Inter-enterprise business transaction management in open service ecosystems

Lea Kutvonen, Alex Norta, Sini Ruohomaa
Department of Computer Science, University of Helsinki, Finland
Email: {lea.kutvonen, alex.norta, sini.ruohomaa}@cs.helsinki.fi

***Abstract*—One of the difficult challenges in inter-enterprise computing is aligning business transactions and technical management of distributed transactions, especially in breach situations. We propose and analyse a two-level business transaction management framework that allows injection of business-level concerns to the control processes of inter-enterprise transactions. The two levels are associated with a) the metamodel of the collaboration, captured in eContract governing the collaboration, and b) the transactional interactions between collaboration member services. The two levels are bound together to form a reflective model; while business level breaches to the eContract can disturb the normal interactions, the metalevel possesses processes for managing (rolling back, compensating, ignoring, triggering ecosystem-level consequences) the failure in manners that align with the business incentives. This framework enables correctness, coherence and efficiency of processing in open inter-enterprise environments, i.e., service ecosystems.**

***Keywords*-networked business, inter-enterprise collaboration, business transaction management, open service ecosystems**

I. Introduction

Along with the increased expectations on networked business, enterprises face requirements on creating enterprise computing environments capable of addressing the collaboration needs. In our earlier work, we have introduced *open service ecosystems* as such an environment, providing a coherent architecture framework and facilities for shared infrastructure services [1], [2], [3], [4]. Two key concepts in open service ecosystems are those of

- *inter-enterprise collaboration*, which involves multiple partners through their software-based business services and their mutual interactions that are declared by a set of associated business processes (that we call a *business network model, BNM*); and
- *eContract*, which governs the inter-enterprise collaboration and captures both business and technical level aspects of control, as well the large-granule state information about the collaboration (the dynamic eContract agent provides the relevant context information for the transactions).

In this context, we have studied how to improve the efficiency and conceptual correctness of business transactions, and have found many of the solutions applicable to other business transaction frameworks as well.

One of the challenges in inter-enterprise computing is aligning business transactions and technical management of distributed transactions, especially in breach situations. The challenges arise from several sources. First, in the global business, each enterprise is autonomous, i.e. independent on making decisions on its business partners, software portfolio, and technology adoptions. This causes heterogeneity of partners in the business sense as well as in technical choices. Second, many transaction management systems provide solutions for managing breach situations caused by partners cancelling their activities or by technical failures to perform the activities. Thus, the partners in business transactions have limited ability to individually control and manage the transaction operation. Third, transaction management systems do not address the needs of being controlled by regulatory systems.

Therefore, we aim for an effective business transaction architecture in which autonomous partners control multi-partner business transactions according to their

- private business-level enterprise policies,
- private trust and privacy-preservation decisions,
- perceived nonfunctional properties of the business transaction in comparison to the collaboration contract,
- subjective choice of criteria for private decisions on accepting the state of the collaboration, and
- jointly-made collaboration-wide decisions on whether the transactions should be aborted and rolled back or just abandoned, recovered by predetermined forward recovery mechanisms, or should cause reorganisation of the whole collaboration contract.

The solution proposed and analysed in this paper is a two-level business transaction management framework that allows injection of business-level concerns to the control processes of inter-enterprise transactions. The two levels presented are associated with a) the metalevel model of the collaboration captured in the eContract governing the collaboration, and b) the transactional interaction level between member services in the collaboration. The two levels are bound together to form a reflective model: while business-level breaches of the eContract can disturb normal interactions, the metalevel holds processes for managing (rolling back, compensating, ignoring, triggering ecosystem-level consequences) the failure in manners that align with the business incentives. This framework enables correctness, coherence and efficiency of processing in open, inter-enterprise environments.

The remainder of this paper approaches the research question of how to bring business control to transaction management in three steps. Section II provides an overview of the reflective model and control processes involved, showing what kind of decision-points participating enterprises have, and how open service ecosystem founding services can be utilised. As the research question has two distinct parts – control processes and decision-making to choose from the alternative control paths – both of these elements need their appropriate research methods and considerations of related work. Section III takes the discussion to a more technical-level by discussing the breach management sequences in the Pilarcos collaboration management lifecycles. This provides the control framework for business transaction processing; the framework has been formally modelled as Coloured Petri Nets (CPN), which exercise shows that the protocol is alive, deadlock-free, and feasible to implement. While the process in Section III provides alternative behaviour routes for different business situations, Section IV discusses the mechanisms available for decision-making in enterprises in these situations. It also discusses the available methods for triggering strategy or trust consequences at the ecosystem level. The paper is concluded with some contrasting reflections on related work and a summary of the contributions.

II. Injecting business concerns to business transaction management

In the present inter-enterprise computing era, business transactions take place between independent enterprises (or organisations) in order to complete joint workflow or business process with mutual incentives: creating value-added services, creating return on investment, receiving a larger share of the market, or fulfilling some business domain need [5]. Instead of integrating enterprise computing systems together for this goal, it has become more attractive to run business transactions that temporarily involve a select set of business transactions from different service providers [6]. Therefore, the focus has to be moved from technical integration of transaction systems to the management of cobehaviour of services within contract-governed service collaborations [1]. *Cobehaviour* refers to the externally detectable exchange of messages between the services involved.

The following sections give our definition of the concept business transaction, provide an overview of our business transaction management system and its necessary functionality and two levels of transaction support, namely ACID transactionality of eContract state across the managed business transaction and subjective transactionality of interactions at the level of selected business network.

