# From trading to eCommunity population: Responding to social and contractual challenges

Lea Kutvonen, Janne Metso, Sini Ruohomaa
Department of Computer Science, University of Helsinki, Finland
( Lea.Kutvonen | Janne.Metso | Sini.Ruohomaa )@cs.Helsinki.FI

## Abstract

*The emergence of networked eBusiness and the wave of service-oriented computing facilities create new challenges for automating inter-enterprise business process management and eContracting. This development leads to strategical benefits for agile enterprises, but also to new challenges on enterprise system architectures and platforms. This paper discusses the techniques of introducing trust-related decisions into eContracting, and their effects. This work enhances the web-Pilarcos project results on B2B interoperability middleware; the architectural model supported comprises of autonomous business services forming loosely-coupled, eContract-governed eCommunities.*

## 1 Introduction

In the current trend, electronic business networks are built from autonomous business services. This trend is to be seen in the use of Web Services [2], various consortia standards on inter-enterprise business process management (e.g., [16, 27]), and in the rise of service-oriented architecture (SOA) [18, 26]. It can also be seen in the number of eContract-related research projects in action (e.g., [1, 3–6, 8, 13, 25, 28]).

We call the collaborative, inter-enterprise business networks eCommunities. They are established dynamically to serve a certain business scenario or opportunity. Their operation is governed by an electronic contract negotiated dynamically by the participants. Evidently preliminary work, such as initial trust relationships, is needed before these electronic contracts can be formed. Given the necessary prerequisites, the eCommunity is established by multilateral negotiations on the properties of the business network.

Our contribution is to provide generic middleware services for inter-enterprise collaboration management [11, 12]. Within this frame, the contractual aspects addressed range from information representation issues to technical and business aspects. This range makes the suggested solution differ from most eContracting proposals.

The management services include a number of pervasive functions as follows. First, tools and repositories support developing and publishing of new models for business networks, and defining new service types for business services in such a way that the service types match the needs of the business network roles [24]. Second, service offer repositories enable enterprises to publish business services to the open service markets together with metainformation for automated matching to roles and for interoperability testing against peers in the business network [12]. Third, means are required for declaring policies that govern the use and the availability of business services. Fourth, new protocols are needed for negotiating eContracts to govern a new business network [11]; the establishment phase is partially performed by a third-party population process, partially by a collective, refining or dropping-out negotiation protocol between becoming peers. Finally, facilities are needed for monitoring the behaviour within eCommunities and manage breaches within them as specified in the eContract [14].

We believe that by this kind of generic B2B middleware services that are available through private agents at each enterprise, the right kind of software investment cycles can be supported. The middleware services themselves are separated from the application software, thus making applications less dependent on the platform technologies. At the same time, the granularity of provided services grows to the level understandable at the business strategies level; understanding the relationship between business services and the computational counterparts is a necessary requirement for controlling them [10]. Furthermore, the development of B2B middleware and SOA-guided eContract-based architectures require the separation of various business and technical concerns in the contracting process, for example, security, trust and reputation, and business policies.

This paper focuses on the business network establishment phase in which decisions on required interoperability are done and enhances it by addressing issues of trust management. The middleware agent that performs the analysis

is called the populator, and its task is to fill the different roles of a business network model with service offers of acceptable types, and to check that the selected services are able to interoperate. In the present situation, the importance of the populator lies in its ability to check interoperability conditions, but not in becoming an automated contract initiator with new partners from open service markets. The main hindrance in automated selection of partners is the lack of trust in unknown service providers and the lack of any framework contracts to govern the service markets.

This paper discusses the effects and techniques of introducing trust-related decisions into eContracting. Trust is evaluated between peers, while the middleware layer should provide a trustworthy platform from which trustworthy information can be retrieved, and where trustworthy private agents are running. A secure communication infrastructure is assumed to be in place.

This paper is structured as follows. Section 2 discusses forming dynamic collaborations from open service markets. The populator functionality and its implementation are discussed in Section 3, while Section 4 introduces the trust concepts and discusses embedding trust considerations into the populator functionality.

## 2 Addressing social needs by eContracts

Establishing new eCommunities from business services at the open markets rises problems that can be considered as social: The interoperability demands between partners emerge to sharing external business processes, meeting on business value, understanding the pragmatics of policies, and furthermore, embedding management of trust between potential collaborators.

At present, there are no commonly accepted eContract structures that would sufficiently cover the various business and technical aspects of the eContract. We believe the necessary aspects should be captured within a common upper-level ontology that is further refined with published business network models. Final details are rolewisely added from service offers as partners enter eCommunities. In addition, the eContract structures should address the needs of eCommunity membership and life-cycle management at runtime, including interoperability testing and monitoring.

The business network models, specific to their business-areas, should define a sufficient structure for each eCommunity type to support the actual eContracting (negotiation, establishment, monitoring). These models bring in aspects of regulatory systems, business targets, and common practises; the descriptions of available business services in turn define the limits within which the providing enterprises are willing to assume responsibilities in the potential eCommunities. Information related to the eContracts becomes defined by designers, policy creators, service implementors, and enterprise system owners in separate steps of systems engineering and use. Fig. 1 illustrates the flow of business-related and technology-related metainformation in the eContracting process in the fundamental steps of metainformation and software production processes for inter-enterprise collaborative systems [10, 24]. The elements are described below.

A business network model defines the topology of an eCommunity in terms of roles and interactions between them. A role is a placeholder for a business service: the role definition sets direct requirements with which the service types must conform, and it can, in addition, define assignment rules for other features, for example non-functional aspects or the identities of the participants acceptable for the role. The interaction declarations set conformance requirements for the business processes to be executed between participants. The design of business network models is a profession on its own, requiring understanding of regulatory frameworks on the business area, best business practises, and strategical methodologies suitable for the business.

