Inter-enterprise Collaboration Management in Dynamic Business Networks

Lea Kutvonen, Janne Metso, and Toni Ruokolainen

Department of Computer Science, University of Helsinki, Finland
{Lea.Kutvonen, Janne.Metso, Toni.Ruokolainen}@cs.Helsinki.FI

Abstract. The agility to collaborate in several business networks has become essential for the success of enterprises. The dynamic nature of collaborations and the autonomy of enterprises creates new challenges for the operational computing environment. This paper describes the web-Pilarcos B2B middleware solutions for managing the life-cycle of dynamic business networks in an inter-enterprise environment. The use of B2B middleware moves the management challenges away from the individual enterprise applications to more global infrastructure services, and provides a level of automation into the establishment and maintenance. The middleware services aim for a rigorous level of transparent interoperability support, including awareness of collaboration processes, and collaboration level adaptation to breaches in operation.

Keywords: E-services architectures and technologi; Inter-enterprise eCommunity management, interoperability; Corss-organisational process support, contracts.

1 Introduction

The present, rapid globalization of business makes enterprises increasingly dependent on their cooperation partners; competition takes place between supply chains and networks of enterprises. The level of dynamic integration capabilities between independent enterprise ICT systems is critical for the success of such business networks. Enterprise ICT systems are expected to participate into several, potentially heterogeneous networks simultaneously. They should also be able to react fast to changing partnerships, and use technology-independent tools for managing technical and semantical interoperability.

Traditional inter-enterprise integration solutions are typically based on tightly coupled application level integration (EAI) or they rely on some common meta-model to generate interoperable business applications. The use of these integrated or unified collaboration models usually guarantees the correct operation of inter-enterprise communities as all the needed interoperability information is implicitly contained in the resulting inter-enterprise applications. However, interoperability is achieved at the expense of autonomy, reusability and flexibility of business services and networks.

Possibilities for service reuse and evolution, as well as business network adaptivity, can be enhanced by the use of a federated collaboration model. The federated model builds collaboration relationships between already existing services, based on their interoperable functionalities.

The federated approach needs a platform with facilities for inter-enterprise network management and for making the interoperability information explicitly available during community operation. A strategic breeding environment for new business networks is needed with facilities for deciding on the shared business process, and roles of partners within it; selection of component services from the partners' IT systems; ensuring and enforcing interoperability between the component services; and establishing the business network. An operational environment for maintaining and controlling the business network is needed with facilities for joining and leaving the network; automated monitoring of the behaviour of the network and intelligent methods for adapting to technological changes and heterogeneity in the processing environment; and adapting to collaboration changes in terms of network membership and breach management.

The web-Pilarcos project aims for a decrease in the cost of establishing and operating electronic business networks, especially in the cost involved into changes of the business processes, partnerships, application services, and platform technologies. The main investment must be placed on the right kind of middleware that is able to use metainformation on the changeable elements for governing the overall collaborations. As the web-Pilarcos middleware is directed for enterprises participating in multiple, heterogeneous business networks where gradual evolution is to be expected, we call it B2B middleware. It forms a loosely-coupled collaboration layer on top of distributed, service oriented middleware.

The web-Pilarcos architecture uses meta-level information – such as business process model and service descriptions of the participants – that via reflection mechanisms governs the business network operation. The meta-level information can be renegotiated and changed, and these contractual changes are automatically reflected to the underlying computing system configuration. Likewise, automated mechanisms are build to observe the underlying system status, and reflecting that back to the status of the meta-information.

The web-Pilarcos architecture represents an approaches where meta-level contracts are used for inter-enterprise collaboration management. In contrast to most other architectures, web-Pilarcos does not use the metamodels for executing the collaborative workflow, but to check the potential for process-level and pragmatic interoperability and to monitor conformance to the agreed collaboration model. When interoperability is achievable, the collaboration establishment phase is able to automatically configure some adaptors to the runtime environment; when operational-time breaches are detected, resolution processes can be automatically initiated across the collaboration. This approach is more cost-effective in terms of tolerating changes in local and collaborative business processes, provided services, and platforms.

This paper describes the web-Pilarcos B2B middleware solutions for managing the life-cycle of dynamic business networks in an inter-enterprise environment. Section 2 introduces the B2B middleware services and eCommunity contracts. While Section 3 briefly addresses the role of breeding environment, Section 4 elaborates on the operational time services. Section 5 discusses the prototype implementation. Related work and future development issues conclude.

2 The B2B Middleware Services

To facilitate joint management of collaborations in the web-Pilarcos architecture, inter-enterprise collaborations are modelled as eCommunities that comprise of independently developed business services. The inter-enterprise collaboration management is supported by concepts of

– an eCommunity that represents a specific collaboration, its operation, agreements and state; the eCommunities carry identities and are managed according to their eCommunity contract information; and
– services that are provided by enterprises, used as members in eCommunities, and are made publicly available by exporting service offers.

These concepts are used by a set of B2B middleware services for establishing, modifying, monitoring, and terminating eCommunities. Looking from the application service point of view, operations are made available for joining and leaving an eCommunity either voluntarily or by community decision.

For the eCommunity management, interoperability is a fundamental issue. Interoperability, or capability to collaborate, means effective capability of mutual communication of information, proposals and commitments, requests and results. Interoperability covers technical, semantic, and pragmatic interoperability. Technical interoperability means that messages can be transported from one participant to another. Semantic interoperability means that the message content becomes understood in the same way by the senders and the receivers. This may require transformations of information representation or messaging sequences. Finally, pragmatic interoperability captures the willingness of partners for the actions necessary for the collaboration. The willingness to participate involves both capability of performing a requested action, and policies dictating whether the potential action is preferable for the enterprise to be involved in. In the pragmatic view, process-awareness in terms of collaborative business process model is needed, augmented with nonfunctional aspects, some of which are related to business policies.

