Web Service Selection in Virtual Communities

Aldo de Moor, Willem-Jan van den Heuvel
Infolab, Department of Information Systems and Management
Tilburg University, The Netherlands
ademoor/wjheuvel@uvt.nl

Abstract

Virtual communities increasingly make use of standard Internet-enabled web services to support their collaborative activities. Such web services need to offer the right amount of functionality to meet community requirements. However, both requirements and enabling services are continuously in flux. A critical challenge therefore is that the community can efficiently ensure that web service changes are both technically feasible and socially acceptable.

In this paper, we outline a selection approach for virtual communities that takes into account both the feasibility and the acceptability of web services. To this purpose, we adopt a semiotic view on the selection process, showing that for the adequate selection of web services three subprocesses are required: (1) syntactic discovery, (2) semantic matching, and (3) pragmatic interpretation. We then present a meta-model of web service selection support that is grounded in this view. This model can be used to detect gaps in web service selection support. This knowledge is essential for the construction of better selection support methodologies. We apply the meta-model to analyze a case on a courseware development community.

1. Introduction

With the rise of the Internet, virtual communities are gaining importance as a new business model for virtual collaboration, as demonstrated by the proliferation of trading and education communities. In an increasingly networked society, with ever more need for global, and flexible ways of professional interactions, virtual communities are natural candidates to fill collaborative gaps in traditional, hierarchical organizations. With the advent of more user-friendly and powerful web applications, business is also discovering the power of virtual communities. For example, virtual communities of consumption are affiliative groups whose online interactions are based upon shared enthusiasm for, and knowledge of, a specific consumption activity or related group of activities [1], e.g., the James Bond Yahoo Community[1]. Such communities allow consumers to critically evaluate products and companies to gather valuable data on product characteristics from loyal customers.

What is a virtual community? Communities are not just aggregates of people, temporarily interacting. A community has been defined as a group of people who share social interactions, social ties, and a common 'space' [1]; as a social network of relationships that provide sociability support, information, and a sense of belonging [2], and as a set of relationships where people interact socially for mutual benefit [3]. The key seems to be strong and lasting interactions that bind community members and that take place in some form of common space. A virtual community differs from other communities only in that its common space is cyberspace. Virtual communities therefore describe the union between individuals or organizations who share common values and interests using electronic media to communicate within a shared semantic space on a regular basis [4].

In virtual communities, the common space is provided by a suite of collaborative and communicative functionalities, ranging from simple mailers to advanced web applications [5,6]. This functionality mostly consists of standard tools or components, so that information systems development becomes more a process of functionality selection than building whole new systems from scratch [7]. This standard functionality increasingly comes in the form of web services. Simply stated, web services constitute reusable and reasonably fine-grained software components, which can be invoked through the Internet. A characteristic of web services that is of particular interest is that they are self-describing,

[1] http://groups.yahoo.com/group/theclubofjamesbond/

which means that they contain metadata that advertise their functionality. Web services are everywhere on the Internet. Prominent examples like the Microsoft MSN.NET web services of Mappoint, password service, chat service, and its NetMeeting video-conferencing service, the Yahoo Chat service, and, ADG's discussion web services are only the very tip of the web services iceberg.

Web services are particularly interesting for virtual communities, as they allow non-technical community members to combine them in new value-adding applications. For example, a chat web service of supplier X, and a document sharing web service from supplier Y could be aggregated in a new, "higher-order" web service for cooperative report writing. Other examples of often used web services in communities include file management systems and discussion boards. Many web services provide partially overlapping functionality, offering many shared and some unique functionalities. Selecting from the plethora of web services, given the complex nature and rapid evolution of the information needs and available technologies of a typical virtual community, is therefore a daunting process. Given that this change process is so costly, many essential changes to the socio-technical system of a virtual community often do not happen, therefore inhibiting natural community evolution.

Finding ways to catalyze the selection process is thus very important for virtual communities to remain viable. The selection of which web services to use is not trivial, as there is an incredible variety of design choices in communities of practice, and their requirements even vary within particular stages of their lifecycle [8]. To address these issues, a new paradigm is emerging: that of community-centred development [5]. Two key objectives of community-centered development are that sociability and usability are achieved: sociability entails that social policies are developed that are in line with the community's purpose and understandable and acceptable to its members; usability should ensure that the implementing technologies support rapid learning, high skill retention, and low error rates. To achieve these objectives, the *development process* of community information systems should be participatory and evolutionary. This means that community members are to play an active role in the whole process from requirements elicitation to implementation. Furthermore, it should be a continuous process of refinement and extension, instead of a one-time waterfall-type development project. Methodological support that takes into account these complex community constraints is required for software design in the form of selection and tailoring of functionality components [5].