A service type defines the syntactical structure of interfaces, the semantics of documents to be exchanged, and the service behaviour in terms of the local business process, as observed outside of the software module providing the service. For each service type, there is a set of associated properties that are required for each service offer for this type. A service offer is a declaration of a provided service, naming its service type and giving values to the required properties.

A computational service is a collection of business-relevant software modules. However, it has been a design aim here that the software elements do not need to consider the business strategies or policies. Instead, the runtime environment provides metainformation-driven monitors for governing the software elements. We call the combination of the monitor, the governing rules, and the computational service a business service. It should be noted that a part of the governing rules are public as well as a part of the eContract, while others are private and known only to the provider of the business service.

While the eContract structuring by business network models capture most social behaviour requirements in the eCommunity, we must consider other layers of interoperability simultaneously. We understand interoperability, or the capability to collaborate, as the effective capability to mutually communicate information in order to exchange proposals, requests, results, and commitments. The term covers technical, semantic and pragmatic interoperability. Technical interoperability is concerned with connectivity between the computational services, allowing messages to be transported from one application to another. Semantic interoperability means that the message content becomes understood in the same way by the senders and the receivers. This concerns both information representation and messag-

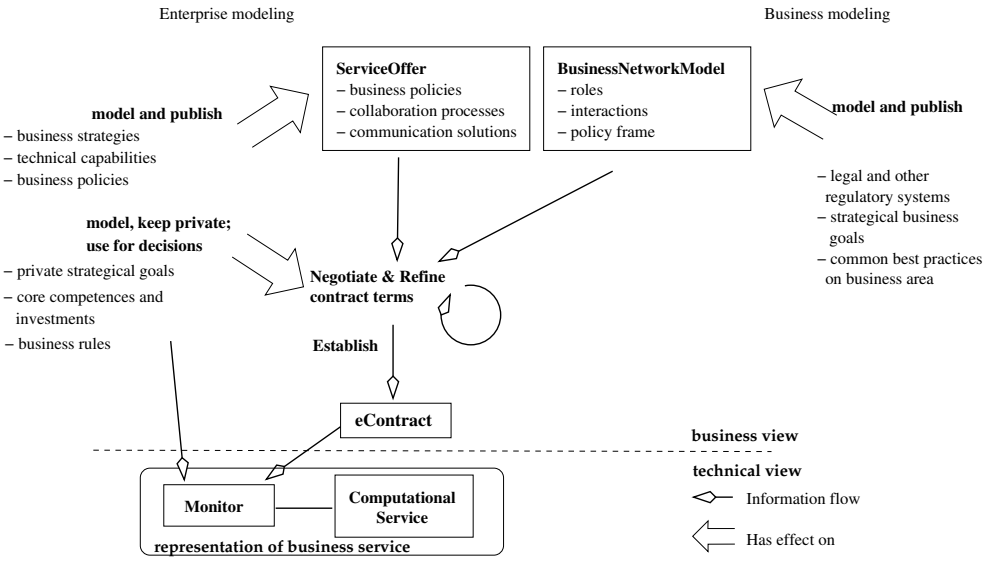


**Figure 1. Information flows for building eContracts and business services.**

ing sequences. Pragmatic interoperability captures the willingness of partners to perform the actions needed for the collaboration. This willingness to participate refers both to the capability of performing a requested action, and to policies dictating whether it is preferable for the enterprise to allow that action to take place.

To capture these interoperability levels, we use the five ODP-RM viewpoints (Open Distributed Processing Reference Model) [9] to structure the metainformation in service offers and eContracts. The Enterprise viewpoint is focused on defining the roles and interactions needed between them in order to reach the goal of the community. This corresponds to the definition of external business processes and policies over the eCommunity. The Information viewpoint is for defining the information repositories and the exchange of information elements, as well as calculi for invariants and well-formed changes of the state of the information. The Computational viewpoint is for defining the computational services involved with the community, in terms of interfaces and behaviour towards them. The techniques for describing and comparing behavioural types of services are still immature [24]. The Engineering viewpoint is for expressing how the computational services and the supporting infrastructure are to be used. The Technology viewpoint is for expressing which standard solutions are required for computing or communication platforms, or information exchanges.

The brief analysis above brings us to structuring eContracts and service offers as shown in Table 1. The eContract is structured according to the roles defined in the business network model, and refined by instructions found for each service type required in the roles. The final level of detail captures the requirements on the technical communication. The eContract must also address breach detection and recovery by choosing a published model for that.

In contrast to some upper-level ontology development initiatives, where the aim often is to define a universal contract structure, we consider the business network model developed for a specific business domain as the right scope for the "universe of discourse" when defining contract structures and ontologies. First, the full range of elements affecting interoperability is not present. Due to the autonomy of service providers, part of the knowledge is private, and failures to conform to the category-forming selection criteria or monitoring rules will raise issues to be addressed by breach recovery processes at the community level. Second, the structure of an eContract is not defined by one template only, but the construction rules for the eContract structure are retrieved from the business network model, service type descriptions, and service offers.

To pair up with this structure of the eContract, the corresponding protocol stack is depicted in Fig. 2. The main difficulty to overcome here is that each stack layer involves different set of participants. The technology level protocols are used by the peers in the business network to fulfil basic communication interoperability needs, while service level protocols are used between potential peers and the open service market to determine the compatibility of single services. The community level business processes are used to manage the dynamicity and interoperability of the business network as a whole. Besides this, the architecture must support mapping of the business rules and enterprise policies of the members of the eCommunity to the community management protocols on the layer below. Even contract breaches should be resolved by community-level business processes.