The interoperability challenges are addressed both by the breeding facilities and operational environment. The breeding environment supports establishment of eCommunities in such a way that the open markets of business services is exploited, but at the same time the strategic and pragmatic restrictions of involved enterprises are taken into consideration. The operational environment is responsible of observing the behaviour of the participants in the eCommunity, react to breaches of the eCommunity contract, and to respond to eCommunity administrator or participant requests (human intervention) on renegotiation of some contract aspect. The breeding and operational environments are not isolated from each other, as for example joining new members to an existing eCommunity involves services of the breeding environment. The metainformation services and their use for interoperability ensurance is briefly summarized in Section 3 (a more thorough discussion is given in [1,2]).

- *reference to the business network model;*
- *information about the epoch in which the network is;*
- *process for changing epoch;*
- *for each role*
  - *assignment rules that specify the requirements on*
    - *service type;*
    - *nonfunctional aspects;*
    - *restrictions on identity, participation on other business networks, etc;*
  - *conformance rules that are used for determining conformance to the role which the assigned component is in the role; similar as above;*
- *for each interaction relationship between roles*
  - *channel requirements*
  - *locations of the channel endpoints*
  - *QoS agreement; security agreement*
  - *information presentation formats*
- *for each policy that governs the choices between alternative behaviour patterns in the business network model*
  - *acceptable values or value ranges;*
- *references to alternative breach recovery processes;*
- *objective of the business network as business rules*

Fig. 1. Information contents of the eContract

The operational environment supports the metalevel model of eCommunity by maintaining (distributed) eContracts. The semantical contents of the eContracts is summarized in Fig. 1. The eContract captures information about the agreed business network model, the participants with their locations and service access points, rules for accepting partners into the network, rules for monitoring whether existing partners can be accepted to continue in the network, and agreed collaborative process models for breach recovery situations. For each communication relationship, the eContract also captures the requirements for the abstract communication channel needed; this channel is then further mapped to suitable distribution platform services. The eContracts can be changed by negotiation between agents representing partners (enterprise) of the eCommunity. These agents do not themselves provide the involved services, but reside at the B2B middleware level, and act based on business-rules and process-models defined for the application service in question and notifications reporting the progress and the failures of the collaboration in question. The agents can control the local operating environment through the local service management facilities. For interoperability, the shared metalevel notations in the eContract are transformed to the locally understood management data and methods.

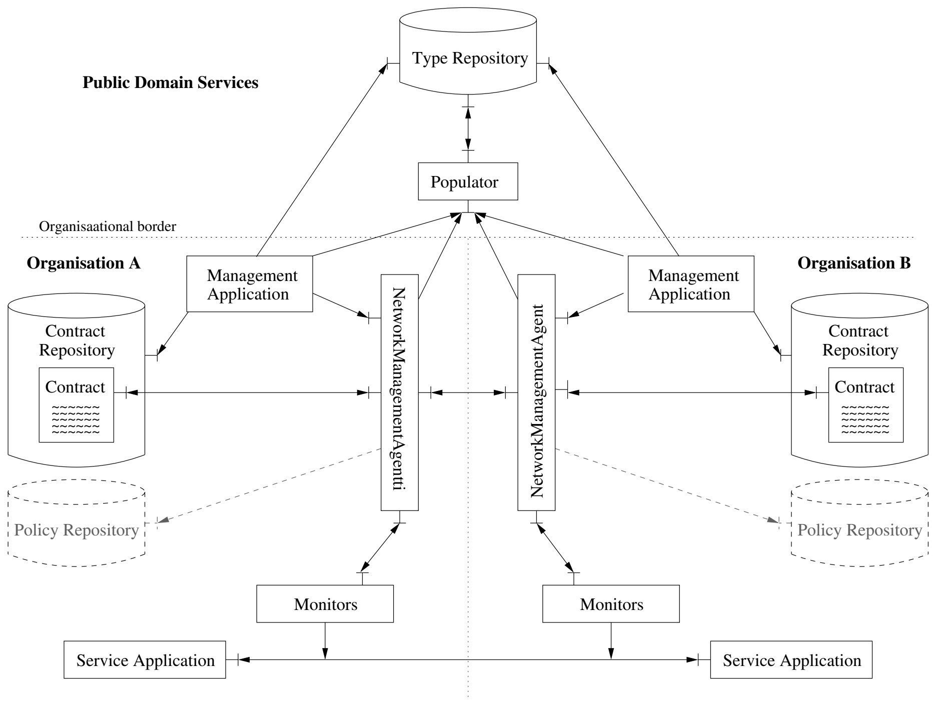

Fig. 2. Service agents of the operational environment

Fig. 2 illustrates the B2B middleware service agents of the operational environment. Each site or administrative domain, representing an autonomous ICT system, is expected to run a business process management agent. By autonomy we mean the potential for control over the private computing systems, and moreover on strategic business processes and policies. Breeding environment services like populators and type repositories are not required from all sites, but can be provided as infrastructure services as a business on its own right.

3 Breeding Environment

Establishment and maintenance of eCommunities relies on the interoperability knowledge on business network models (BNM), service types and associated information, and service offers.

The business network models are defined in terms of roles and interactions between the roles. For each role, assignment rules define additional requirements for the service offer that can be accepted to fulfill it, and conformance rules determine limits for acceptable behaviour during the eCommunity operation. Thus the business network model defines the structure and behaviour of the collaborating community. A verified business network model acts as a template for the eCommunity.