A. Business transaction

We define a *business transaction* as a complex interaction between multiple business services that

- strives to accomplish an explicitly shared business objective, one that is potentially extended over a period of time, or is of continuous nature;
- has clearly defined cobehaviour leading to this objective in terms of exchange of information and behaviour controls; that is, has
  - mutually negotiated conditions of success in terms of a change in the state of the business relationship; this state change is restricted to relevant features of the states of the business services involved; and
  - mutually negotiated but subjectively detectable breaches of the contracted cobehaviour;
- has clearly defined, mutually negotiated cobehaviour for breach recovery for each identified class of contract breaches.

Here, the term *business service* denotes a software-supported service that is provisioned by an independent enterprise, organisation, public agency, or even an individual. The granularity of a business service is such that it becomes relevant for creating collaborations between business service providers for value-added activities. Further, with the choice of *subjective detection* of breaches we want to emphasise that each partner defines the rules for detecting an unacceptable situation independently; some of these rules are derived from the joint contract, but there is no obligation to monitor all of the contract and no obligation not to react a contractually valid but uncomfortable situation.

The above definition combines several contemporary definitions (e.g., [7], [8], [9]). It is also closely related to the definition of community in the open distributed processing reference model ODP RM [10], [11], and maps to our work on inter-enterprise collaboration management within open service ecosystems [4].

New elements in this definition include the expectation of the business transaction to work across autonomous decision-making domains and form a peer system where each of the parties need to monitor its own concerns about the fulfilment of the objective and occurrence of breaches against its own interests. Furthermore, breach situations cannot be solved bilaterally, but require the attention of all the business transaction parties. In contrast to the above-mentioned related work, the breach recovery process cannot be embedded in the shared computing platform (as there is no uniform platform) but needs to be explicitly contracted before the business transaction is started. Furthermore, the effects of the recovery can be more disruptive than the related work items indicate.

We have identified two kinds of business concerns that need to be injected to the business transaction management mechanism. The first group comprises situations where there is a need for changing the collaboration topology or membership. Examples include trust changes and enterprise policy changes that cause the enterprise to drop

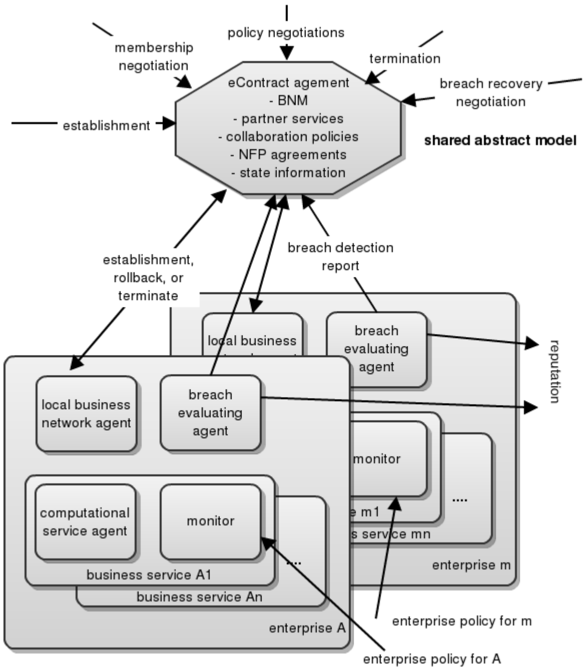

Figure 1. Reflective management of inter-enterprise business transactions.

off an existing collaboration. The second group comprises situations where independent partners need to make private, subjective decisions about the continuation of the transaction and thus causing the choice of the appropriate alternative in the transaction control processes. Examples include not being able to provide ordered goods, provide agreed SLA (service level agreement), or a client cancelling an order for private reasons.

B. Business transaction management system overview

In this kind of design challenge where conceptual consistency needs to be technically supported, the reflective system pattern is useful: *A reflective system* [12] is a system that can reason about its own structure, behaviour and state, and furthermore, can act upon itself. For reasoning and action-triggering, the system includes a causally connected metalevel model of its own characteristics, making it able to detect changes in itself or its environment, make decisions based on that information, and change its own status or create interaction with its environment. In the reflective model, a system model provides a metamodel of the controlled real system. The system model can be changed due to a) requests by human administrators of an involved enterprise, for example, or b) actions triggered by asynchronous events arising from the underlying system. Changes in the system model trigger management operations on the real system. In reflective systems, this two-dimensional dependency is termed causal connection.

The reflective business transaction management system is depicted in Figure 1. The joint business-level concerns are captured into the eContract agent that is used for reflective management of a set of business services with subjective decision-making capabilities in monitors. The eContract comprises the business network model in use, partner information derived through the service offers, contract policies embedding regulatory rules as well, nonfunctional property agreements, and technology requirements. Furthermore, the eContract carries process models for known recovery cases. At each partner, subjective decision-making rules can be injected in part to the local monitors. These rules can determine whether the local strategical policies are fulfilled by following not only the eContract policies but also potentially overriding local enterprise policies. These rules can be formed to determine a locally acceptable ”sensible state of affairs” from the business point of view. As monitors are a potential performance bottleneck, the arbitration on which triggers eventually are forwarded to the eContract is performed as a separate, local step.

The potential changes of the eContract include changes of the services involved and thus some of the technological aspects involved, policies and nonfunctional property agreements. Explicit operations on the eContract agent modify the topology of the systems involved in the business transaction. In these operations, the eContract agent must seek agreement from all involved parties before transactionally committing to the change. The operations of importance address the establishing or termination of the collaboration, changes in the membership or collaboration policies, and starting of a selected breach recovery process. Agreement by the parties involve private decisions on strategic commitments, trust on the collaboration as a whole, and privacy needs.

In breach situations, triggered operations on the eContract must address the transactional properties of the collaboration: is the collaboration to be terminated, rolled back to a previous consistent state in terms of business semantics, or recovered forward to a next ”sensible business-semantics state”. These recovery actions are not always strictly following ACID properties, but rely on a relaxed interpretation where compensation operations, subjective acceptance of a situation as ”sensible” based on pre-selected criteria for the collaboration state, and ecosystem-level facilities like reputation dissemination are effectively used.