**Table 1. Technical structure and XML-tags for eContract contents.**

| Contract element label | Information type and source | Explanation |
|---|---|---|
| **Identification and state management** | | |
| contractID | String assigned by the initiating NMA | Identity for the eCommunity; potentially jointly with sessionID |
| description | String assigned by the BNM designer | Describes the purpose of the business network model in a sufficiently detailed level for generating monitoring rules. |
| startDate | Set by initiating NMA during the negotiation process | If the contract validity is time-triggered, the startDate and endDate are used, indicating date and time. |
| endDate | Date and time, as above | |
| state | Integer upkept by the NMA. The eCommunity life-cycle is controlled by a state machine with states of populated, in-negotiation, agreed, established, in renegotiation, terminated. | During the established phase the progress of the conversations (external business processes) can be viewed as steps of considerably large task blocks. |
| **Management of repetitive execution of eCommunity behaviour** | | |
| sessions | Array of contractSessions where elements encoded in string-valued tagged fields | Each ContractSession element contains the contractID and sessionID within that contract, identifier for the current epoch, and an integer coded state indicator. |
| allowedSessions | Integer, not mandatory | Maximum limit of sessions for this eCommunity. |
| usedSessions | Integer | Counter for controlling the max limit. |
| concurrentSessions | Integer | Limit for maximum number of concurrent sessions. |
| **The eCommunity structure and behaviour** | | |
| businessNetworkModel | String | Identifies the correct model in repository |
| participants | Array of participantInfo; participantInfo elements encoded as string-valued tagged fields | A participantInfo element contains service offer information, especially logical and technical addresses of communication end-points for the participants, the management interface location, the partner's electronic signature, the role it is associated with and whether this participant is the coordinator or the eCommunity. |
| bindings | Array of logical connections assigned by NMAs | Reference to the binding type for the mediating channel; provides technical requirements. |
| modelPolicies | Array of policies; policies expressed as a name-value pair. | Policies governing the eCommunity over all epochs. |
| architecturePolicies | Array of policies | Policies governing the eCommunity during one epoch. |
| rolePolicies | Array of policies | Policies governing each role in an architecture. |
| globalRecoveryProcess | Array of process references | Process models are available in the type repository. |
| conversationRecoveryProcess | Array of process references | |
| roleRecoveryProcess | Array of process references | |

Therefore, the community level processes form a backbone for interoperability and collaboration management, placing high demands on the supporting middleware to enable that. In addition, the lack of workflow enactment in the stack is intentional. The business applications are expected to execute their private (local) business processes independently, only interacting according to a monitored external business process. As the coordination approach here expects business services to be able to initiate the necessary activities themselves, only breach detection and recovery processes are needed. The essential failures of service behaviour that we should expect to address are involved with various non-functional aspects (NFA), such as trust, security, QoS, or discrepancies between business policies of autonomous participants.

As the populators, repositories and business network

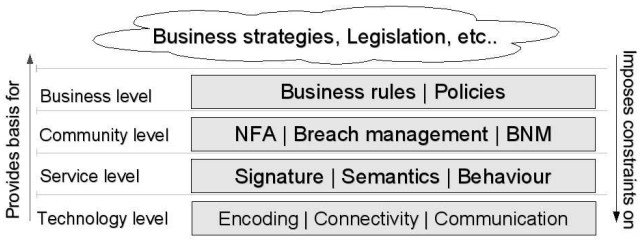


**Figure 2. Interoperability management [23].**

agents have an essential role in manipulating the eContracts, we cannot omit discussing the trust placed on this eContracting infrastructure.

The metainformation elements the infrastructure provides through repositories must be trustworthy to begin with. If the validity of model and typing information cannot be relied on, the information storage becomes useless. A populator builds on the model and typing information to refine it into business network proposals. The refined information is passed to a network management agent, which controls the negotiation phase finalising a business network and an eContract to govern its operation. Trusting the eContracting infrastructure requires strict control over the type repository and business network model repositories. Before published entries can be stored, they must be validated, also in relation to the existing entries. The asserted relationships between stored entries must remain consistent.

These kinds of repositories have a considerable organisational effect as well: they provide a means to regulate electronic service markets. Service offer repositories can be controlled by requiring well-formed offers, or even requiring certified enterprises to test offers before accepting them. However, these kinds of methods do not change the fact that the service provider remains autonomous, and its actions in the eCommunity may not be in accordance to the service offer or the negotiated eContract. In other words, trust in the infrastructure does not directly imply trust between potential partners in the eCommunity that is being formed. Trust between eCommunity partners is a concern of its own, and is one of the aspects to be included into the eContracting process.

The populator uses the type and service offer repositories to produce interoperable business network proposals. Like the repositories, population can be provided as a service by a third party, although a peer implementing a populator for itself is not unfeasible either. A populator must be trusted by the initiator of an eCommunity to match the business network model and service offers as specified, but no further. The populator operates on published information only, and it is not necessary to trust it with for example private partner preference policy, unless there is a benefit in doing so. The populator is not told which of the proposals it produces is accepted in the end.

A network management agent (NMA) represents an eCommunity member in the business network [14]. It handles negotiations with potential new members and renegotiations if members are changed, it upkeeps state information for the eCommunity, and determines the suitable reaction to the information passed to it by local monitors. For example, if the monitors detect a breach of the terms of the eContract, the violation can at worst lead to a reorganisation of the business network. Every member of the eCommunity has its own network management agent, and they are considered to be fully trusted local agents.

In order to bring trust considerations into the decision processes, support for trust management mechanisms must be added into the infrastructure. Our approach is based on a dynamic combination of experience information and a subjective analysis of the situation in which trust is needed. Earlier experience with the eCommunity member being evaluated is gathered both locally and received through a global reputation network, and it forms a basis for predicting the member's future behaviour. On the other hand, subjectively estimated risk and tolerance for it depend also on various factors not directly dependent on the particular member being evaluated, and our model contains factors to accommodate for these considerations as well.

From the business point of view, there are two contradictory requirements for making business services available. On one hand, it is preferable to have all potentially marketable services openly available for all potential clients and collaborators, while on the other hand, the integrity and privacy of enterprise ICT systems require efficient access management, secure transfer of information and strict authentication procedures.