In this paper, we investigate in-depth the web services selection process. Currently, most service selection takes place in an ad hoc fashion. The objective of this paper is to allow virtual communities to construct information systems out of web services that are better tailored to their specific requirements. To this purpose, in Sect. 2, we first chart the role of virtual communities in an application domain increasingly dominated by web services: courseware development. In Sect. 3, we adopt a semiotic view on the web service selection process, arguing that it should consist of a syntactic discovery process, a semantic matching process, as well as a pragmatic interpretation process of web service functionalities. Sect. 4 introduces a meta-model for web service selection support and applies it to provide an alternative scenario for the service selection support of the courseware development case of Sect. 2. We end the paper with conclusions in Sect. 5.

2. Courseware Development Communities

E-learning is going to be one of the main applications of the Internet [9]. Proper software to support the complex didactic and administrative processes is crucial for its success. Even though e-learning is still in its infancy, the market is already saturated with software packages that aim to provide full support for all electronic course needs[2]. However, such large, atomic applications do not match the unique, complex, and evolving needs of educators.

An approach that much better suits their way of courseware development are web service methodologies, in which tailored applications are constructed out of many small web-components of functionality [10]. Such modular courseware is inherently flexible and allows educators to create integrated learning environments that are tuned to particular courses and groups of students [11].

A very important factor in successful courseware development are the stakeholders to be involved and the way they interact in communities. Well-organized communities may include any subset of potential beneficiaries pooling their "collective expertise", such as courseware developers, courseware publishers, learning resource managers, lecturers, teachers, students and distance learners [11,12]. Most of these communities are at least partially virtual, as they often have a global and cross-organizational membership.

[2] E.g. http://www.edutools.info/course/compare/all.jsp

Courseware development communities to a large extent are driven by the didactic needs of educators and students. For web-based courses to be as effective as traditional courses, collaborative learning strategies need to be implemented in which relatively small classes or groups are actively mentored by their instructor [13]. Thus, educators and students need to be core members of such communities.

Educators and software developers are also mutually dependent: the educators have good insight in what subtle didactic requirements need to be satisfied, the developers know about developing, packaging, and making available the components that match those needs. In the emerging educational component market, educators and software developers must have a shared communication language to discuss component requirements [10], which can be developed by collaborating in a community.

Apart from these direct stakeholders, there are the actors who deal with the underlying business models of courseware digital rights management. On the one hand, many informal open source communities have sprung up, often existing of lecturers offering 'homegrown' components and interested in furthering higher-quality education. Advantages of such open courseware are, for instance, that it is more easily corrected by global peer assessment, more material is reused, teachers are more easily able to get peer recognition and feedback, and universities can gain better teaching reputations [14]. On the other hand, there is an increasing number of commercial course aggregators. In between, there are mixed-motives online consortia of universities. No matter how commercial the objectives, though, courseware vendors must be sensitive to social and communicative aspects of teaching for the communities to be successful [12]. Thus, university administrators and software vendors need to play a role in extended courseware development communities as well. Furthermore, web service applications often introduce serious security problems in an organization [15], requiring, for example, the approval of a system administrator.

Summarizing, many stakeholders are to be involved in courseware development communities. They have to adhere to many different and subtle norms, related to didactic, usability, rights management and security issues. Without taking into account these norms in the web service selection process, it is bound to fail. To ensure that these norms are not violated, human interpretation of proposed service selections is essential. In the next section, we will conceptualize the selection process in such a way that providing systematic support for dealing with such community issues can be more effectively implemented.

Before analyzing the selection process, we first illustrate the rationale of proper support for the web service selection process by descibing a typical courseware web service selection process in which methodological selection support is not yet present. To this purpose, Sect. 2.1 describes a case that illustrates how ad hoc courseware development currently often takes place.

2.1. Case: Web Service Selection for Making Group Assignments

One of the authors teaches a course on Quality of Information Systems. The course lasts 13 weeks. The 2002 course counted about 80 students, divided into groups of four. Each week, there was a lecture and an assignment. The lecture took place in a classroom, the logistics of the assignment were handled electronically via the Blackboard CourseInfo 4.0 system[3]. Immediately after the lecture, the assignment was made available to the students. Every week, a number of groups had to make the assignments. These groups submitted their versions of the assignment to the teaching assistant at the end of the week. Once all assignments were in, the TA made them available to all students, together with a standard answer sheet. Then, other student groups used these standard answers to review the submitted assignments. Any remaining questions were posted on the course discussion forum, to be answered by the lecturer.

