A reference architecture for managing dynamic inter-organizational business processes

Data & Knowledge Engineering 91 (2014) 52–89

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Editorial

A reference architecture for managing dynamic inter-organizational business processes

Alex Norta [a], Paul Grefen [b], Nanjangud C. Narendra [c]

[a] Tallinn University of Technology, Department of Informatics, Akadeemia Tee 15A, EE-12616, Tallinn, Estonia
[b] Eindhoven University of Technology, Faculty of Technology and Management, Department of Information Systems, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
[c] Cognizant Technology Solutions, Bangalore, India

ARTICLE INFO

Article history:
Received 13 September 2011
Received in revised form 1 April 2014
Accepted 4 April 2014
Available online 25 April 2014

Keywords:
eSourcing
Inter-organizational
Reference architecture
Business process
Ecosystems
Cross-enterprise

ABSTRACT

For improving the efficiency and effectiveness of business collaboration, the need emerges to inter-organizationally match e-business services. Recent research activities show heightened attention into that direction with the ecosystems-emergence of service-oriented computing clouds. As this increases the business-, conceptual-, and technical governance complexity, a need exists for using a reference architecture to evaluate and design standard-, and concrete architectures for business-to-business (B2B) collaboration. In this paper, we fill that gap by presenting the eSourcing Reference Architecture eSRA that emerges from B2B-research projects and we check this with a scenario-based validation method. We demonstrate how eSRA enables a quick evaluation of not only research-based B2B-architectures but also of industry application suits. That way, we show the usability and applicability in that with the help of eSRA, system designers directly establish a comprehensive understanding of fundamental B2B concepts and develop higher-quality domain-specific architectures.

1. Introduction

Companies envision that inter-organizationally connecting their business processes allows them to transact in a faster and more cost-effective way with better quality of service. We define an inter-organizational business process as a B2B-composed flow of related activities that together create a customer value. This must happen in a dynamic way in that they are formed by the automatic integration of the sub-processes of the involved organizations. Thus, dynamic means that during the setup phase, collaborating organizations find each other by searching business-process marketplaces and match externally contractual-sphere processes of larger in-house processes.

Such collaboration between manufacturing companies in the B2B-domain is complex from a business, conceptual, and technological point of view. A need exists to develop business-collaboration systems according to a uniform vocabulary and template set of guidelines for achieving meaningfully automated business-interoperability for which a reference architecture serves as a suitable artifact. We define a reference architecture [31] as a template solution for an architecture of a particular domain such as service-driven enterprise architecture where the structures and respective elements and relations provide templates for standard-, or concrete architectures.

Recent research publications with relevant standard architectures exist, most notably the CrossWork architecture [49] and also further scientific proposals of business-collaboration architectures [27,62,67,106]. The main feature of a reference architecture is the design in a top-down fashion while standard architectures evolve in a bottom-up way abstracting from concrete architectures. Yet, a

E-mail addresses: alex.norta@gmail.com (A. Norta), p.w.p.j.grefen@tue.nl (P. Grefen), ncnaren@gmail.com (N.C. Narendra).

http://dx.doi.org/10.1016/j.datak.2014.04.001
0169-023X/

standard architecture should be reference-architecture adhering. Similarly, concrete architectures for application systems facilitate the automation of inter-organizational business processes to different degrees. The most notable examples are SAPs Sourcing [95], ORACLEs Sourcing [83] application suits, or the Sourcing platform [56] from IASTA.

While these bottom-up emerged architectures above focus on specific aspects of inter-organizational business processes, they cannot serve as general guiding architectures. This paper fills the gap by investigating the following research question: how does one equip system designers with the means to rapidly establish a comprehension of the suitability, completeness and quality of inter-organizational business-process architectures and their application instantiations? To answer this question, we employ a required top-down development approach by choosing the suitable *eSourcing* concept [75,76] that results from studying best practices of B2B with the CrossWork industry partners from the automotive industry. eSourcing is a framework for harmonizing on an external layer the intra-organizational business processes of a service consuming and one or many service providing organizations into a B2B supply-chain collaboration. Important elements of eSourcing are the support of different visibility layers of corporate process views for the collaborating counterpart and flexible mechanisms for service monitoring and information exchange.