In open business networks, traditional hard security falls short in protecting an enterprise, because it divides other actors too narrowly into those trusted (authenticated and authorized) and untrusted (all others), with little ability to adjust to for example the misbehaviour of trusted actors. Social control methods, such as trust management, allow the system to be more open for collaboration, while still protecting itself both from unknown actors as well as those authorized for the time being [20]. In the centre, the service itself is aware of its required integrity and security constraints, and refuses access that would break these limits, regardless of the requestor.

The following sections go into detail on how trust in peers can be taken into consideration in eCommunity establishment. A corresponding analysis of trust-guarded transactions and actions triggered by trust-breaches during the eCommunity lifetime has been performed separately, see [22].

## 3 eCommunity establishment

We use a two-phased approach in eCommunity establishment. First, a populator is used to match multiple service offers into a frame formed by a business network model. Then, the eCommunity participants are further negotiated based on the proposed eContracts. The negotiation is performed by network management agents, NMAs, that represent each enterprise.

The populator is responsible for providing a reliable facility to produce interoperable sets of service offers in such a way that they fulfil the requirements of a selected business network model. The interoperable set of service offers means that based on the network model, each service that must communicate with each other can do it technically and semantically. The willingness of the participants to interoperate (i.e. pragmatic interoperability) is not considered during the population process and it will be determined at a later time during the negotiations.

The populator chooses the most suitable service offers for each role. First of all, the offers must be of an acceptable service type for the role. The selection is based on client defined constraints to roles, constraints imposed by the network model itself, service offers included by the client, and restrictions on the service providers which can be used.

The population process results into a set of eContract proposals, still requiring a negotiation round amongst the proposed partners before the eCommunity establishment phase is completed. The protocols in itself is simple, the populator client sends out a proposal to all partners referred to in the proposed eContract. These peers can respond by accepting the proposal, or making a refined proposal, or rejecting. The responses are sent back to the initiator for combination and further refinement cycles or initiation of a new round with the next eContract proposal.

The populator is used by network management agents (NMAs), which represent an organisation in the community. Each participating organisation has its own NMA. The NMAs are used to create the population requests and get back the results from the populator. The populator returns a specified number of compatible sets of service offers or all of the available ones. The number is specified in the population request. Then the initiator, which made the population request, chooses one of the business network proposals for further negotiation. During the negotiations, the participating organisations refine the contract terms until they are satisfactory. The NMAs run the negotiations among themselves and consult users for decisions. The technical environment of the populator is created by the other web-Pilarcos middleware services. The middleware environment is described in [11, 12].

As a representative of open service markets, the populator uses a service offer repository, a service in many respects

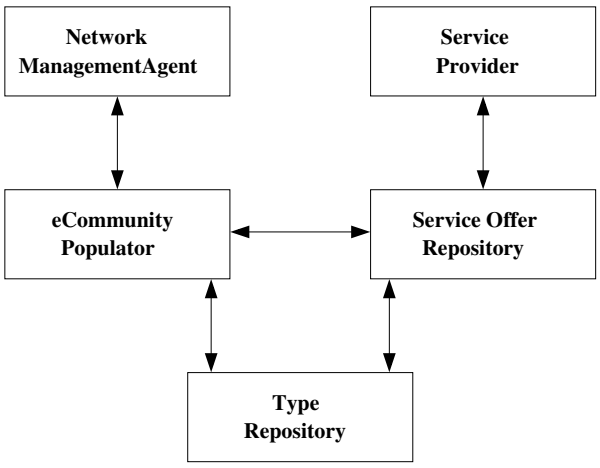


**Figure 3. The populator's environment [19].**

similar to UDDI [15] or CORBA Trader [17]. The significant addition to a basic trading facility is its ability to match multiple service offers to a interoperable virtual organisation. The populator uses a metainformation service, a type repository, to achieve this functionality. For the populator to be reliable, the metainformation must be reliably managed and consistent.

The populator and its connections to other services surrounding it are shown in Fig. 3. The populator and the service offer repository work in very close cooperation with each other during business network population. The populator uses the type repository for business network model (BNM) discovery and uses the BNM and its topology for the service offer interoperability checks. Service providers use the service offer repository to publish their services. During the publishing, the offers are verified to be consistent with their alleged service type.

The population request carries two information elements. The first, general part includes a reference to the business network model to be used during the population and directions for the populator for selecting service offers for any of the roles. These directions can advise on the desired number of returned sets of offers, or the maximum time the populator can use for searching the interoperable sets. The directions can also restrict possible service providers or attribute values. The populator client can also refine the properties expressed in the business network model. The model itself expresses requirements for the eCommunity participants, for example, the offers can be required to indicate capability to support transactions. The second part expresses advise on filling each role separately and can include a pre-selected service offer, or directions to use specific selection criteria, or role-based utility functions. The client can also fill in service offers for known partners which will participate the following virtual organ-

isation. The populator respects these preliminary choices made, and even makes use of the knowledge by restricting the potential search space accordingly.

Although the populator client is not required to include its own service offer in the population request to represent its own role in the business network, this is beneficial. The included offer will go through the same checking process as all other service offers that will be considered for the business network. At the same time the included service offer and its attribute values acts as the starting point of the properties for the virtual organisation. Similarly, there are properties in the business network model which have an effect on the eCommunity and its properties.

Population is based on a eight-step algorithm [19]:

1. Receive population call
2. Retrieve the business network model and service types referred to in the role descriptions from the type repository
3. Create role populators, set utility functions
4. Request matching service offers for roles from service offer repository using all appropriate service types
5. Check the interoperability of pre-filled roles
6. Find service offers for each role
7. Walk through the search tree and test the interoperability of service offer combinations in the roles
8. Return business network proposals.

In the second step, the populator requests the model from the business network model repository. The model infers the roles and properties of interest. If there is a conflict with the properties of the business network model and the properties given in the population call, the population algorithm is terminated.

For the third step the populators creates role populators for each role in the network model to contain role specific information. The information includes current limits of the attribute values, the available service offers based on the attribute values, and if a service offer is selected for the role or not. Utility functions are set as defined in the call; general utility functions are individually set to each role.