How did the web service selection process take place for the class of 2002? At the moment, decisions on which courseware to use are entirely the lecturer's. The university computer centre heavily promotes the use of CourseInfo (currenly Blackboard Learning System v6). Although CourseInfo does allow for customized functionality to be added through APIs, not much on-site development of these components takes place yet. A lecturer is thus practically forced to make use of the standard web services functionality enabled by CourseInfo. These functionalities are organized in modules and include the following:

- *generic*: send e-mail, discussion board, virtual chat, student roster
- *course information management*: announcements, course information, course documents, and assignments.
- *group pages*: discussion board, virtual chat, and file transfer.

One of the main activities to be supported by the functionalities, is the making of group assignments.

[3] http://www.blackboard.com/

This activity is subdivided into four sub-activities: (1) collecting information, (2) group discussion and collaboration, (3) submitting the results, and (4) feedback from peers.

To support these activities, a mix of the various CourseInfo functionalities has been used until recently: for example, file management mainly took place using the individual and group drop boxes, group discussion was supported by the asynchronous discussion board and synchronous virtual chat modules, and for asynchronous communication amongst students the e-mail functionality of the package was often used. Contrary to previous years, however, this year a student evaluation of the usefulness of the various components for the group assignment activity was done. Student groups had to score the importance of the various subactivities, the importance of a particular module for enabling a subactivity, and the efficacy of a module in doing so. Participation by students was high, and resulted in detailed responses. Some interesting lessons were learnt from the aggregated results, as summarized by the software manager of the university computer centre, who analyzed the results:

- Especially the basic functionalities of CourseInfo (file transfer, announcements, and send e-mail) were considered important by students.
- File-transfer, however, is not implemented well, an alternative will be sought by the computer centre. The e-mail functionality provided by CourseInfo is only very basic, other mail functionality such as Outlook, Eudora, or P-Mail is better suited, but not integrated in the platform.
- The student roster and virtual chat components were not considered important at all by students. Explanations are, respectively, that there is already an electronic study guide with better functionality, and that MSN is much preferred as chat functionality.
- It would be most welcome if applications like CourseInfo and MSN would be open in that they support better integration of functionality modules, however, this is not likely to happen soon.
- Open source courseware could be a valuable addition. However, given that the university has made a large investment in Blackboard licenses and training, a major transition will not take place in the near future. Still, experimenting with specific components in order to allow a possible (partial) migration in the future is promoted.

Lessons Learnt

Several limitations preventing the current web service selection process from being optimal can be discovered in this case:

- The functionality from which currently to choose is very limited in scope and restricted. It is not modular at all in that new components from other providers cannot easily be included nor can results between components be exchanged. Also, it is not clear what other components are available, as for an individual educator with a non-technical background it is hard to know where to look.
- Once components have been selected as likely candidates, it is not easy to see how they can be investigated on their functional properties. For example, what functionality is included in a discussion board component? Currently, the easiest way to find out, is by actually installing the component and testing it. Such an approach is still feasible in a limited, restricted environment like CourseInfo. The moment that large repositories of educational components are going to be available, however, such an approach is no longer feasible.
- Although many stakeholders should have a legitimate say in which services can best be selected, currently the privilege - and burden - is left solely to the lecturer, who mostly involves the other stakeholders in a rather ad hoc fashion. Furthermore, it is very hard to decide on which components are useful to the community of users, and according to which criteria. It is clear that stakeholders must be involved in making the assessments, not just at the main software package level, but also in the choice of individual components of functionality. Complex issues, such as integration, licensing, usability, and security need to be taken into consideration by the relevant stakeholders in the community. Furthermore, many of these assessments to a large extent depend on tacit knowledge that cannot be formalized, but should instead be provided by the right human stakeholders when appropriate [16].

To ensure adequate community involvement in the selection process, in which the many stakeholders play their legitimate roles to the full, more sophisticated web service selection support should be provided. In the next section, we first model this process in greater detail, before looking at how to conceptualize selection support methods in Sect. 4.

3. The Web Service Selection Process

Web services can be loosely defined as self-describing, interoperable and reusable business components that can be published, orchestrated and invoked through the Internet, even when they reside behind a company's firewall [17]. Web services constitute both design- and runtime, platform-agnostic distributed enterprise building blocks that can be dynamically composed into higher-order assemblies that support (inter- or intra-) organizational business transactions.