In summary, the business transaction management system provides two levels of transactionality: firstly, traditional ACID transactionality over each eContract that governs an inter-enterprise collaboration; and secondly, a relaxed transactionality over the specific type of collaboration specified by the eContract by the selected business network model. The relaxation of ACID properties involves the following: a) each partner can subjectively decide which information, control, and nonfunctional property elements they monitor and accept; b) each eContract has a selected business process declared for the agreed compensation and recovery needs;

and c) the eContract state information can be used by each partner to detect any orphaned inter-enterprise computing requests that they can terminate as wasted effort.

We have to note that this business transaction management activity takes place within a service ecosystem. The ecosystem governance regulates the quality and validity of the BNMs and eContracts, and collects feedback from collaborations within the ecosystem for further improvement purposes [1], [3], [4].

The above reflective model is easily mappable to the Pilarcos open service ecosystem framework (further details [1], formalisation background in [4]) where ecosystem infrastructure services facilitate the management of inter-enterprise collaboration lifecycles. Essentially, the Pilarcos open service ecosystem infrastructure agents include a) the populator service that discovers sets of interoperable service offers to fulfil a given business network model, and b) eContracting and breach recovery support [1].

On an enterprise level, local support agents handle decision-making tasks that cannot be outsourced from the enterprise domain of control, such as contract negotiations, policy enforcement, trusted monitoring and private risk analysis. In the context of this paper, automated contract negotiations involve making choices between prepared templates embedded in the business network models, and adjusting parameters such as exact pricing [1].

Through local monitoring, the outcome of a transaction produces experience information that is shared with other actors in the ecosystem through reputation systems. The shared information, in turn, affects the trust decision-making in other enterprises on other transactions. In short, this means that local events have an effect also beyond the specific transaction where they occur.

C. *eContract state changes*

The eContract agent is responsible for maintaining the collaboration lifecycle state changes and provide interfaces for the model-level operations. Figure 2 illustrates the major phases that are briefly described below.

The lifecycle starts with an initiator first picking an existing business network model from the available ecosystem repository and asking the populator to either suggest service offers that could fill the roles declared in the model or, if the collaboration parties are already known, to check the eContract proposal interoperability potential.

For a plausible eContract proposal, a negotiation round is used to allow each partner to either accept or reject the proposal, or to provide a counteroffer. Each partner utilises local policies for decision-making on whether collaboration is interesting, acceptable and worth the risk. In terms of benefit, partners must evaluate their incentives to participate in a collaboration; whether it provides monetary gains or better networking, for example, and ensure that the proposed collaboration model is in alignment with the goals of the

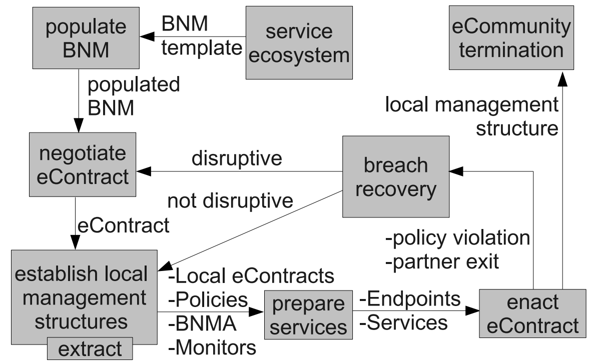

Figure 2. Lifecycle of inter-enterprise collaborations.

enterprise. For acceptability, the shared business process must not violate local policies, which override negotiated contracts in conflict situations. Furthermore, trust decisions are made to evaluate whether a given action is worth the estimated risk [13]. The risk estimate is based on experience information on the outcomes of earlier collaborations with the suggested partners. As first-hand experiences are not always available, experiences are shared through reputation systems [13].

When all partners agree, the eContract is passed on to the next phase where all partners receive a copy of the eContract agreed upon. Rejection of an eContract proposal terminates the collaboration. A third option is that one of the partners propose an alternative contract.

The establishment of the eContract at each organisation involves deduction of locally binding policies from the contract, deciding on which ones are to be monitored, and setting up monitors to control eContract-based policies as well as local enterprise policies. the eContract copies can be held by each organisation's local business network management agents (BNMAs). Once this local management structure is set up, it technically realises the behaviour demanded in the local copies of the contracts with the help of local services. After picking these services, communication endpoints need to be created to allow communication channels to be bound.

The above-mentioned steps of collaboration establishment involve, at each partner, the enrollment of local management structures, as shown in Figure 3. The eContract acts as a coordinating agent for the respective local copies. The local copies each have a set of policies assigned that govern local service behaviour. The additionally assigned monitors check for local behaviour compliance and the BNMA ensures compliance within the collaboration. Finally, the prepare stage populates the lowest technical level with matching services and corresponding endpoints for communication channels. For dismantling, the terminate stage removes the entire management structure step by step without leaving behind a disorderly state of the business semantics, orphan processes or database entries.

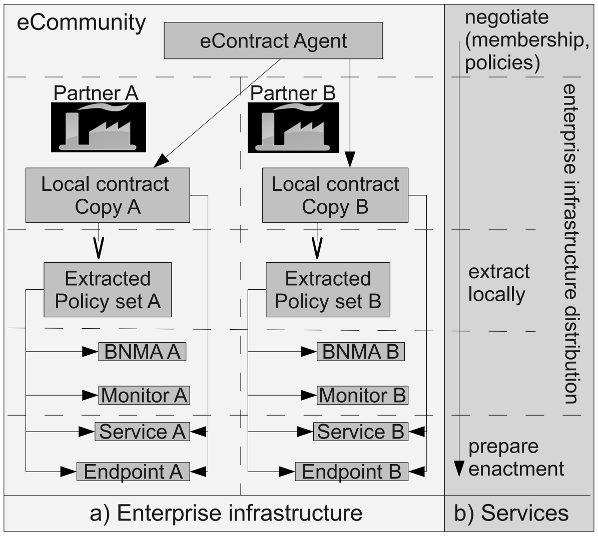

Figure 3. Local management structures.