Steps from four to six can execute concurrently. During the fourth step the role populators request service offers from the service offer repository for their own role. The request contains the current restrictions for the role and the service offer Repository only returns matching offers. While the role populators are waiting for the offers, they check the interoperability of the pre-filled roles and their service offers. If the pre-filled roles cannot interoperate, the populator terminates the population request.

The sixth step forms the main body of the populator. The population advances as a depth first search by selecting a matching service offer based on the requirements from available offers and locking it to a certain role. The locking restricts the possibilities to other roles further and the new restrictions are propagated to other role populators. When the role populators receive the restrictions, they remove the mismatching service offers temporarily from the possibilities. The temporary removal allows the process to roll back in case one or more remaining roles have no possible offers left. The locking of service offers to roles is repeated until every role has service offer locked, all possible combinations are exhausted, or the time limit is exceeded. During the population the role populators can request more service offers from the service offer repository.

Finally the populator returns the business network proposals to the requesting network management agent. The number of proposals depends on the requested amount on the call and the service offers. The populator cannot guarantee that it finds the requested amount of proposals.

During the aforementioned depth first search, the algorithm tries to select the service offer one role at a time. This creates a search tree of the service offers. Depending on the propagation of the selection criteria the size of the search tree varies. Because the selection criteria changes every time a certain offer is selected for a certain role, the size of the remaining search tree changes as well. By taking this into account the populator is able to reduce the search time to find suitable business network proposals. The populator has several variations for doing the criteria propagation, but we only illustrate the technique called *forward checking*.

The populator uses *attribute frameworks* to manage chains of attributes in the roles of the network model that must all have the same value, because the value has an effect on the interoperability of all roles in the chain. An example of such a requirement is transaction support along the whole supply chain. Essentially this means that each service offer must have the same attribute value for a given set of attributes if a role is a part of an attribute framework. Attribute frameworks make the propagation of constraint values easy, and they enable the populator to detect which attribute values effect which roles.

Each role populator maintains a set of service offers suitable for the business network. Role populators are able to query several service offer repositories at the same time. The amount of service offers received from each repository varies, as does the time to get the offers. Because of this the role populators only request a certain amount of offers during the setup phase. Once the already requested offers are exhausted, the role populators will request more. The offer queries are built in a way that allows the service offer repository to determine which service offers have already been sent to the role populator. The queries are also always

made with the starting constraints included. By doing this the role populators can be sure that the requested offers will match even if the process has to be rolled back. The role populators will temporarily remove offers that do not match the current criteria, however.

The populator propagates the new constraint values to the role populators after a role is filled with a service offer. The propagation itself is done by copying the new attribute value limits from the just filled role populator to the remaining unfilled role populators with the help of attribute frameworks. At this point the role populators representing unfilled roles temporarily remove all mismatching service offers from their set of available offers. Using this kind of strategy, the remaining offer tree gets rapidly smaller each time a role is filled with a service offer. If the mismatching offers were not removed before a certain role is reached in the search, the search tree would be considerably larger and there would be many more combinations to check.

The technical contents of a service offer is described in Table 2. Service offers have mandatory typeIDs which define the service type of the offer. A service offer is created from port offers which represent the partial interfaces of the service. There are also properties to define the attributes of the service itself as well as the service provider.

The populator is able to match several different types of attributes while testing service offers. The main XML Schema simple data types are supported. These include all numeral types (such as float double, etc.) and string, anyURI, time, date, datetime, and boolean. In addition, there are a few different ranges which can be used. These are OneOf, SomeOf, Exactly, and NoneOf. The *OneOf* range means that any single one of the given values must be same and supported by the service offer. *SomeOf* means that a number of the given values must be the same but not all. *Exactly* means that all values must be the same that in other service offers. *NoneOf* is an exclusive range and means that none of the given values are suitable for the service offer. For continous values, the ranges are given as a minimum-maximum value pair and for non-continuous values the ranges are given as sets of values.

Utility functions are used to determine the benefit of including a given service offer to the eCommunity. Utility functions can be role specific, network model specific, or the client can do the population without them. The utility functions in the populator are defined as follows [19]:

U($a_1$, ..., $a_n$) = $\sum_i w_i f_i(a_i)$

where $a_i$ is a constraint on attribute i, $w_i$ is the weight of the attribute, and $f_i$ is the function to calculate utility based on the value of the attribute. The function returns a value from range [0,1]. The sum of the attribute weights are scaled to 1. It follows that the value of an utility function U is always from the range [0,1]. The higher the value, the higher the utility.

Even though the populator can use utility functions and tries first the offer with the highest utility value, it does not mean that the resulting business network proposal has the highest possible total utility. This is because the depth first search. For example, if the best offer for role two is chosen, the populator will try every possible offer to role three before selecting the second-best offer for role two. Therefore the best offer for role two can result as a lower utility on the whole than the second-best offer for role two. This all depends on the values of the attributes in a given service offer and the effect the values have on the remaining roles.

The populator has been found feasible to use for eCommunity discovery [10]. The performance behaviour of the populator is acceptable both in terms of delay and scalability. The performance of the populator is dependent on the constraint propagation scheme used. The forward checking model is efficient in reducing the size of the remaining search tree. The size of the search tree will effectively determine how many possible combinations are left at a given time during the population. The size of the tree is not consistent through the whole population. As more roles have been filled with service offers, the size of the search tree will decrease. If the process has to roll back a role, the tree will grow in size again. The main cost in this model is dependent on the efficiency of calculating new constraints on the service offer attributes and propagating them. These constraint values are always recalculated when a role is filled during the population. The utility functions are just another way of calculating the constraints on the service offers. However, the complexity of an utility function plays a factor when using them. The more complex the utility functions, the more time it takes from the populator to calculate the utility value.

Compared to traditional trading facilities such as CORBA Trader [17] and UDDI [15], the main advantage of our approach is the ability to match multiple service offers into a functioning eCommunity. This cannot be achieved using the traditional traders with just one request.