The model to be used as a contract template is first negotiated between the potential partners, involving comparison and matching of strategical, pragmatical goals of members in the network. The matching of network models is too hard a problem to solve by an automated process in general cases for a heterogeneous modeling environment. Therefore, we have focused on practical goals: What is needed is a grouping of similar models, where there is suitable transformers or adapters available for configuring a communication channel between peers so that the information exchange becomes understood correctly and there is no known deadlock in the sequence of message exchanges. The adapters can address modifications at multiple levels of interoperability, such as data representation modifications, and changing the communication pattern (for example, splitting a request of a task to a set of requests for subtasks from the peer). The service type repository is used for holding such relationships between models and the transformation information associated. The actual adapters are produced in a separate process starting from the service type descriptions [3].

Potential participants for the defined roles are retrieved from the service offer repository based on their service type and published properties that are associated for that type. Service type information captures syntactic and procedural interoperability; semantics over behaviour is considered too hard yet [4]. On one hand, the business network models associate roles with required service types; on the other hand, the service offer repository associates the service with the service types. The quality of these assertions is essential for the correctness of the created communities. Furthermore, the density of the created relationship network of alternative but interoperable service types determines the usefulness of the middleware service [5].

The resulting eCommunity contract object is an active agent itself, and provides a service interface with operations for initiating re-negotiations, and receiving progress reports by participants. It also takes initiative in message exchanges with local service management agents at each site involved.

The repositories that maintain a growing and increasingly interrelated set of interoperability knowledge are feed by independent processes: publication of a) service types and b) business network models, c) service offers, and d) eCommunity population and negotiation processes. These processes are inter-related but not tightly dependent; for example new service types can be published without a business network model using them. Fig. 3 illustrates these processes. The service publication functionality is similar to the UDDI [6] or the ODP trading mechanisms [7]; while the type management system resembles the ODP type repository function [8] and enforces a typing discipline to follow over service offer repositories. The BNM repository is a shared storage of business collaboration information that enable enterprises to share business transaction models, such as the ebXML-repository [9], although with more automated and repeatable breeding process. The used notations are not discussed here, but they resemble ODP enterprise language and use XML-style notations (see [10] and [2]).

In the service publication process (step 1), service providers send service offers to the service offer repository, to state claims about the type and properties of the services. A service offer describes functional and non-functional properties of the service to be published: the actual service interface signature, service behaviour, requirements for technical bindings (e.g. transport protocol), and attributes such as service quality and trust related commitments. The service offer repository then initiates a conformance validation process. For this purpose, a service type corresponding the claimed service type is retrieved from the type repository (step 2). The service type defines syntactical structures for service interface signature and messages, externally visible service behaviour and semantics for exchanged messages [2]. Conformance validation is executed by the service type repository holding the corresponding service type (step 3). Only after a successful validation, the service offer is published (step 4a), otherwise a service typing mismatch is reported between the service offer and its claimed type (step 4b).

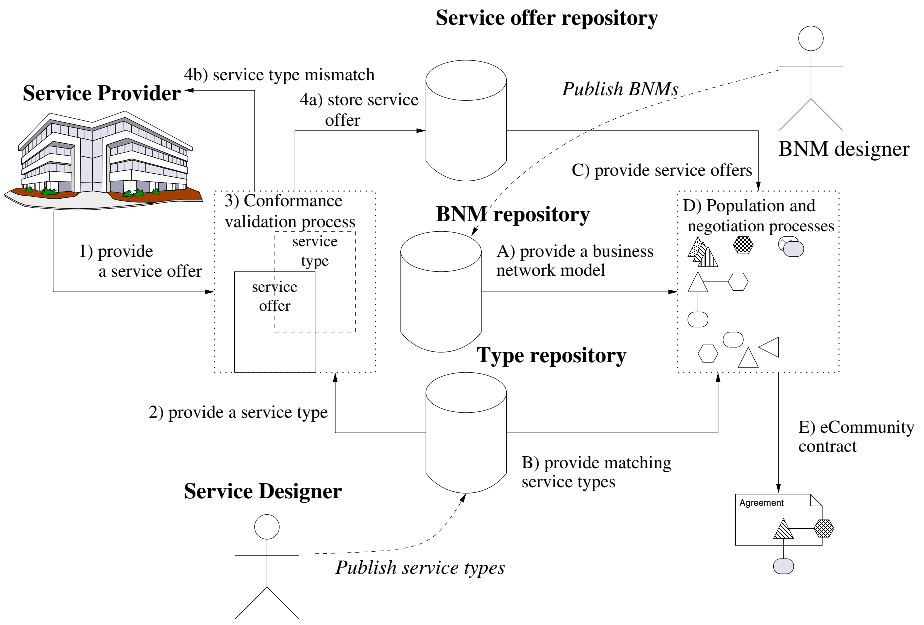
Service offer repository
4b) service type mismatch
Service Provider
4a) store service offer
Publish BNMs
BNM designer
C) provide service offers
3) Conformance validation process
BNM repository
D) Population and negotiation processes
1) provide a service offer
service type
service offer
A) provide a business network model
Type repository
2) provide a service type
E) eCommunity contract
B) provide matching service types
Service Designer
Agreement
Publish service types

Fig. 3. Repository usage during eCommunity life-cycle

When an eCommunity establishment process is initiated by a willing partner, the corresponding business network model is first fetched from the BNM repository (step A). The population process (step D) provides a set of interoperable eCommunity proposals where roles of the BNM are filled with potential partners. For this purpose, the type repository is consulted for providing service types matching the requirements of the business network model (step B), after which the service offer repository can be used to provide the corresponding service providers (step C). After population, and the subsequent negotiation, the eCommunity contract is received (step E) and distributed to every participant.

The service interoperability and correct operation of the community assumes that the metalevel information on BNMs, service types, and service offers is correct. Therefore, we find it necessary to collect the metainformation into repositories, where the trustworthiness of the information source can be controlled, and the quality of the information can be validated by the repository management actions. These aspects must be weaved into the tasks involved with eCommunity establishment, such as service publication or discovery [11, 5].