During the collaboration enactment phase (i.e., operational phase), a total termination, partner replacement, or policy violation can be triggered by any of the local monitors.

Further elaboration of the business process control processes are discussed next in Section III. The discussion includes technical level efficiency discussions on abandoned computations. The subjective detection of breaches, and decision-making logic on strategy, trust and privacy reasons are elaborated in Section IV.

III. Business transaction management system

For the construction of the business transaction model and ecosystem framework, our work has had three threads. First, we have constructed prototypes of the ecosystem infrastructure services that facilitate the state changes of eContracts [1]; processes were informally mapped to transaction management processes. Second, we have created a formal model of the processes controlling each collaboration on the structural level [14], as is described in this section; interleaving with this structure controlling work, subjective decision-making algorithms are discussed in the subsequent sections. Third, the ecosystem metamodel framework has been created to ensure that the information artefacts manipulated by the control processes meet the needs of engineering and collaboration governance, and are manageable by the ecosystem infrastructure services [2], [3].

For the model formalisation, we chose Coloured Petri Nets (CPN) [15], a language for design, specification, simulation and verification of systems. CPN has a graphical representation with a set of modules, each containing a network of places, transitions and arcs. The modules interact through well-defined interfaces. Places may hold multiple tokens that carry colour, i.e., attributes with values. Transitions fire when all input places hold the required sets of tokens and produce condition-adhering tokens into output places.

In this paper, we focus on only two CPN-models that cover enactment, rollbacks and termination, while the formalisation of the full lifecycle can be found in [14]. The utilised data-flow tokens are briefly shown before the process models.

A. *Data flow*

For the formalisation, the data elements of the overall lifecycle were declared for all refinement levels. Figure 4

level	CPN-module	token color	description	type
1	eCommunity lifecycle	sO	service offer that matches a service type	integer
		sOs	service offer source for communication channel est.	
		sOt	service offer target for communication channel est.	
		pA	partner of an eCommunity	
		rO	role a partner can fill	
		eC	eCommunity identification	
		eCo	eContract based on which partners of an eCommunity transact	
		n,r,k,p,l,q,s	counter variables	
		assigned	service offer assigned to a role	boolean
		processed	partner prepared for eContract counteroffer re-distribution	
		decision	for negotiated contract proposal (agree|disagree|counter)	string
		outcome	input for eCommunity continuation or termination (agree|disagree|counter)	
2	create	bNM	business network model that gets populated	integer
		m	counter variable	
		sT	service type required for BNM role	
3	populate	ch	channel of communication between services	integer
4	interoperability checking	rOs	role source for communication channel establishment	integer
		rOt	role target for communication channel establishment	
4	contract extraction	spec	specification of extracted eContract	string
4	agreement finalizing	result	whether all eCommunity partners agree on an eContract proposal or not	boolean
		distributed	contract distributed to partner	
4	disagreeing	z	counter variable	integer
		eCo_new	new eContract from a counteroffer to be negotiated	
2	perform	bnma	business network model agent	integer
		sE_l	local service of a respective eCommunity member	
		mO	monitor for observing policy adherance of eCommunity partners	
		sE	electronic service that is enacted	
		lp	local policy extracted from a local contract copy	
		lpnr	counter of local policies	
		s,x	counter variables	
		lC	local contract for respective eCommunity partners extracted from the eContract that coordinates the first	
4	contract establishment	insert	service inserting to local contract	boolean
		extracted	instances of contract copies for negotiation	
5	governance distribution	errorID	error identity	integer
		error	error for synchronizing main contract and local copies	boolean
5	prepare	eP	published endpoint for allowing services to communicate	integer
6	preparation error	prepErr	preparation error in the context of assigning an electronic service to a service offer	integer
		assignErr	assignment error in the actual assignment of an electronic service to a service offer	
		sEr	service error related to concrete electronic servce, e.g., deadlock	
4	operate	tc	termination criteria, either full for eCommunity or partial for disruptive partner change that rolls back to a negotiation stage	integer
5	enact	startErr	start error when a stopped service is re-started again	integer
6	policy service removal	pAnr	number of partners	integer
4	nondisruptively manage	vl	local vote for replacing a policy-violating partner or reconciling	integer
		lp1	another local policy	
		vote	for policy violation of partner to leave or stay in eCommunity	string
4	nondisruptively choose	pA_new	new partner to replace one who violated a policy	integer

Figure 4. Acronyms, names and descriptions of token colours.

lists relevant token colours [14] with their hierarchic service-refinement availability mentioned in the left column (1 for the top level and 6 for the most detailed refinement). Token colours are present for all lower but not for any higher CPN-refinement-hierarchy levels. The fourth column explains the purpose of a token colour for a lifecycle. The integer-type tokens mostly represent an identification number and string-type tokens are either eContract negotiation outcomes or eContract proposals. Boolean-type tokens represent decision points in the lifecycle.

B. Process formalisations

The first key CPN-module depicted in Figure 5 formalises eContract enactment and triggering of alternate recovery and termination processes. Here, the tokens representing the business services in the collaboration (i.e., filling the roles in the eContract), gather into the central state *enacting services*. We assume that these services are well formed and thus discrete business-process specifications with a unique start state and tasks relating to each other in sequences or parallelisms that lead to a unique end state. Respective services may stop for a period of time and restart again for enactment.