Web services-based information systems development is no longer grounded in the waterfall paradigm, in which custom-built systems are analyzed, designed, and implemented from scratch, often by outside consultants, and new versions are kept to a minimum, as such an approach is very disruptive to the organization [18]. Instead, more organic and continuous systems development approaches are required to support and control the design, implementation and maintenance activities during the web service lifecycle.

3.1. Web Services-Based Information Systems Development

To clarify the role of web service selection in the overall systems development process, we show its place in the Web services-based IS Development Paradigm, widely adopted in one form or another in the field:

Web Service Selection

In the first stage, web services are generally selected from a repository system or marketplace on the basis of its interface description, basically comprising a list of provided methods, and several non-functional properties such as the geographical location of the service provider, performance, its price, and so on. Web service selection can be performed from two perspectives: bottom-up and top-down. Top-down selection of web services starts from the business processes, e.g., setting up a course, and then identifies those services whose capabilities and quality aspects conform best. The bottom-up perspective, on the other hand, starts from the available web services, and tries to select those that fit best. In practice, both selection approaches are often combined

This phase can, for instance, be supported with the Universal Description, Discovery and Integration (UDDI) standard that provides rules for building service directories and facilitates top-down querying capabilities [19].

Web Service Adaptation

Many existing approaches for developing web service-based applications assume that services can be reused as is. However, it is unlikely that one is able to identify services with a perfect match. In practice, available web services typically only partially match with all requirements, so some adaptations need to be made. Technologies for transforming XML documents, such as the Extensible Stylesheet Language Transformations (XSLT) standard, provide mechanisms to tailor generic into customized web services.

Web Service Combination

Once services have been adapted so that they comply with new requirements, they can be combined, or 'wired'. Actually, the real added value of the paradigm lies in its ability to allow loosely-coupled services to be dynamically orchestrated into new constellations, possibly offered by different organizations. For example, a travel plan service can be developed by combining several elementary services such as hotel reservation, ticket booking, car rental, sightseeing package, etc., based on their service description.

Some standards for parts of the web service selection, adaptation and combination processes have been emerging. Examples in the area of web service composition are UDDI, BPEL4WS, XLST and BTP. However, if present at all, these standards are still to a large extent error-prone, cumbersome, restricted, and inflexible, and do not interrelate, so that rigorous web services-based IS development methodologies are still far away. As a first step on the way to more systematic Web services-based IS development, we define, in Sect.4, a meta-model with which to assess selection support quality. The basis of this meta-model is a semiotic perspective on web service selection.

3.2. A Semiotic Perspective on Web Service Selection

Our focus in this paper is on web service selection, the first stage of web services-based IS development. We focus on this crucial first stage, as it provides the basic building blocks for IS construction. Web service selection that can deal with the problems highlighted in the previous section comprises three subsequent activities: discovery, matching and interpretation. First, potential service resources need to be *discovered* by quick-scanning their "fingerprint". From these, the

services that are potentially relevant need to be identified by *matching* the discovered services in some meaningful way with the service request. This implies that community stakeholders are to link web service technologies to their requirements expressed in terms of their own community ontology. Finally, the shortlist of services found in this matching process needs to be interpreted on their usefulness by relevant stakeholders of the community. Essential is that these human interpreters, unlike computers, are able to take into account all kinds of non-formalizable considerations.

To analyze and design ways to support these subprocesses, we adopt a semiotic view on the selection process. This view consists of three complementary perspectives: (1) a syntactic (structure) view, (2) a semantic (structure-based meaning) view and (3) a pragmatic (context-based meaning) view [20]. In our opinion, the first two stages can be automated, at least to a large extent. The pragmatic view, however, cannot, as tacit knowledge embodied in committed human beings can, nor should, always be completely explicitly represented. In this paper, we equate the discovery process with the syntactic view on selection, matching with the semantic view, and interpretation with the pragmatic view. Of course, in practice, discovery has semantic and pragmatic aspects as well, just like interpretation activities can benefit from syntactic and semantic support. Future refinements could make more subtle connections between selection subprocesses and semiotic perspectives. However, the main focus of each selection subprocess is on the respective semiotic views with which we associate them here. For the purpose of making quick scans of methodological support, the current mappings suffice, therefore. By adding a semiotic view to selection processes, results from the vast literature on semiotic theory and analytical techniques can be added to investigate to what extent a selection method is complete and sound. For example, in the field of pragmatics, much work has been done on criteria for proper collaborative, goal-oriented conversations such as needed in community IS specification [21]. Vice versa, by classifying a selection method by its semiotic properties, it becomes clearer which (discovery, matching, or interpretation) stage of the selection process it can best support.