Compared to the CORBA trader, the populator has more elaborate service types. They both have similar white and yellow page information, but the green pages are much richer in the populator. They allow behavioural descriptions and thus can be used to verify the interoperability of two given service types. The CORBA trader has a simple type repository compared to the type repository in web Pilarcos middleware. The typing structure is similar and both support subtyping. The web-Pilarcos type repository verifies the type hierarchy and the interoperability of the service types using the richer typing information. As such the CORBA type repository cannot be used for interoperability checking.

The UDDI registry is a the standard Web Services discovery service. UDDI provides an extensive facility to describe the white, yellow and green pages. UDDI is designed

**Table 2. Technical structure and XML tags of service offers.**

| Element | Mandatory | Instances | Explanation |
|---|---|---|---|
| typeID | yes | 1 | Identifies the service type the offer is based on. |
| portOffer | yes | 1-* | Defines operations and their order regarding one port. Describes the properties of each port, and contains the pre and post conditions of each operation. |
| syncStruct | no | 1 | Provides causal relation of the events for synchronization. |
| typingContext | yes | 1 | Defines the typing hierarchy that contains the service type which is used by this service offer. |
| serviceProperty | no | * | Gives values to service attributes. Defines a name-value pair. The value can either be a single type or a value range. The attributes must correspond to the ones in the service type. |
| providerProperty | yes | 1-* | Describes properties of the service provider. The description is based on a common ontology. |

to support queries made by humans and does not suit well to automated searches. UDDI has a typing system but does not have a service like the type repository to maintain the consistency of the typing information. As a result it cannot be relied upon while doing automatic interoperability checks or verification.

## 4 Trust in eCommunity establishment

In this section, we describe methods for enhancing the eCommunity establishment process to consider trust and reputation of potential partners. First, the trust model developed in the TuBE project [22] is outlined, then various alternatives for the relationship between the peers' trust management systems and the populator are discussed, as the organisation affects both the security as well as the controllability of the resulting system.

### 4.1 Trust model

We define trust as *the extent to which one party is willing to participate in a given action with a given partner, considering the risks and incentives involved*. Trust decisions are made by each peer in two areas of the eCommunity life cycle. First, decisions are needed during the eCommunity establishment phase to determine the willingness to participate in the eCommunity. We will focus on this decision point. Second, trust decisions are made during the eCommunity operation, to determine whether particular risk-relevant commitment is deemed reasonable [22]. A trust decision is the result of a subjective evaluation of local information combined with additional third party experience information received via a reputation network.

Our trust model has 7 factors: *trustor, trustee, action, reputation, risk, importance* and *context*. The trustor, trustee and action parameters, together with the current state of the system, determine the situation the trust decision is made in. The party making a subjective trust decision, the trustor, is the guarded service, represented by an agent. The target of the decision is the trustee, another peer in the network. The action parameter denotes a group of SOAP messages exchanged, and a trust decision is triggered at a risk-relevant commitment point. For partner selection purposes, the action parameter can be seen to extend to cover the entire collaboration from a risk estimation point of view. Technically, however, it remains a set of messages exchanged with the populator, who in essence acts as a proxy of the actual trustee by suggesting it as a possible partner for a collaboration.

Reputation is the measure of a peer's perceived trustworthiness. It is based on a subjective view combined from experience information received from local monitoring as well as experiences reported by other peers through a global reputation network. The credibility and information content of the statements is evaluated by the recipient in order to build a local reputation value.

The risk factor provides a tactical cost-benefit estimate on the action considered. It expresses the potential benefits and costs of a positive trust decision to different assets, such as money, security and customer satisfaction. The information is stored as probability values for each severity class of effects to a particular asset, for example a 0.1 probability of a "considerable" loss of security, 0.3 probability of a "minor" loss and 0.6 probability of no effect. For for example monetary assets, a positive result is both possible and desirable. The action parameters and the reputation of the trustee affect this estimate, as well as context adjustments described later.

The importance factor represents strategic valuations in the enterprise, which are independent of any estimate of what the trustee might do. These considerations, such as the cost of denying an action defined in the eContract, or

the benefit of good service to creating a working partnership, guide the tolerance of risk.

The context factor represents temporary adjustments made to other factors, especially risk and importance. The changes can be initiated by any of three possible source types: the internal state of the peer's system, the state of the enterprise in general or the state of the business network the peer is a member of.

To produce a trust decision, the trust management system checks whether its completed risk analysis is within tolerated values for that situation. A situational cost-benefit estimate and representation of the tolerance for the particular situation are generated dynamically from the 7 factors, and a trust decision is produced by comparing the two.

For partner selection, an ordering between candidates is needed for those that pass the minimum threshold comparison. In order to determine which result pair denotes the most trust, i.e. the highest willingness to collaborate, a weighted sum of the differences between probability values in the risk tolerance and the risk estimate can be used, for example. Classes of trust may be more interesting than an absolute ordering, however, as parameters such as the price of a service are likely to be interesting criteria for ordering as well. Combining trust and price ordering in the grid environment has been experimented on by Griffiths and Chao [7].

### 4.2 Trust decisions for membership

In the web-Pilarcos middleware, the population of a business network can be provided as a service to the initiator by a third party [11]. This separation of concerns requires that the initiator trusts the populator to match given requirements as specified, similarly as service offer repositories must be trusted to faithfully reproduce offers stored in them. The populator has access to the public business network models and service offers. It produces a set of interoperable business network proposals, and passes them on to network management agent of the initiator, either as one set or a few at a time until the initiator is satisfied for one reason or the other. The populator does not know which proposal is accepted in the end, if any.

In the negotiation phase, the initiator passes the proposal, including the adjusted service offers and the network model, on to the selected members of the business network. They can then accept the network as is, reject it or modify the proposal to suit them better and return it. These modifications do not include changing members in the network; if the proposal is not accepted, another composition is proposed until no further options are available.