4 Operational-Time Environment

The operational-time environment comprises of the business process management agents maintaining the metalevel agreement of the collaboration, the local service management facilities at each enterprise, the monitoring embedded to each communication channel in each business network, and the business models used for resolving collaboration-level exception states. Section 5 follows with implementation detail.

4.1 eCommunity Management by Collaborative Agents

The community life-cycle includes steps for establishment (population and negotiation at the conceptual level, establishment of the community at the technical level), termination, reacting to change requests, and resolution of breaches. The eCommunity life-cycle is presented in Fig. 4.

The state transitions are performed by middleware agents. The eCommunity is placed into the initial state (populated) by the populator agent in the breeding environment. For other state transitions the responsibility is on the agents in the operational environment, Business Network Management Agents (BNMA, agent) and the eCommunity contract object, in collaboration. At each administrative domain, there is a BNMA agent responsible for managing the inter-organizational coordination and management protocols, global state information management, community participant management and contract breach management. The contract object is responsible for making decisions for the community it represents (currently, most decisions are referred to humans).

In the populated state, the BNMAs see a set of potential eCommunity contracts where the partners have a matching view of the business network model, a non-empty set of options for policy values representing communication and information representation aspects of the collaboration, and a matching set of requirements for platform services. The first contract draft is available to the initiating partner only, while in the in-negotiation state the suggested eContract is under the consideration of all participants. During the negotiations, the eContract can gain further decisions on joint policies and technologies in use, as described below. The technical establishment phase involves the local service management facilities. When unwanted situations are detected by the monitoring system and BNMAs agree that the case is a major fault, a reorganization process is started, potentially causing changes in the partnership. In addition, the business network model involved may include epochs; an epoch change captures a major reorganization of the collaboration structure.

The negotiations are implemented as a coordinator-driven n-to-n negotiation. The coordinator is elected during the first negotiation round among the participant candidates. At each negotiation round the whole contract is sent to all candidates for consideration of the terms of the contract, with responses of agreement, disagreement, and possibly counter-offers. The coordinator merges the requested changes and proceeds to another negotiation round until all participant candidates agree on the contract terms or terminates the negotiation in lack of agreement. When any of the candidates refuse to participate the negotiations, the suggested eCommunity is moved back to population state.

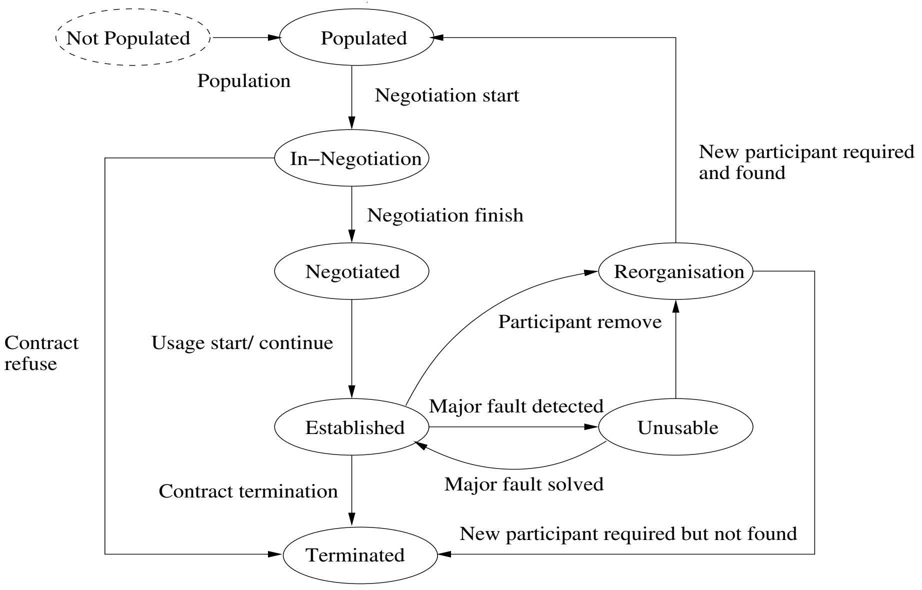

Fig. 4. Life cycle model of the community

The negotiations involve two categories of decisions. First, the business network models can describe alternatives for joint behaviour, and a policy decision needs to be made which alternative is used (for example, whether services must be prepaid or not). Second, some technical details such as information representation formats need to be agreed on. However, all business rules directing the local behaviour at each enterprise are not negotiated, only those defined in the business network model. To illustrate a common negotiation process, we assume there are three possible technology solutions for communication channels in a proposed eCommunity. The partner in the coordinator role sends the three-way proposal of the eCommunity contract to other participants. The participants responds with counter-offers containing the acceptable choices of technology to them. As the counter-offers give different sets, it is the coordinator's task to find a cut of the sets and make the final decision within that set.

After a successful negotiation cycle, the contract is established by sending an establish request to all participants of the eCommunity. At this signal the provided services are prepared and monitors are configured with contextual metadata derived from the contract. The participants respond to the coordinator when their eCommunity elements are set up.

The eCommunity contract is a distributed object. Only the metainformation contents of the contract is distributed (because of heterogeneous technologies), and the distribution is embedded to the negotiation protocol of eCommunity contract. The local contract copies are kept in loose synchrony through BNMAs.

The messages for state management and breach notifications form a basis for a reflective mechanism that keeps the meta-level information in the eCommunity contract and the actual service provision at the involved sites in synchrony. These messages cause changes in the contract contents, assuming that the change request are not contradictory. Changes in meta-data trigger activities for checking whether community participants need to be notified or requested a local act.