Next, we give a few illustrative examples of approaches and methods supporting the various selection subprocesses. Other examples could have been chosen. Their point is, however, to indicate the kind of semiotic differences that can be observed in practice in selection support.

3.2.1. Service Discovery: Syntactic Selection

Syntactic discovery of web services is concerned with retrieving web service interfaces on the basis of surface-level syntactic interface descriptions. The syntactic description of service capabilities typically includes the service name, its input parameters types, and the result types. Syntactic discovery deals with comparing the typed artifacts in a target specification, e.g., input parameter of a service port, with those of available ("source") resources without any regard of the actual meaning of the labels (e.g. parameter names) that are used. This implies that the quality of syntactic discovery of web services is largely determined by the syntactic richness of the service representation language in which the service is described.

Example of Syntactic Selection Support

Assume that a lecturer needs functionality to support discussions with her students. Therefore, firstly the interface of the `Discussion` software component needs to be known. This interface could be specified (in WDSL) as follows:

```
Interface Component Discussion {
  void initialize_discussion{Int
discuss_id, Int teacher_id, Int
student_id, Int nr_of_allowed
connections, String Topic};
  void terminate_discussion{Int
discuss_id, Int teacher_id};

};
```

Subsequently, the available repositories can be searched for services that conform to this interface.

3.2.2. Service Matching: Semantic Selection

Syntactic discovery is based on the assumption that service suppliers all use the same dictionary. This might be true for simple vertical domains, with a limited number of participants and highly standardized vocabulary, for example, in the form of a service resource catalogue. Not so, however, in more complex and volatile domains with thousands of participants and components. For making more sophisticated matches of required and enabled component functionalities, discovery is thus not sufficient.

We claim that, in practice, syntactic discovery is particularly effective to find service repositories or collections. Once these initial resources have been discovered, however, more refined, semantic approaches are needed for finding potentially relevant components. For example, the Semantic Web

community aims to increase the semantic power of the Web so that more meaningful queries can be answered. Semantic Web technologies like XML enable the structured description of meta-information of web elements, such as services. On top of that, the Resource Description Framework (RDF) allows for the development of lightweight ontology systems to support the exchange of knowledge on the Web[4]. Using such semantic enrichments, approaches are being developed that allow for more precise service matching than possible with syntax-only methods. Lately, several initiatives have started to leverage the notion of web services by enriching their signature with semantic information, e.g., by using a dedicated ontological markup language such as DAML. This enhancement brings automatic semantic selection and composition of web services one step closer.

Example of Semantic Selection Support

One way of semantic matching is provided by the BALES methodology [22]. Within the context of BALES, a shared interpretation of a domain, hence ontology, serves as the basic armature around which service descriptions can be compared. We have selected the WordNet ontology [23] as the common ontological framework for BALES, but in principle, other ontologies could have been used here. This web-based ontology is equipped with a massive semantic web that is freely available on the market for experimentation, focuses on meanings of terms rather than forms, and incorporates a taxonomy comprising synonym sets, hyponymy/hypernymy, and so on, thus allowing for rich service meaning descriptions to be composed.

The activity of semantic matching comprises the following two tasks:

1. *Linking Terminology to WordNet*.

In the first task, the terminology used in service specifications is linked to semantically meaningful terms in WordNet. In other words, the descriptors, which are deployed in (WSDL) interfaces of services, are connected to similar terms in the common ontology (hence WordNet). In case no matching descriptors are available, new concepts need to be created and linked to the existing ontology using synsets, hypernyms and homonyms. E.g. the entity `Discussion_manager` is added to WordNet as a concept (ID900000002) which in turn is a hyponym of concept (106945718) (manager).

2. *Calculating the Semantic Distance between Specifications.*

After the construct descriptors have been linked to WordNet, a similarity measurement is calculated to express the semantic correspondence between two service specifications. This matching algorithm not only takes into account the distance of descriptors of a target and source web service specification, but also the structure of the specification by introducing weights for each pair of descriptors that is compared. The details of these calculations are complex, and are not relevant for the purpose of this paper. What is important, is that such calculations can be used to, if needed in very elaborate ways, precisely select potentially relevant components from a potentially huge repository, using sets of formal criteria that may differ for each community.