To perpetually enact respective services requires the involvement of the related BNMAs, monitors and policies. These local management structures come into existence by the service *governance distribution* that is not shown here. Each independent monitor raises a violation trigger when it detects a failure to adhere to the agreed policies, behaviour or service-level agreement. Examples of common failures to detect include nonresponsive systems.

The second model covers breach recovery, as shown in Figure 6. When a collaboration behaviour violates the eContract policies, the corresponding violation assessment is triggered, and thus enables alternative recovery scenarios that may either be disruptive or non-disruptive. The enactment stage may trigger either disruptive or non-disruptive business-semantics rollback and compensation steps for policy violations and partner changes.

Termination of an enactment is either partial or complete: In the case of total termination, the local management structures are stepwise dismantled and the lifecycle of the transacting collaboration is brought to its final logging state. Partial termination performs a subset-dismantling of the local management structures and rolls the collaboration identification back to the negotiation phase where the partners of a collaboration may populate roles differently. Besides the total termination, also rollback processes use the partial termination module.

Disruptively reset contract proposal and service offers uses the collaboration identification from the *enact*-service of Figure 5. This simultaneously triggers a partial termination after which the collaboration identification enters a re-enabled negotiation. Before the negotiation starts, it is necessary to reset the service offers for extracting contract proposals for each partner. When a re-negotiation begins, the same populated business network model is the foundation for the disruptively rolled back collaboration identification.

The module labelled *disruptively choose replace contract partner* consists of two parts. One part removes the local management structures and enables selection of the new partner. The second part, *nondisruptively change local contract* reinserts the new local contract with renewed identification keys. As a modification trigger, we consider, e.g., a minor change in the business environment of the party that is not significant enough to justify entirely newly created local management structures.

In *nondisruptively manage policy violation*, the first procedure is a vote about the severity of the violation. At present, the formalisation assumes a randomly chosen vote to determine the response. If the collaboration ignores the policy violation because of its insignificance, the token representing the violated policy and the related service are input for resuming the eContract enactment. The collaboration may also decide to either disruptively or non-disruptively replace a partner, as described above. The final option is to vote that the violation to be desirable, because of an unpredicted change of the collaboration environment that results in a "pragmatic" violation, for example. Thus, the response is to replace the unsuitable local policy. Finally, the collaboration agrees to reconcile the committed policy violation, e.g., a warning issuance for the concerned party to not repeat a violation.

C. Model-checking results

We used CPN Tools[1] for the modelling and subsequent correctness and performance checking, especially on aspects relevant for system developers:

- reachability of CPN-module end states in manual or fully automated simulation token games (as state explosion makes full computational verification challenging for this size of models);
- detection of loops as they are a potential source of livelocks that prevent desired termination reachability; loops require specific attention with respect to effectiveness of exit conditions, such as elements of business-level policy control (Section IV);
- performance peaks during runtime either for the design of sufficient resources or for restricting the load with business-level policy control (Section IV);
- full system utilisation for ensuring that each part of the modelled system actually is used in some scenario; and
- consistent termination, i.e., consistent home markings that ensure simple testing of a real system.

The model-checking results (Figure 7) we generate by focusing on CPN-modules where the generated state-space

[1] http://cpntools.org/

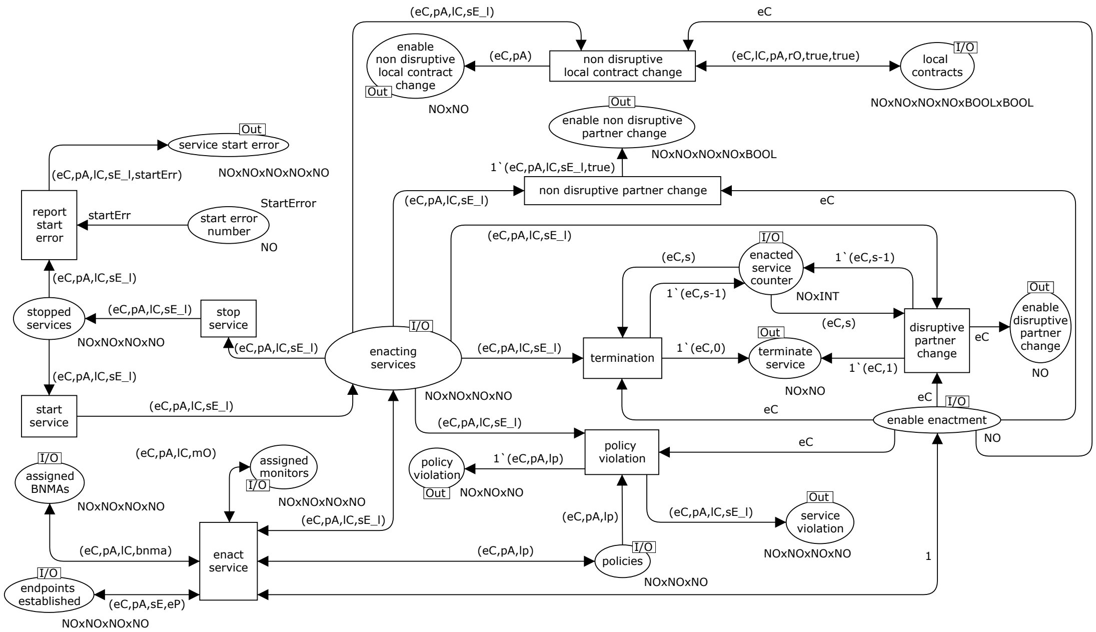

Figure 5. Enacting an eContract and triggering collaboration recovery or a termination (*enact*).