Each site has a local service management agent that holds and uses knowledge about locally deployed services and their various management methods [1]. The local management interfaces are homogenized by a protocol for requesting the system to prepare for running a service (resourcing), querying about communication points, and releasing the service. For local service-management services we propose to use generic service factories so that the actual service platform is irrelevant to the agent. Service factory can then be called with simple operations like *startService* to start a service or *stopService* to stop a service when it is not needed. Error reports allow the local management services to determine local and remote failures and to adhere to the agreed behaviour. This information can be used to improve performance of the observing domain and to file error reports to other domains in the community.

The local service management interface allows BNMAs o collaboratively manage the community consistency. For example, initiative to replace or move a participant of the eCommunity causes requests to change the service point to the new participant's location, and recreate the bindings between new locations.

Besides indirect management by BNMAs, the local services are controlled by local enterprise policies (i.e., business rules). Each enterprise is expected to have a private policy repository that captures rules for accessing services and distributing documents. These resource guards can be implemented in a similar style as has been presented in other policy-based management work considering also deontic policies [12, 13]. Each resource (processing unit, document) is governed by a monitor that consults the local policy repository for permission to proceed with a requested interaction. The local policies may change during the operational time of an eCommunity, and the local policies may override all community commitments. This may lead to policy conflicts during the eCommunity operation. Although the conflict styles identified are similar to static analysis approaches (see for example [14]), we start dynamic conflict resolution by situations triggered by the monitoring system. A prototype implementation addresses problems of mismatch between organizations on access permissions, prohibitions, and obligations [15].

Communities can be terminated in a natural or forced way. A natural termination takes place if the contract is expired or the specified amount of sessions in the contract is exceeded. Contract session is specified as a one execution of the community functionality as described in the contract. The contract termination is forced if it is a consequence of a resolution process.

At any time, a participant may request that another participant in the community is removed, or inform others that it withdraws itself. The community contract defines the compensation process for such an event. The request naturally causes a negotiation cycle amongst participants. After the participant has left the eCommunity, the remaining participants will hold an election, lead by the coordinator, to decide if they will find a new participant to fill the now vacant role or terminate the community.

4.2 Context-Aware Monitoring

Monitoring is performed locally by each participant of a community, at the communication channel end-points. The monitors continuously evaluate whether observed behaviour is conformant to the expected behaviour explicated in the eContract. For example internal policies of organizations, service evolution and technical failures are causes for dynamic errors only detectable by active runtime monitoring. The monitors report progress of local business processes and detected breaches to local BNMAs; if needed, the BNMAs negotiate about required corrective actions.

Monitors are configured by BNMAs with context information and related rules retrieved from the eContract. The eContract carries information on current progress state of the collaborative business process, and requirements for the correct progress of collaboration (service choreography), as well as process models for exceptional situations. The context relevant for the monitor represents an active part of the progression: the expected choreography, and freely designable monitoring criteria.

A monitor follows the behaviour of a service against the service choreography (external business process) represented by a two level state-machine, where the upper level represents task groups, and the lower level represents interrelated messages within that group. The upper-level machine is used to provide a coarser view to the progress through the choreography [16]. The task model is quite close to the work unit model in WS-CDL [17]; however, the model is not used for execution but for observing conformance. The state machine notifies completion of a task once all expected messages in the task are exchanged in an acceptable order. Problem notifications can be raised for order breaches, missing messages, and information contents. The monitoring facilities use the lower level only internally and reports to local BNMA using the task level. Reports from the monitors include meta-events when a specific task is completed and when behaviour of the services are not correct.

The steps taken when a message is sent between two service applications are shown in Fig. 5. The illustration also indicates the intercepting location of the monitors in the communication channel architecture. In the figure, the box WS-Tr. represents additional services in the channel, such as aspects of distribution transparency and transaction support (for more details, see [5]).

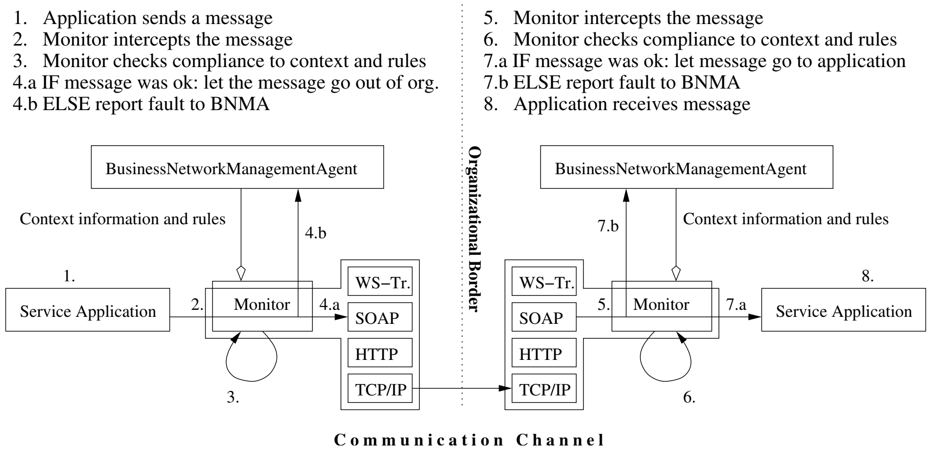
1. Application sends a message
2. Monitor intercepts the message
3. Monitor checks compliance to context and rules
4.a IF message was ok: let the message go out of org.
4.b ELSE report fault to BNMA
5. Monitor intercepts the message
6. Monitor checks compliance to context and rules
7.a IF message was ok: let message go to application
7.b ELSE report fault to BNMA
8. Application receives message
BusinessNetworkManagementAgent
Context information and rules
4.b
7.b
Organizational Border
Service Application
Monitor
WS–Tr.
SOAP
HTTP
TCP/IP
1.
2.
3.
4.a
5.
6.
7.a
8.
Communication Channel

Fig. 5. Monitor as a part of the communication channel during a message send

The task level state information is not aggressively distributed to all participants, but relayed to the contract object to be retrieved through it by those participants needing it, when needed. Still, the level of traffic generated may cause undue overhead unless business network structure is reasonably designed. The task boundaries are annotated on the choreographies by the designers, and the analysis of the models should therefore include also the cost of the model.