3.2.3. Service Interpretation: Pragmatic Selection

Despite its obvious importance, the Semantic Web still has major problems in a community context, as specialized communities of practice continuously use web services in novel ways. Resulting problems concern service description, service discovery and location, and interactions of services when composed, among other things. To deal with these problems, a pragmatic approach is required. The syntactic (structure) and semantic (structure-based meaning) levels of analysis are still needed, but also the users'context-dependent needs should be taken into account in the sense that service description, discovery, and invocation are tied to the context of the intended compositions [20].

Returning to our research problem of web service selection in virtual communities: how to ensure that the selection process is pragmatic? Given that a formal functional match is performed at the semantic level, what does pragmatic selection mean? What makes up the community context in the pragmatic selection process? We claim that this is a process of *interpretation*, in which the relevant stakeholders assess matched services on a wide range of quality of service considerations, assessments that cannot be automated, but require human expertise, skills, and diplomacy, as demonstrated by, for example, the indispensable roles of students and computer centre representative in final decisions on the applicability of CourseInfo modules

However, community intentions are much harder to capture than those of individual users. For example, users can be interviewed in a prototyping session by the implementor, or simply be asked which interface they prefer. Of course, these specifications may still be

[4] http://www.w3.org/RDF

distorted and not capture the essence of the problem, but at least they are individual distortions that can be traced back to a particular person who is then capable of reframing her requirements. Not so in a community, virtual, or otherwise. Communities are typified by the bonds between members, the shared interests, and norms [24]. Changes to the socio-technical system can have far-reaching consequences for the efficacy of community operations.

One major problem specific to pragmatic *community* information systems development, and thus also to service interpretation, is *who* to involve in the development process [21]. It is not sufficient to have a software engineer make a model of the community, select some web services, and, automatically, an information system has been created. First of all, change processes are subtle and continuous. Much tacit knowledge is needed to interpret the need for changes in the socio-technical system, and to produce the actual specifications. It is therefore essential that it is known when and exactly in what role community members are to take part in the specification process of their requirements and web services used, so that the sociability of the community information system can be ensured [5].

Norms, defining acceptable behaviour, are a key element in any community. They define which workflow and evolutionary behaviour may, must, or may not be performed by which actors [21]. Online communities use norms of behaviour (or policies) to guide the interactions of members, for example in the form of tacit assumptions, rituals, protocols, rules, and laws [5,24]. Norms are powerful regulatory constructs in communities, especially where people are not governed by traditional organizational hierarchies, as is true for complex knowledge creating online networks. Thus, for modelling pragmatic selection support, *change* norms can be used to define the governance of the community IS, amongst other things by letting communities clearly define who is to be involved in their selection processes.

Example of Pragmatic Selection Support

In a courseware development community case, a student may notify the community that a particular tool for group discussion, such as Virtual Chat, is not efficient. The decisions about whether and how to select a new tool are to be made by, for instance, the lecturer for assessment of workflow impact, and the computer centre representative for technical and security considerations. Thus, key at the pragmatic level of the selection process is that enough context is provided that legitimate selections of web services can be made by the relevant members of the community. We call this the problem of guaranteeing the *legitimacy* of specification changes, meaning that they must be both meaningful and acceptable. This problem is addressed within the RENISYS methodology [21,25]. RENISYS is a legitimate user-driven approach for community information system specification. Its main components are a set of *ontologies* to model the structure, operations, and evolutionary processes of socio-technical system of the community; a mechanism for defining and using *composition norms*, which define who is to be involved in which particular stage of what particular type of change to the socio-technical system; and a *conversation* module to support discussions about proposed specification changes.

To ensure the acceptability of specification changes to the socio-technical system, in this case concerning the selection of web services, the selection of the *relevant user group* is key. In [25], we discuss in detail how to do this. Essential is that the specification process is seen as change process of socio-technical knowledge definitions, to be guided by composition norms defined by the community itself. To find out which users to legitimately involve in the initiation, execution, and evaluation of a particular change process (the compositions), a set of *applicable composition norms* is calculated for each user and composition. For each of these sets, a *resultant deontic effect* can be calculated (the details of which are outside the scope of this paper), prescribing whether a user may, must, or may not be involved in the composition. For example,

$$\mathbf{de_r}\,(D_{CN\_APPL}(\text{Jane}, \text{Exec\_Select\_Type}(\text{Discussion})) = \textit{Req}$$

means that user Jane (who may play several roles, like educator, developer, etc. in the community) is required (= must) participate in the actual selection of discussion services. Using the ontologies of service types, it can be precisely determined, based on the community's own composition norms, which users when to involve in a particular service selection. A powerful feature of RENISYS is that norms can be defined at different levels of specificity. For example, an informal, egalitarian community may have only one, very generic norm, saying that all members are to be involved in all stages of all selection processes. In more realistic situations, some selection processes will be defined at a very generic level, while others may be defined at great level of detail. Thus, in RENISYS only (and all) relevant users are included, the criteria for relevance of course defined by the community itself in their own composition norms. The legitimacy of web service selections can thus be increased considerably.