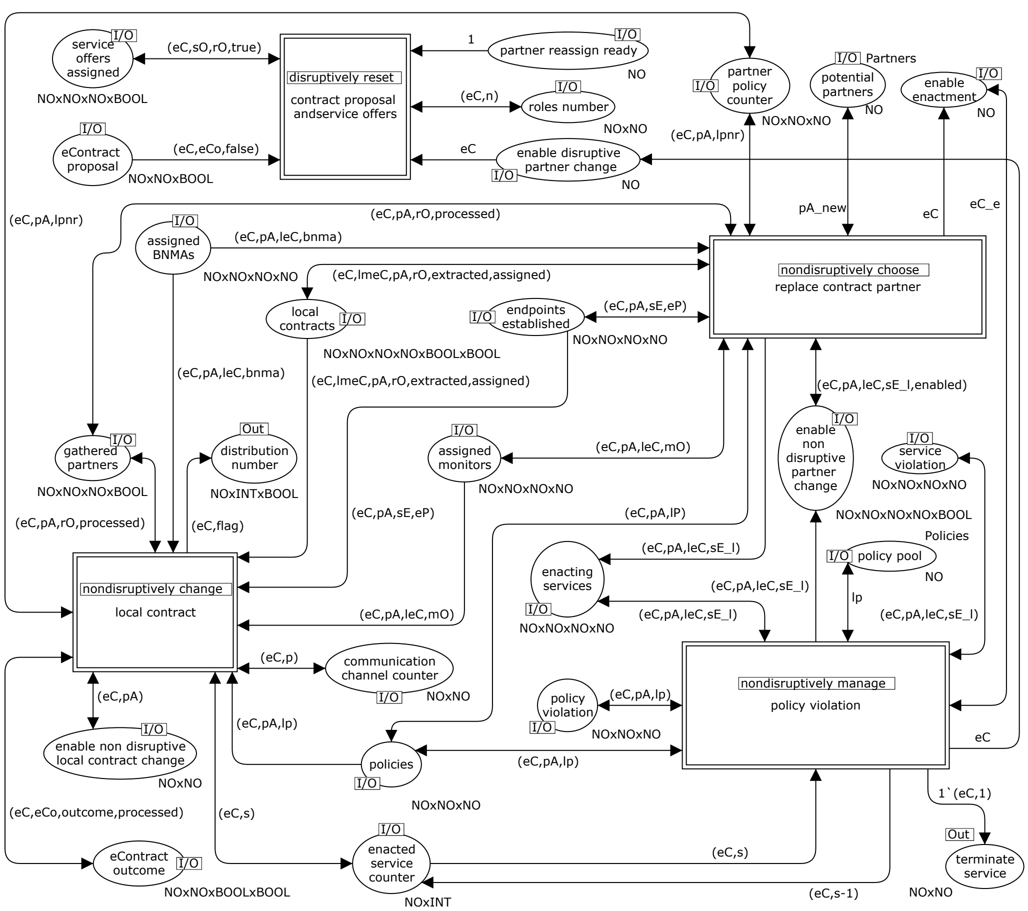

Figure 6. Options for rolling back violations in an eContract enactment (*business-semantics rollback*).

is computationally feasible to analyze automatically. The results are grouped into modules within which only atomic transitions take place. For the $negotiate$-service there are three separate outcomes. Either all collaboration partners agree on a contract proposal, or a counteroffer is newly negotiated, or one partner disagrees entirely and terminates the proposal. For model checking, we separately generate results where only one respective outcome option is enabled and two remaining options disabled.

Loops exist in the services for contract negotiation and enactment. The contract negotiation loop is self-restricting in nature, as it only walks though the collaboration partners one by one. However, finalisation of the negotiation comprises loops, because a counteroffer triggers a restart. To limit these logically infinite loops, the prototype realisations provide resource-limiting parametrisation for partners to limit the loop [16]. The enactment loop occurs for repeatedly stopping and starting the enactment of a service until either an exception occurs or the overall enactment enters a termination phase. The test results for remaining services in Figure 7 show they do not contain loops.

Performance peaks represent places in the system that are pontential performance bottlenecks, as they require peaks in computing power. Peaks exist in all but the services *BNM selection* and *terminate*. For the $populate$-service, peaks occur not only for populating roles but also for checking if channel requirements and data-semantics match. Again, the populator prototype has resource policy guidance for limiting these peaks, in order to avoid denial-of-service attacks, for example [16].

For all three $negotiate$ cases, peaks are visible due to counteroffers and negotiation finalisation. Also modules for *governance distribution* and $extraction$ of policies are demanding, as well as $preparation$-service that assigns local services, their endpoints and communication channels. All these elements are effectively dependent on the amount of populations and negotiations performed, and therefore, stay limited.

While no tested service has any home marking, the model-checking results for dead markings differ. Having multiple dead markings and no home markings means that the testing of implementations is more time and resource intensive as only a big number of test cases ensure correctness. With respect to utilisation tests, the results show that there were no unintentionally unused subsets of the models; in Figure 7, entries yes* indicate that a subset of the CPN-module was intentionally disabled as an available runtime path to manage state-space explosion issues or for allowing an analysis to focus on specific enactment paths.

		Model property				
	Service	**Loops**	**Performance peaks**	**Utilization**	**Homemarking**	**Dead marking**
	BNM selection	no	*evenly balanced*	yes	no	multiple
	populate	no	populating roles, interoperability checking	yes	no	no
negotiate	**negotiate with forced agreement**	yes	contract extraction, agreement finalizing	yes*	no	multiple
negotiate	**negotiate with forced counteroffer**	yes	contract extraction, distribute eContract to partners	yes*	no	no
negotiate	**negotiate with forced disagree**	yes	contract extraction, agreement finalizing	yes*	no	multiple
	governance distribution and extraction	no	extraction of local policies	yes*	no	multiple
	prepare	no	assign service, create service endpoint, publish endpoint, check if service operational	yes*	no	multiple
	enact	yes	enact service, report start error	yes*	no	multiple
	terminate	no	*evenly balanced*	yes	no	multiple

Figure 7. Results from model checking.