Trust can be used in the population process in two ways. Any member of the eCommunity can have a requirement for a minimum trust in other members in order to be willing to participate. In addition, the initiator can use trust measures as an ordering factor, similarly to price or other measures, when passing the proposals to the other members.

When trust information is used in connection with populating the network, there are privacy concerns involved in passing such information to the populator. Service offers and business network models are public information, but trust information includes private evaluations which can have averse effects if they become public knowledge. For example, a subcontractor may not wish to make its distrust in a large vendor known to the world, nor reveal details of its evaluations of risks and incentives related to a particular business network composition. Participants should therefore be able to set trust requirements related to their business network models and service offers, while retaining control of their private trust information. In addition, even these trust requirements should be made public only if it adds value to the process.

Trust requirement information can be made public in three ways: in a business network model by the modeller, as a parameter to the populator by the network initiator using the model, or recorded in a service offer by the provider wishing to become a member of an eCommunity.

Trust requirements can be brought into the population process without requiring participants to publish related information at all. In this approach, the populator provides business network proposals completely unaware of the trust requirements, and only considers public factors. In the end, it gives several proposals to the network initiator who passes them one at a time to the potential participants, asking whether they are satisfied with the proposal. This is a normal part of the negotiation, and during it the potential participants can adjust their offers further or reject the proposal altogether for any private reason. For example, the price range of a service can be narrowed to smaller values if other participants of the eCommunity are well-trusted, or to larger values if the venture seems more risky.

The main benefit of this approach is that the trust decisions are fully left to the parties with vested interests in their results. No subjective trust information needs to be passed around, and the requirements can be kept private. The main drawback is that trust information is introduced fairly late in the process. Some resources could be saved if the populator could be made capable of pruning candidates based on trust information before forming final proposals.

To make trust decisions, the populator needs a source of trust information: Either the populator has its own trust management system based on public information, or it pulls the necessary information from the interested parties.

As the populator is not an expert in anything else than population, it can only gain experience from third parties, and it will not have a model of the risk and benefit analysis of an action available. Should it make decisions on

its own, therefore, it can only base them on reputation information. In addition, it will not be able to tell by public reputation information only whether an actor that relies on it for pruning has a completely reversed subjective reputation view of a party. Reputation networks have many error sources; the biggest problem with blindly relying on one is that good reputation in a network is not predictable; it is also not a given that the interested party agrees with the reputation network.

Third party information used by the populator does not necessarily have to come from a reputation network. A different approach is to accept certification for certain kinds of basic trustworthiness, which may be achieved through for example insurances and guarantees. The benefit this approach has over the reputation system one is that a particular certificate of trustworthiness can be clearly defined. Service providers will then present their certification in their offers, and demands for certification are matched with the offers as with any attribute. Checking the validity of the certification can be done either directly in the populator or later by the interested party, for those potential members that have not been dismissed before for other reasons. However, certificate-based trust is different in nature to experience-based dynamic trust [21].

The populator can also draw upon the trust information of the interested parties as needed. This approach becomes especially interesting when an initiator provides its own populator service. In all other cases, there are privacy implications in passing sensitive trust information to the populator, which is generally not an ultimately trusted third party. To minimise the trust information that needs to be passed to the populator, the populator can ask for each actor only whether they fulfil the requirements or not. As the trust decision is produced by the interested party, the process remains opaque to the populator. The requirements can be unrelated to trust altogether, or positive judgements can be made at random and the proposal refused in the negotiation phase if necessary.

There is a technical challenge involved in this approach. Although current traders support dynamic values in service offers, the result of a trust decision depends on the actor the query is made on as well as context information, most importantly the network being built. This would require a populator implementation which supports embedding remote function calls with nontrivial parameters in service offers or network models.

The current populator can also be passed an utility function to use for ordering service providers. To apply trust values in this utility function, the remote call technology described above can be used to request trust values or categories for the actors. As an option, the trust utility value can be built on public information only, by using the trust-relevant certificates described earlier as a basis for scoring. Functions and weights for different certificate fields are defined by the initiator, and support for this kind of trust ordering is already present in the populator.

Finally, the populator allows the initiator to pass it a list of service offers of given partners to fill a particular role, or a blacklist of service providers to not fill the role with. This feature enables the initiator to make preemptive trust decisions, if it so wishes and has the information to do so, for example from earlier eCommunity instances.

We plan to continue by implementing trust decision making in the negotiation phase, described as the first option in this section. The approach is a strong combination of informed decisions and protecting the peers' privacy. In addition, these kinds of decisions should be possible to make also after the populator has done some pruning or ordering by itself.

## 5 Conclusion

This paper proposes an automated, generic method for selecting eCommunity participants with focus on the social and contractual aspects, especially external business processes, concept of utility, and trust on potential collaborators. The solution is based on multi-partner matching of service offers, guided by a jointly selected, public business network model. It thus extends the traditional trading or brokering architectures. The presented eContract structure pulls out publishable aspects of interoperability issues, still leaving some pragmatic aspects private.

In comparison to related work on eContracting solutions, the proposed method is unique in the way it captures all three aspects, social, contractual and technical, into an automated process where all functional and non-functional aspects of the collaboration are treated according to a few simple principles. The main design goal has been to separate interoperability and eCommunity management tasks into a B2B middleware layer, that is founded on metainformation repositories for business networks, business services and contractual rules. The solution is closely related to work on virtual enterprises and virtual enterprise breeding environments, but takes a more pragmatic view on the separation of generic B2B negotiation and eCollaboration management routines.

## Acknowledgement

This work has been performed at the Department of Computer Science at the University of Helsinki. The research group involved, Collaborative and Interoperable Computing group, builds up on work done in various projects funded by the national technology development center TEKES and industrial partners.

## References


[1] S. Angelov and P. Grefen. The 4W framework for B2B e-contracting. *Int. J. Networking and Virtual Organisation*, 1(3), 2003.