4. A Meta-Model of Web Service Selection Support

We have defined the web service selection process to consist of three subsequent steps: syntactic discovery, semantic matching, and pragmatic interpretation of the proposed services. The syntactic step serves to select those services from a repository system, that have high structural resemblance with a target specification in terms of their interface. Semantic matching builds on top of syntactic discovery of services and aims at retrieving specifications with a high overlap in terms of the ontological terminology that has been adopted in a particular community. Lastly, pragmatic interpretation places the results from the syntactic discovery and semantic matching in context by allowing relevant members of the community to assess their usefulness. In order to facilitate the making of a quick scan of how well the web service selection process is supported, we have summarized our approach in a meta-model of web service selection support. The meta-model connects each semiotic view to a selection subprocess. Furthermore, we make a distinction between *notation* and *method* at each level. This because many efforts focus on terminology development (e.g. DAML), but not on the method with which to provide better support for the selection process.

To illustrate, we apply the meta-model to the ad hoc courseware development case of Sect. 2. We can see that there is support for the syntactic level, but notation-wise only: in order to select services, the syntactic labels of the CourseInfo modules have been used. There is virtually no support for the semantic level for both notation and method. At the pragmatic level, no specific notation was used, but students and computer center representative were involved in an assessment of how well the CourseInfo modules allowed the making of group assignments, so there was some method.

In Fig.1, we apply the meta-model again, but this time not to represent the average, ad hoc, courseware development situation, but the way state-of-the-art web service selection support would look like. This would include syntactic notations such as UDDI, BPEL, and WSDL (for an extensive treatment of these technologies, we refer to [19,26]. A semantic matching (including syntactic discovery) method could be provided by BALES, which has only a thin selection ontology, but much attention for the methodological approach in which matching is to take place. The same goes for RENISYS, which has only a basic domain ontology and attention for semantic definitions of

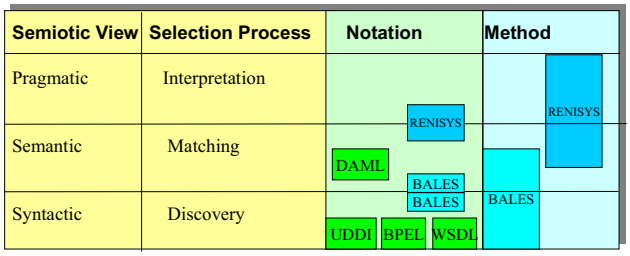

Figure 1. Applying the metamodel to state-of-the-art selection methods

workflows, but focuses its attention on providing a comprehensive and sound pragmatic method. Still, we can see in the picture that all these methods are isolated: notations and methods are not connected, terminology nor process-wise. Future state of the art support methods for web service selection would therefore show much more coverage of – and interconnections between – the various notation and method cells. For example, if BALES and RENISYS would share a (partial) ontology, semantic matching could be automatically done on ontological terms defined as relevant in RENISYS conversations for specification.

5. Conclusions

Web service selection constitutes a critical stage in web services-based IS development. From a semiotic point of view, it encompasses a syntactic service discovery stage, a semantic service matching stage, and a pragmatic service interpretation stage. In this article, we have introduced a meta-model of web services selection support for quickly charting how well this complex process is supported, notation and method-wise. Most service selection processes currently are ad hoc, as demonstrated by our typical case of courseware development communities. To increase the efficacy of community IS development, more systematic support is essential. Our - still rudimentary - meta-model allows the quality of provided selection support to be analyzed.

Another important contribution of this paper is the focus on the pragmatic selection of web services. In this paper, we have only hinted at the complexities of the roles that norms play in the selection process. Very subtle norms often exist, defining permitted, required, or forbidden behaviour between many different stakeholders. Norms can be generic and apply to many different interactions, or specific, and apply to only few stakeholders in a few situations. Norms can interact and conflict, leading to complex *resultant*

deontic effects. Communities of different types may require completely different norms for apparently similar selection processes.