IV. Subjective decision points and ecosystem-level consequences

The above-discussed CPN-models include important decision-points where, by design choice, the actual decision rules are not specified: the business transaction management system allows subjective business-level decisions and technical control according to enterprises' private policies. Below, the injection of subjective decision-making facilities is briefly discussed. Furthermore, ecosystem-level effects from all business transactions is discussed as an essential addition to the model.

The *enterprise-level decision-making points* form two axes: on one hand, the targets of decisions made depend on what stage the collaboration is in its life cycle. On the other hand, the types of decision policies form a hierarchy of three levels: whether an action is interesting, acceptable and worth the risk.

In the population phase, the business network initiator first selects the type of business network model based on what kinds of business are in its field of interest. It may also perform static analysis on whether the transactions specified in a business network model are in immediate conflict with its local policies relating e.g. to the privacy of sensitive information. For example, local policy may require that customer credit card information is not passed unencrypted to other collaboration roles than the payment handler. Once the populator has produced a set of eContract proposals with the roles populated with service offers, the initiator makes a trust decision on which proposal it will start negotiations, based on a reputation-dependent risk analysis.

In the negotiation phase, all actors make their local trust decisions on whether the proposed collaboration, possibly with suggested changes, is similarly interesting, acceptable and worth the risk. If all accept the contract, the collaboration can be set up.

During the operational phase, actors must dynamically determine when an action allowed by the eContract is violating higher-priority local policies, which would make the action unacceptable. A privacy policy and the eContract may both allow, for example, that a partner can make queries to get information on specific customers of the enterprise, while access to the full customer database is locally protected with a limit of a specific number of queries within a given time window. They also privately monitor the behaviour of other actors towards themselves, producing experience information that is stored locally and shared through reputation systems. This information feeds back, through the ecosystem facilities, to later trust decisions on whether the collaboration continues to be worth the risk; the decisions are made whenever new resources are committed to the collaboration. A logistics service provider may accept the general pattern of delivering goods as a part of a given collaboration, for example, but if it finds that payments for its service do not arrive on time, it may refuse to commit additional transport capacity into a new delivery order if it is better used elsewhere.

On the ecosystem level, the main goal is to enable quality control and evolution of the ecosystem as a whole. Indeed, if a service or business network model constantly cause recovery needs in collaborations, the ecosystem knowledge should be accumulated to trigger corrections or restricting the advocating of untrustworthy partners.

Feedback from specific collaborations to the ecosystem emerges in two major forms: feedback directed at actors, and feedback directed at collaboration models, BNMs. On one hand, collaborators provide feedback on each others' behaviour in the form of shared reputation information. This is stored and disseminated through reputation systems, which help establish peer-based control in the ecosystem by punishing misbehaviour through reputation loss, which in turn affects the misbehaving actor's chances at being selected as a partner in further collaborations [13].

On the other hand, the outcomes of entire collaborations reflect on the collaboration models used by them, and detected shortcomings direct the evolution of the collaboration models. These shortcomings can include altogether missing collaboration patterns, for example, or problems in specific models, such as structural privacy issues in business processes demanding sensitive information to be shared, but not applying adequate protection measures to support it.

Put together, the infrastructure services reduce the cost of establishing and running inter-enterprise collaborations within the ecosystem, and provide support for distributed governance through ecosystem-wide processes for e.g. disseminating reputation information.

V. Related work reflections

Related work for our business transaction management concepts is found in very different domains, ranging from traditional transaction management systems adapted to SOA architectures to those of developments on service ecosystem domain.

When we consider the business transaction lifecycle and the recovery processes for it, the formalised model gives a solution that resembles on the highest level a Saga [17] transaction that is based on compensation mechanisms of chained transactions. Sagas divide a long lasting transaction into sequentially executed relaxed sub-transactions with ACID properties to meet the needs of business semantics in inter-enterprise collaboration.

However, our eventual goal includes support for developing new recovery and compensation models for new business network models. These additional models are dependent on the business network model properties, such as the use of stateless services, invariants involved, and external side-effects of the processing. At present, only the infrastructure-supported general recovery modes have been modelled, but these design-specific alternative recovery collaborations are further elaborated in our metamodelling framework [18].

In this era of independent organisations involved with shared business transactions, many frameworks relax the ACID properties of transactions because of the autonomy of partners (for example, those surveyed in [19]). Our solution resembles to some extent the Delta Grid approach [19]: a) use of assurance points (in Pilarcos: epochs or tasks) for reached secure states of the collaboration, b) use of abstract model for defining the collaboration, c) each partner in the collaboration monitoring user- or designer-defined, contracted invariants as no locking is appropriate, and d) utilising domains of processing together with local recovery that can be escalated to more collaboration-wide processing.

In related work, ecosystem-level considerations arise in the field of inter-enterprise trust management [13], but they have so far not been connected into business transaction management frameworks. TrustCoM, for example, monitors the virtual organisation members' adherence to service level agreements, and reports the experiences to a reputation system [20].

VI. Conclusion

We have investigated a way of injecting business concerns into the management of inter-enterprise business transactions. Our proposal continues our earlier work on open service ecosystem by adapting the ecosystem infrastructure services to facilitate state changes in the eContract lifecycle.

The proposal extends our work by formalising the generic recovery processes while still leaving space for business network model-specific alternatives. Moreover, the formal model includes decision-making points at which enterprise policies and local decision-making processes enable each partner to direct the eContract lifecycle. These subjective decision-points allow injection of strategical decisions, as well as decisions based on trust management and privacy preservation. For monitoring the progress of the joint business transaction, each partner can use self-selected acceptance criteria. This relaxes the normal ACID properties significantly, allowing each partner to protect their resources and assets as closely as they feel necessary. The relaxation also helps to encapsulate the heterogeneity of information and service models that takes place across partners.