In the monitoring criteria, it is possible to use rules that consider the business network status as well, for catching behaviour rules such as "payment must be received by the bank before the warehouse can ship the delivery". The monitors can also control aspects of information representation, trustworthyness of the service requests, and other nonfunctional aspects of the collaboration.

4.3 Breach Management and Epochs

Breach management is triggered by the monitors by notifying their local BN-MAs both of minor and major discrepancies. The BNMA decides whether the discrepancy is to be considered a breach, or can be passed with local recovery actions or by ignoring the occurrence. When a breach is detected, the detecting BMNA notifies the eCommunity coordinator with information on the event and the participant considered responsible of the failure. For the resolution of a serious breach, the eCommunity enters an intermediate state during which decisions are taken (potentially negotiated) on the corrective actions.

Architecturally, the recovery scenarios should not be fixed into the agents, but need to be derived from the business network model repository and become part of the eContracts in the eCommunity establishment phase. The recovery processes are one of the elements to be either matched in population or negotiated thereafter. However, in the current prototype, the offender can either admit or deny the breach. For admitted breaches, the compensation processes agreed in the eCommunity contract are used, and the activities of the eCommunity can be continued normally. For denied breaches, human intervention is used, and potentially leads to change of the faulty participant. The agent provides a method *changeParticipant* that invokes a negotiation on whether the offender needs to be excluded from the eCommunity, and subsequently involves the populator to assist in selection and interoperability assurance for a new participant. Alternatively, the breach can lead to total termination of the eCommunity after negotiations.

The transition to the separate resolution process requires that the involved service providers are prepared to run additional, infrastructure-level processes in addition to the original business process. The enterprises are expected to provide business facilities able to respond to for example sanction negotiations, thus conforming to a best-practices expectation. In addition, at the middleware level, facilities for epoch management are required.

An epoch is defined as a period during which roles and services of the network participants are stable. Two subsequent epochs can have different sets of roles and services involved, and between epochs transition rules can be defined. Participants in certain roles are required to reappear in a specific role in the next epoch, while some others leave the community. Transition between two epochs require synchronization between partners.

5 Prototypes and Lessons Learned

The web-Pilarcos middleware prototype is implemented in Java, with a mix of J2EE technology and standard Java objects. The prototype is built using JBoss and services are distributed as Web Services. The organization-oriented middleware services (Contract object, Contract Repository, NetworkManagementAgent, and Monitors) are implemented with J2EE; the public domain services (Type Repository, Populator, Service Offer Repository, and BPM Repository) are implemented as standard Java objects. All components, except the eContracts residing in the Contract repository, are accessible through Web Services interfaces, either within an enterprise, or across enterprise boundaries. The most important components of the implementation include the eContract, the BNMAs, and the monitors. The set of prototype services also includes an application for visualizing the inter-enterprise business process and its progress [18]. This application provides human access to the partner change and eContract renegotiation methods through BNMAs.

The NetworkManagementAgent component implements BNMA interfaces for eContract life-cycle management, negotiation, establishment, global state man-

```
Lifecycle management interface:
    String[] populateArchitecture(String architectureName, String myRoleName,
                                  int maxOffers, int maxTime,
                                  String usedPolicies)
    int[] negotiateContract(String contractID)
    int[] instantiateContract(String contractID)
    void terminateContract(String contractID)
    String createNewSession(String contractID)

Negotiation and establishment interfaces:
    acceptContract(ContractContent contract, String participant)
    acceptContractResponse(String contract_id, String participant,
                           NegotiationResponse negotiation_response)
    establishContract(String contractID, String participant)
    establishContractResponse(String contractID, String participant,
                              boolean success)
    renegotiateContract(String contractID)
    renegotiateContractResponse(String contractID, String participant,
                                boolean renegotiate)
    announceResult(String contractID, boolean renegotiate)
    cancelContract(String contractID, String participant)

Global state management interface:
    updateTaskState(String contractID, String sessionID, String taskID,
                    String newState, String participant)
    epochChanged(String contractID, String sessionID, String newEpoch,
                 String participant)

Monitor input interface:
    updateEpochState(String sessionID, String roleID,
                     String epochID, String stateID)
    sessionEpochFinished(String sessionID, String epochID)

Monitor configuration interface:
    addSession(String sessionID, String[] epochIDs, String[] policyIDs,
               String[] myRoleIDs, String[] otherRoleIDs)
    addRole(String sessionID, String roleID, String[] roleEpochIDs,
            String[] policyIDs)
    addEpochAutomata(String sessionID, String roleID, String epochID,
                     StateAutomata epochAutomata)
    addChoreographyAutomata(String sessionID, String roleID, String epochID,

    setActiveSessionEpoch(String sessionID, String epochID)
    deactivateSession(String sessionID)
    isSessionActive(String sessionID)

    deactivateRole(String sessionID, String roleID)
    isRoleActive(String sessionID, String roleID)

    updateEpochState(String sessionID, String roleID,
                     String epochID, String stateID)
```

Fig. 6. Interfaces of middleware services

agement, and monitor input. The first three interfaces are used between enterprises, the rest by local services. The BNMA interfaces are described in Fig. 6.

The BNMAs form an agent-style discussion amongst themselves: the initiator suggests a collaboration using a named model for a group of named partners, and the group members make counter-offers to the suggested details. The propositions are taken as believable facts (we have trust-management extensions planned, which change this). As an extension to the traditional agent discussions, the BNMAs are able to detect breaches to the agreed behaviour, and start negotiations on the caused situation. The BNMAs are not self-contained as agents, as initiatives to actions are received by users, applications, and changes in the local computing services.