In summary, many *partial* approaches exist, especially at the syntactic and semantic level, which are relevant to web service selection in virtual communities. What has been lacking is an analytical lens to clarify their exact role in the overal selection picture. Our purpose was theory construction, not testing. We did this by clearly describing the problem of service selection, and grounding the solution in a theoretically sound semiotic perspective. The resulting meta-model allows for the systematic identification of gaps in selection support. We illustrated its validity by applying it to a realistic case.

In future research, we will focus our attention on the development of reference models of selection norms and support methods in different types of educational and other communities. It would be interesting to see which norm and support patterns emerge that can are invariant across communities, and which ones need to be tailored to a particular subtype of community.

References

[1] Kozinets, R. V. E-Tribalized Marketing?: The Strategic Implications of Virtual Communities of Consumption, *European Management Journal* **17**(3), 1999, pp. 252-264.

[2] Wellman, B. Computer Networks as Social Networks, *Science* **293**, 2001, pp.2031-2034.

[3] Smith, M.. Tools for Navigating Large Social Cyberspaces, *Comm. of the ACM* **45**(4), 2002, pp. 51-55.

[4] Schubert, P. and M. Ginsburg. Virtual Communities of Transaction: The Role of Personalization in Electronic Commerce., *Electronic Markets* **10**(1), 2000, pp. 45-55.

[5] Preece, J. *Online Communities : Designing Usability, Supporting Sociability*. John Wiley, Chichester, NY, 2000.

[6] Surman, M. and Wershler-Henry, D. *Commonspace: Beyond Virtual Community*, Pearson, 2001.

[7] Sawyer, S. A Market-Based Perspective on Information Systems Development, *Comm. of the ACM* **44**(11), 2001, pp. 97-102.

[8] Gongla, P. and C. R. Rizzuto. Evolving Communities of Practice: IBM Global Services Experience, *IBM Systems Journal* **40**(4), 2001, pp. 842-862.

[9] Friedman, T. Next, It's E-ducation, *New York Times*, 17 november, 1999.

[10] Roschelle, J. et al. Developing Educational Software Components, *IEEE Computer* **32**(9), 1999, pp. 2-10.

[11] EPOC Working Group. *Towards a Framework for Open Courseware: The Third Report of the TLTP Working Group on Open Courseware*. Teaching and Learning Technology Programme, Bristol, 1996.

[12] Werry, C. The Work of Education in the Age of E-College, *First Monday* **6**(5), 2001.

[13] Hiltz, S. R. Collaborative Learning in Asynchronous Learning Networks: Building Learning Communities. *WEB98*, Orlando, Florida, November 1998.

[14] Newmarch, J. Lessons from Open Source: Intellectual Property and Courseware., *First Monday* **6**(6), 2001.

[15] Rasmussen, M. *IT Trends 2003: Information Security Standards, Regulations and Legislation*. Giga Information Group, December 18. 2002.

[16] Nonaka, I. and Takeuchi, H. *The Knowledge-Creating Company: How Japanes Companies Create the Dynamics of Innovation*. Oxford University Press, New York, 1995.

[17] Fremantle, P., Weerawarana, S., and Khalaf, R. Enterprise Services, *Comm. of the ACM* **45**(10), 2002, pp. 77-82.

[18] Brooks, F. P. *The Mythical Man-Month : Essays on Software Engineering*. Addison-Wesley, Reading, MA, 1995.

[19] Walsh, A.E. *UDDI, SOAP and WSDL*, Prentice Hall, 2000

[20] Singh, M. P. The Pragmatic Web, *IEEE Internet Computing* **6**(3), 2002, pp. 4-5.

[21] de Moor, A. Language/Action Meets Organisational Semiotics: Situating Conversations with Norms, *Information Systems Frontiers* **4**(3), 2002, pp. 257-272.

[22] van den Heuvel, W.J. Integrating Modern Business Applications with Legacy Systems, *PhD Thesis*, Tilburg University, June 2002.

[23] Miller, G.A. WordNet: A Lexical Database for English, *Comm. of the ACM* **38**(11), 1995, pp. 39-41.

[24] Talbott, S. *The Future Does not Compute : Transcending the Machines in our Midst*. O'Reilly, Sebastopol, CA, 1995.

[25] de Moor, A. and M. A. Jeusfeld. Making Workflow Change Acceptable, *Requirements Engineering,* **6**(2), 2001, pp. 75-96.

[26] IBM. Business Process Execution Language (BPEL) for Web Services, Version 1.1, *IBM Technical Report*, 2003