The design contributes to the correctness, coherence and efficiency of business transactions in several ways. First, the correctness is facilitated by protecting ACID properties of the eContracts themselves, while the eContract contents embeds dynamic correctness invariants [4] for the collaboration. Coherence and alignment with business needs in each involved enterprise or organisation is supported by the two-level reflective model that allows private policies and decision-making to be utilised within the framework. The efficiency is supported by allowing each involved party to judge, based on the eContract state information, when a processing has been abandoned by its clients, thus enabling effective resource management.

A significant difference to other business transaction management techniques is the use of the ecosystem level for a dynamic source on shared collaboration information. More importantly, the ecosystem is seen as a target to which experience information on the behaviour of partners, services and business network models is reported. This supports the healthy evolution of the ecosystem as a whole, and thus makes the ecosystem-based business transactions more dependable.

VII. Acknowledgements

This work has been performed in the CINCO group (http://cinco.cs.helsinki.fi), especially in its Trusted Business Transactions project with funding from the Academy of Finland.

References

[1] L. Kutvonen, J. Metso, and S. Ruohomaa, "From trading to eCommunity management: Responding to social and contractual challenges," *Information Systems Frontiers (ISF) - Special Issue on Enterprise Services Computing: Evolution and Challenges*, vol. 9, no. 2–3, pp. 181–194, Jul. 2007.

[2] T. Ruokolainen and L. Kutvonen, *Handbook of Research on Non-Functional Properties for Service-Oriented Systems: Future Directions*. IGI Global, Dec. 2011, ch. 21: Framework for managing features of open service ecosystems.

[3] T. Ruokolainen, S. Ruohomaa, and L. Kutvonen, "Solving service ecosystem governance," in *Proceedings of the 15th IEEE International EDOC Conference Workshops*. Helsinki, Finland: IEEE Computer Society, Aug. 2011, pp. 18–25.

[4] L. Kutvonen, "ODP RM reflections on open service ecosystems," *Computer Standards & Interfaces*, 2012, in press. [Online]. Available: http://dx.doi.org/10.1016/j.sci.2012.01.005

[5] FInES Research Roadmap Taks Force of EC, *Future Internet Enterprise Systems: FInES Research Roadmap 2025*, Mar. 2012, draft document, version 2.0.

[6] X. Xu and Z. Wang, "State of the art: business service and its impacts on manufacturing," *Journal of Intelligent Manufacturing*, no. 22, pp. 653–662, 2011.

[7] M. P. Papazoglou and B. Kratz, "A business-aware web services transaction model," in *ICSOC*, ser. Lecture Notes in Computer Science, A. Dan and W. Lamersdorf, Eds., vol. 4294. Springer, 2006, pp. 352–364.

[8] T. Wang, J. Vonk, B. Kratz, and P. W. P. J. Grefen, "A survey on the history of transaction management: from flat to grid transactions," *Distributed and Parallel Databases*, vol. 23, no. 3, pp. 235–270, 2008.

[9] *Clark, J., Casanave, C., Kanaskie, K., Harvey, B., Clark, J., Smith, N., Yunker, J., and Riemer, K.: ebXML Business Process Specification Schema Version 1.01. UN/CEFACT and OASIS. 2001.*

[10] *ISO/IEC IS 10746-2 Information Technology — Open Distributed Processing Reference Model: Foundations, also published as ITU-T Recommendation X.902*, 2010.

[11] *ISO/IEC IS 10746-3 Information Technology — Open Distributed Processing Reference Model: Architecture, also published as ITU-T Recommendation X.903*, 2010.

[12] P. Maes, "Computational reflection," Ph.D. dissertation, Vrije Universiteit Brussel, 1987.

[13] S. Ruohomaa, "The effect of reputation on trust decisions in inter-enterprise collaborations," Ph.D. dissertation, University of Helsinki, Department of Computer Science, May 2012.

[14] A. Norta, "Safeguarding Trusted eBusiness Transactions of Lifecycles for Cross-Enterprise Collaboration," University of Helsinki, Department of Computer Science, Helsinki, Finland, Tech. Rep. C-2012-1, 2012. [Online]. Available: http://www.cs.helsinki.fi/u/anorta/publications/TBT_SOCC.pdf

[15] K. Jensen, L. Michael, K. L. Wells, K. Jensen, and L. M. Kristensen, "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems," in *International Journal on Software Tools for Technology Transfer*, 2007, p. 2007.

[16] L. Kutvonen, "Trading services in open distributed environments," Ph.D. dissertation, University of Helsinki, 1998. [Online]. Available: http://www.cs.helsinki.fi/TR/A-1998/2/A-1998-2.pdf

[17] H. Garcia-Molina and K. Salem, "Sagas," in *In: Procs. of the 1987 ACM SIGMOD International Conference on Management of data (SIGMOD'87)*, San Francisco, California, United States, 1987, pp. 249–259.

[18] T. Ruokolainen and L. Kutvonen, "An architecture framework for facilitating sustainability of open service ecosystems," in *Service-oriented Enterprise Architecture for Enterprise Engineering (SoEA4EE) Scientific Workshop (SoEA4EE'2012).*

[19] L. Gao, S. Urban, and J. Ramachandran, "A survey of transactional issues for web service composition and recovery," *International Journal on Web and Grid Services*, vol. 7, no. 4, pp. 331–356, 2011.

[20] M. W. Lutz Schubert *et al.*, "TrustCoM reference architecture, Deliverable D09," TrustCoM WP27, Tech. Rep., Aug. 2005.