The monitors are hooked into the communication channel architecture, as proxies into the JBoss environment to intercept messages. In addition, monitors implement a service interface for metadata configuration. The monitor interfaces are described in Fig. 6.

The scalability of the architecture has been carefully analyzed, mainly resulting to aspects that need to be verified in the business network models used. As a consequence, the analysis gives us guidelines for providing new software engineering tools in the area of model verification and model property analysis.

Across enterprises, communication is restricted to global business network state updates, error resolving, and contract life cycle management. State updates are done only when epochs are changed and tasks completed, so the communication volume depends on the business network design. In the overall service and model production methodology behind our work, it is essential that the business network models are carefully verified and analyzed before publication. One of the essential features to analyze is the cost of the operation of the model. We expect that the task/epoch ratio is kept relatively low.

Within enterprises, the cost of communication is somewhat lower, especially if critical components are appropriately deployed. Monitoring cost is a scalability challenge, but can be partially overcome by suitable selection of monitoring modes: only proactive monitoring that prevents further steps in the business process interferes severely with the overall performance, and should be restricted to carefully selected features. Loose feedback loops from monitors to BNMAs can also be used and still acquire an operational-time detection of frauds and failures in collaborations.

As the population process is essential for the feasibility of the presented architecture, the first phase prototype included only the population process [19, 20], and performance evaluation on that. Having restricted the complex constraint satisfaction problem appropriately, we found the performance mainly dependent on the number of roles in the network and policies per role [21].

6 Conclusion

The B2B middleware developed in web-Pilarcos provides support for autonomously administered peer services that collaborate in a loosely coupled eCommunity. The eCommunity management by design excludes the need for distributed enactment services, but in contrast provides facilities for ensuring interoperability at semantic and pragmatic level. In this respect the federated approach has a different focus from those in most other P2P community management systems, such as ADEPT [22] or METEOR [23], and contract-driven integration approaches, such as ebXML [9]. Even most virtual enterprise support environments, such as CrossFlow [24] and WISE (workflow-based internet services) [25], rely on models for distributed business process enactment. However, the web-Pilarcos approach leaves enactment as a local business processing task, concentrating on interoperability monitoring.

The web-Pilarcos concept of eContracts ties together ICT related viewpoints of ODP (Open Distributed Processing reference model [26]), also ranging to some features of business aspects. The ODP-RM introduces information, computational, engineering and technical viewpoints. Each of these present interrelated but somewhat independent aspects of the collaboration features and its composition using more basic computing services. The web-Pilarcos contract structure captures these aspects in its BNMs, binding requirements, and behavioural and non-functional monitoring rules [10]. In other projects, like BCA [27], contracts have legal and business level focus and detect contract breaches post-operatively [28]. The web-Pilarcos aims for more real-time intervention.

In the web-Pilarcos middleware, the eCommunity life cycle is built to be collaboration-process-aware. The architecture model acts on two abstraction layers, the upper layer involved with abstract, external business process describing the collaboration requirements; the lower layer comprised of actual services bound to the eCommunity dynamically. In this kind of environment, static verification of models and interoperability cannot be complete. In the B2B middleware provided by the web-Pilarcos project, we find it necessary to develop control environments for monitoring and reflectively restructuring the operational eCommunities, besides a breeding environment. The goals are similar to other projects, but the solution methods differ. While ADEPT supports direct modification of the workflow control structures, web-Pilarcos uses negotiated policy-values to choose between predefined behaviour alternatives. The web-Pilarcos solution even requires that well-formed contracts include suitable recovery processes that involve whole communities. In contrast to METEOR-S, the web-Pilarcos platform has no central tool for making the whole of interoperability analysis, but partial static verification is done at the meta-data repositories, and monitoring is used to detect further problems.

The B2B middleware is in some extent comparable to agent-based approaches, such as MASSYVE [29]. The main difference seems to be the separation of business-application services and B2B middleware services from each other. The web-Pilarcos middleware agents do not provide workflow execution, but expect local application management to play that part. In contrast to [30], the middleware agents are responsible of semantic verification and failure resolution, and use separate monitors to help and report.

The web-Pilarcos middleware increases the ability of an enterprise to adapt to changes at strategical business processes, platform technologies, and partners and partners' services within the business networks. The presented middleware services indicate the essential B2B services on which to invest, in order to decrease the cost and reimplementation effort caused by changes in the operational environment. The operational environment of web-Pilarcos described in this paper enhances our earlier work on collaboration partner matching in the Pilarcos project [19] by introducing the monitoring of business processes and local enterprise policies, and by providing a set of eCommunity management protocols at the meta-information level.

The provision of the web-Pilarcos architecture requires further development of business process modeling techniques. The collaboration of business processes or workflows should be modeled without unnecessary revealing of local processing steps. Instead, only the collaborative part (external view) should be agreed on and monitored. Work is already started by the component-driven approach on splitting workflows into Web Services. The structural needs of business process models are also widened by the requirements of incorporating reusable sanctioning, recovery, and compensation processes into eCommunity contracts. Furthermore, shared ontologies and repositories for business process models should be made available. Such facilities would improve the potential for reaching interoperability in an environment where service components are truly developed independently from each other. More fundamentally, ontologies and repositories would create a facility for checking semantical similarity of business process model as part of the interoperability tests during eCommunity establishment.

Acknowledgment

This article is based on work performed in the Pilarcos and web-Pilarcos projects at the Department of Computer Science at the University of Helsinki. The Pilarcos project was funded by the National Technology Agency TEKES in Finland, Nokia, SysOpen and Tellabs. In web-Pilarcos, active partners have been VTT, Elisa and SysOpen. The work much integrates with RM-ODP standards work, and recently has found an interesting context in the INTEROP NoE collaboration.