[2] D. Booth et al. *Web Services Architecture*. W3C Working Group. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

[3] D. K. W. Chiu, S. C. Cheung, P. C. K. Hung, S. Chiu, and K. Chung. Developing e-Negotiation Support with a contract template meta-modeling approach in a Web Services environment. *Special Issue on Web Services and Process Management in the Decision Support System (DSS)*, 40(1):51–69, July 2005. http://teaching.ust.hk/~csit600c/eNeg.pdf.

[4] A. Daskalopulu. Evidence based electronic contract performance monitoring. *The INFORMS Journal of Group Decision and Negotiation. Special Issue on Formal Modelling in E-Commerce*, 2002.

[5] C. Dellarocas and M. Klein. Designing robust, open electronic marketplaces of contract net agents. In *Proceedings of the 20th International Conference on Information Systems (ICIS)*, Charlotte, NC, Dec. 1999.

[6] F. Griffel, M. Boger, H. Weinreich, W. Lamersdorf, and M. Merz. Electronic contracting with COSMOS - how to establish, negotiate and execute electronic contracts on the Internet. In *Proceedings of Second International Enterprise Distributed Object Computing Workshop (EDOC '98)*, 1998.

[7] N. Griffiths and K.-M. Chao. Experience-based trust: Enabling effective resource selection in a grid environment. In *Proceedings of the iTrust 3rd International Conference on Trust Management*, pages 240–255. Springer-Verlag, LNCS 3477/2005.

[8] B. N. Grosof and T. Poon. SweetDeal: Representing agent contracts with exceptions using XML rules, ontologies and process descriptions. In *Proc. Intl. Conf. on the World Wide Web*, 2003.

[9] ISO/IEC JTC1. *Information Technology – Open Systems Interconnection, Data Management and Open Distributed Processing. Reference Model of Open Distributed Processing*, 1996. IS10746.

[10] L. Kutvonen and J. Metso. Services, contracts, policies and eCommunities – Relationship to ODP framework. In P. Linington, A. Tanaka, S. Tyndale-Biscoe, and A. Vallecillo, editors, *Workshop on ODP for Enterprise Computing (WODPEC 2005)*, pages 62–69, Sept. 2005.

[11] L. Kutvonen, J. Metso, and T. Ruokolainen. Inter-enterprise collaboration management in dynamic business networks. In *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE*, Agia Napa, Cyprus. Springer-Verlag, LNCS 3760/2005.

[12] L. Kutvonen, T. Ruokolainen, and J. Metso. Interoperability middleware for federated business services in web-Pilarcos. *International Journal of Enterprise Information Systems*, 2006.

[13] P. F. Linington, Z. Milosevic, J. Cole, S. Gibson, S. Kulkarni, and S. Neal. A unified behavioural model and a contract language for extended enterprise. *Data Knowledge and Engineering Journal*, 51(1):5–29, Oct. 2004.

[14] J. Metso and L. Kutvonen. Managing Virtual Organizations with Contracts. In *Workshop on Contract Architectures and Languages (CoALa2005)*, Enschede, The Netherlands, Sept. 2005.

[15] OASIS. *UDDI Registry - Technical Specification*, Feb. 2006. http://uddi.org/pubs/uddi_v3.htm.

[16] OASIS ebXML Collaboration Protocol Profile and Agreement Technical Committee. *Collaboration-Protocol Profile and Agreement Specification*. OASIS, Sept. 2002. http://www.oasis-open.org/committees/ebxml-cppa/documents/ebcpp-2_0.pdf.

[17] Object Management Group (OMG). *CORBA Trading Service*, 2002. http://www.omg.org/cgi-bin/doc?formal/2000-06-27.

[18] M. P. Papazoglou and D. Georgakopoulos. Introduction. *Communications of the ACM*, 46(10):24–28, 2003.

[19] I. Ponka. Populaattori. Technical report, University of Helsinki, 2004. Internal report, in Finnish.

[20] L. Rasmusson and S. Jansson. Simulated social control for secure Internet commerce. In *Proceedings of the 1996 workshop on New Security Paradigms*, pages 18–25. ACM Press, 1996.

[21] S. Ruohomaa and L. Kutvonen. Trust management survey. In *Proceedings of the iTrust 3rd International Conference on Trust Management*, pages 77–92. Springer-Verlag, LNCS 3477/2005, May 2005.

[22] S. Ruohomaa, L. Viljanen, and L. Kutvonen. Guarding enterprise collaborations with trust decisions—the TuBE approach. In *Proceedings of the First International Workshop on Interoperability Solutions to Trust, Security, Policies and QoS for Enhanced Enterprise Systems (IS-TSPQ 2006)*, Mar. 2006. To appear.

[23] T. Ruokolainen and L. Kutvonen. Interoperability in Service-Based Communities. In C. Bussler and A. Haller, editors, *Business Process Management Workshops: BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS*, pages 317–328. Springer-Verlag, LNCS 3812/2006.

[24] T. Ruokolainen and L. Kutvonen. Service typing in Collaborative Systems. In *Interoperability for Enterprise Software and Applications Conference (I-ESA2006)*. Springer Verlag, 2006.

[25] M. Schoop, A. Jertila, and T. List. Negoisst: A negotiation support system for electronic business-to-business negotiations in e-commerce. *Data and Knowledge Engineering*, 47(3):371–401, 2003.

[26] M. P. Singh and M. N. Huhns. *Service-Oriented Computing: Semantic, Processes, Agents*. John Wiley & Sons, Ltd., 2005.

[27] S. Thatte et al. *Business Process Execution Language for Web Services*. BEA Systems, IBM, Microsoft, SAP AG, and Siebel Systems. ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf.

[28] L. Xu and M. A. Jeusfeld. Pro-active monitoring of electronic contracts. In *Proc. CAiSE 03*, number 2681 in Lecture Notes in Computer Science. Springer-Verlag, 2003.