goals* and how subgoals, tasks and norms relate to these goals. Goals are important since a community is often defined by its shared goals or interests.

The final remark concerns the commercial aspects. Although the association itself is non-profit, it plays a mediating role in commercial activities, such as selling puppies. In this respect, it is an illustrative example of the role communities, virtual or not, can play in Electronic Commerce.

References

[1] S. Chambers *Reasoanable Democracy - Jurgen Habermas and the politics of discourse* Cornell University Press, 1996.

[2] P.J. Denning, R. Medina-Mora Completing the loops *Interfaces* Vol. 25, No. 3, 1995, pp.42-57.

[3] P. Day, The Human-Centred Information Society: A Community-Based Approach, *AI & Society, 1996, Vol.10*, pp.181-198

[4] J.L.G. Dietz. Business modelling for business redesign. In *Proc. HICSS '94*, pages pp. 723–732. IEEE Press, 1994.

[5] H. Clark *Using Language* Cambridge Univ Press, 1996.

[6] M. Cooke *Language and Reason - A study of Habermas' Pragmatics* MIT Press, 1994.

[7] N. Giroux J.R. Taylor, F.Cooren and D.Robichaud. The communicational basis of organization: Between the conversation and the text. *Communication Theory*, 6:1 – 39, 1996.

[8] G. Goldkuhl. Generic business frameworks and action modelling. In F. Dignum, J. Dietz, E. Verharen, and H. Weigand, editors, *Workshop on Communication Modelling - The Language/Action Perspective*, 1996.

[9] J. Habermas *Theorie des kommunikativen Handelns.* Suhrkamp, 1981 (2 volumes)

[10] J. Habermas *Moral consciousness and communicative action* Translated by Ch. Lenhardt and Sh.W. Nicholsen, MIT Press, 1990.

[11] W-J. van den Heuvel and H.Weigand. Ensuring the validity of electronic commerce communication. In Jan Dietz Frank Dignum, editor, *Communication Modelling - The Language/Action Perspective*, Computer Science Reports. Technical Univ of Eindhoven, 1997.

[12] S.O. Kimbrough and S.A. Moore. On automated message processing in electronic commerce and work support systems: Speech act. *ACM Transactions on Information Systems (TOIS)*, 1997.

[13] Th. McCarthy *The critical theory of Jurgen Habermas* Cambridge Mass, 1978.

[14] R. Medina-Mora, T. Winograd, R. Flores and F. Flores. The Action Workflow approach to workflow management technology. in *Proceedings CSCW 1992*, pp. 281-288, 1992.

[15] A. De Moor, and H. Weigand An Ontological Framework for User-Driven System Specification *Proceedings of the 32nd Hawaii International Conference on Systems Science*, Maui, January 5-8, 1999.

[16] A. De Moor *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*, Tilburg University, Ph.D. Thesis, October 1999.

[17] J.R. Searle. *An essay in the philosophy of language*. Cambridge University Press, 1969.

[18] J. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, 1984.

[19] Talbott. *The Future Does Not Compute: Transcending the Machines in Our Midst*, 1995, O'Reilly & Ass.,Inc.

[20] S.R. Tilley. Documenting Virtual Communities, *Proc. of the 14th American International Conference on Systems Documentation (SIGDOC'96)*, Research Triangle Park, North Carolina, October 20-23, 1996, pp.43-49.

[21] V. Van Reijswoud. *The Structure of Business Communication: Theory, Method, and Application*, Delft University, Ph.D. Thesis, 1996.

[22] H. Weigand, E. Verharen, and F. Dignum. Dynamic business models as a basis for interoperable transaction design. *Information Systems*, 22(2/3):139–154, 1997.

[23] H. Weigand, Verharen, E., and Dignum, F., Integrated Semantics for Information and Communication Systems. In Meersman, R. and Mark, L. (eds), *Database Application Semantics*, Chapman & Hall, 1997.

[24] H. Weigand, W.J. van den Heuvel Meta-patterns for Electronic Commerce based on FLBC *Proc. HICSS'98*,IEEE Press, 1998

[25] S.K. White (Ed) *The Cambridge Companion to Haberma* Cambridge Univ Press, 1995.

[26] T. Winograd and F. Flores. *Understanding Computers and Cognition*. Addison-Wesley Publishing Company, 1986.