References

1. Kutvonen, L.: Automated management of inter-organisational applications. In: Proc. 6th International Enterprise Distributed Object Computing Conference (EDOC2002). (2002)
2. Kutvonen, L., Ruokolainen, T., Metso, J., Haataja, J.: Interoperability middleware for federated enterprise applications in web-Pilarcos. In: Int. Conference on Interoperability of Enteprise Software and Applications (INTEROP-ESA'05), Springer Verlag (2005)
3. Kutvonen, L.: Relating MDA and inter-enterprise collaboration management. In Akehurst, D., ed.: Second European Workshop on Model Driven Architecture (MDA), University of Kent (2004) 84–88
4. Ruokolainen, T.: Component interoperability. Master's thesis, Department of Computer Science, University of Helsinki (2004) In Finnish.
5. Kutvonen, L.: Trading services in open distributed environments. PhD thesis, Department of Computer Science, University of Helsinki (1998)
6. OASIS Consortium: Universal Description, Discovery and Integration of Web Services (UDDI) 3. (2002) http://uddi.org/pubs/uddi_v3.htm.
7. ISO/IEC JTC1: Information Technology – Open Systems Interconnection, Data Management and Open Distributed Processing. ODP Trading Function. (1997) IS13235.

8. ISO/IEC JTC1: Information Technology – Open Systems Interconnection, Data Management and Open Distributed Processing. ODP Type Repository Function. (1999) IS14746.
9. Kotok, A., Webber, D.R.R.: ebXML: The New Global Standard for Doing Business Over the Internet. New Riders, Boston (2001)
10. Kutvonen, L.: Challenges for ODP-based infrastructure for managing dynamic B2B networks. In Vallecillo, A., Linington, P., Wood, B., eds.: Workshop on ODP for Enterprise Computing (WODPEC 2004). (2004) 57–64
11. Viljanen, L., Ruohomaa, S., Kutvonen, L.: The TuBE approach to trust management. In: Proceedings of the 3rd iTrust internal workshop. (2004)
12. Kollingbaum, M.J., Norman, T.J.: Supervised interaction: creating a web of trust for contracting agents in electronic environments. In: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, ACM Press (2002) 272–279
13. Lymberopoulos, L., Lupu, E., Sloman, M.: An adaptive policy based framework for network services management. Journal of Network and systems Management **11** (2003) 277–303 Special issue on Policy based management.
14. Dunlop, N., Indulska, J., Raymond, K.: Dynamic conflict detection in policy-based management systems. In: 6th International Enterprise Distributed Object Computing Conference (EDOC2002). (2002)
15. Karppinen, M.: Distributed policy enforcement. Master's thesis, Department of Computer Science, University of Helsinki (2003) In Finnish.
16. Haataja, J.: Monitoring of inter-enterprise collaboration networks in Web-Services environments. Master's thesis, Department of Computer Science, University of Helsinki (2005) In Finnish.
17. W3C: Web Services Choreography Description language. (2004) `http://www.w3.org/TR/2004/WD-ws-cdl-10-20041217/`, Working draft.
18. Henriksson, R., Kare, A., Lähde, M., Mäki, A.J., Stenberg, M., Virtanen, T.: Business network management GUI. http://www.cs.helsinki.fi/group/ohtu/s-2004/ltv.html (2004) Software engineering project.
19. Vähäaaho, M., Haataja, J.P., Metso, J., Suoranta, T., Silfver, E., Kutvonen, L.: Pilarcos prototype II. Technical Report C-2003-12, Department of Computer Science, University of Helsinki (2003)
20. Vähäaho, M.: Trading with architecture models. Master's thesis, University of Helsinki (2003) In Finnish.
21. Kutvonen, L., Metso, J.: Services, contracts, policies and eCommunities – Relationship to ODP framework. In: Workshop on ODP for Enterprise Computing (WODPEC 2005), IEEE Digital Library (2005)
22. Reichert, M., Dadam, P.: Adeptflex – supporting dynamic changes of workflow without losing control. Journal of Intelligent Information Systems **10** (1998) 93–129 Special Issue on Workflow Management.
23. Aggarwal, R., Verma, K., Miller, J., Milnor, W.: Contstraing Driven Web Service Composition in METEOR-S. In: Proceedings of the IEEE SCC. (2004)
24. Grefen, P., Aberer, K., Hoffner, Y., Ludwig, H.: CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises. International Journal of Computer Systmes Sciences and Engineering **15** (2000) 277–290
25. Lazcano, A., Alonso, G., Schuldt, H., Schuler, C.: The WISE approach to Electronic Commerce. Int. Journal of Computer Systems Science and Engineering (2000)
26. ISO/IEC JTC1: Information Technology – Open Systems Interconnection, Data Management and Open Distributed Processing. Reference Model of Open Distributed Processing. (1996) IS10746.

27. Milosevic, Z., Linington, P.F., S.Gibson, Kulkarni, S., J.Cole: Inter-organisational collaborations supported by e-contracts. In: The fourth IFIP conference on E-commerce, E-Business, E-Government, Toulouse, France (2004)
28. Quirchmayr, G., Milosevic, Z., Tagg, R., Cole, J., Kulkarni, S.: Establishment of virtual enterprise contracts. In: Database and Expert Systems Applications : 13th International Conference. Volume LNCS 2453., Springer-Verlag (2002) 236–
29. Rabelo, R., Camarinha-Matos, L.M., Vallejos, R.V.: Agent-based brokerage for virtual enterprise creation in the moulds industry. In: E-business and Virtual Enterprises. (2000) http://gsigma-grucon.ufsc.br/massyve.
30. Daskalopulu, A., Dimitrakos, T., Maibaum, T.: Evidence-based electronic contract performance monitoring. The INFORMS Journal of Group Decision and Negotiation (2002) Special Issue on Formal Modelling in E-Commerce.