The recognized notion of process views is essential to support sourcing [40,69] and the eSourcing concept is to the best of our knowledge the only one for supporting a consumer organization to find and connect to a service provider. Other approaches address the construction of process views [37,44], which is not specific to sourcing [51]. Most importantly, while the eSourcing concept enables diverse ways of process-view creation and matching, other approaches support only one way. Yet another class of approaches considers the problem of matching process descriptions of services [45,59,75] but does not relate this to the problem of projecting internal processes to process views. eSourcing is technology independent but feasible to realize with means of service-oriented cloud computing [103,107], the latter facilitating the creation of complex cloud-ecosystems. Ecosystems have business dimensions of varying semantics and ontologies, social dimension for different stakeholders, and architectural dimensions for platform engineering [39]. The adoption of a decentralized software ecosystem results in intra-organizational processes to behave in a dependable way [18] on a composed inter-organizational layer. Architectures are pivotal in that case for guiding the coordination-design across organizational system boundaries. As the interfaces of service-ecosystems affect coordination soundness, introducing new functionality triggers a reconciliation of the interface specifications [19,24]. Also [104] explains that an ecosystem of services and participants requires a governance structure for management of engaging in a partnership based on contracts as a commitment between legal entities.

The eSourcing Reference Architecture eSRA specifies the governance structure to facilitate the development of high quality ecosystems with means of service-oriented cloud computing. eSRA supports the instant understanding and communication between architecture designers of such ecosystems and is guiding for a fast evaluation of existing architectures. eSRA serves as a guiding reference architecture for the design of application systems with fundamental design principles and specifies high-quality functionalities for rapid application development. For this reason, we define a reference architecture [88] as a division of functionality with exchanged data flow mapped onto software elements that together implement the required functionality for a specific context and the business goals of the stakeholders. The method for designing eSRA is coherent and systematic in that it combines existing best practices from scenario-based evaluation methods [12], business-collaboration models [75,76], architecture styles [26,63,96] and design patterns [48].

The structure of this paper is as follows. First, Section 2 presents the conceptual context of inter-organizational business with a three-layer framework populated by a formal collaboration model. With that conceptual foundation, Section 3 infers a collaboration lifecycle from which we deduce a set of functional requirements. Furthermore, in accordance with literature studies, Section 3 also defines non-functional architecture requirements specified in accordance with the conceptual context and ends with a discussion about what scenario-based evaluation method to choose for eSRA. Next, Section 4 shows the scenario-method evaluated eSRA specification on three refinement levels that takes into account the conceptual business collaboration context and the sets of functional and non-functional requirements. The presentation of eSRA we textually present in a way that shows the implementation feasibility, i.e., the text comprises implementation suggestions, or pre-existing services that are candidates for an eSRA-implementation with means of Service-Oriented Cloud Computing (SOCC), or other concrete pre-existing technology. Section 5 gives the results of a second evaluation step for the reference architecture where we separately check eSRA-adherence to all functional- and non-functional requirements. This incorporates comparisons with existing architectures for business collaboration from research and industry. Finally, Section 6 presents related work and Section 7 gives conclusions and future work. The appendices show results of the ATAM-evaluation and subsequently conducted control-survey.

2. Features of inter-organizational business processes

The prerequisite for reference-architecture development is the study of best practice in industrial reality. For that purpose, we employ the CrossWork [70] case study stemming from a leading truck producer and suppliers from a cluster of small enterprises. A project objective is the sound formalization of inter-organizational business processes [76] to allow for feasible realization with SOCC. For reaching the latter target, a gap exists that eSRA fills. Thus, for reference-architecture development, this section deduces a set of functional- and non-functional requirements from the formalized collaboration model. Based on case studies conducted with industry partners in the CrossWork project [49], Section 2.1 explains the conceptual business-collaboration setting. Finally, Section 2.2 discusses a refined collaboration model within that framework.

2.1. Business-collaborations

Research about business collaboration in the EU research project CrossWork [49], reveals a scenario where an original equipment manufacturer (OEM) rests as a capstone on top of a collaboration-pyramid, is responsible for engineering a product and setting up the machinery and plant construction for production. Suppliers of standardized parts and raw materials are at the bottom of the business-collaboration pyramid.

In Fig. 1, the OEM is a service consumer and suppliers are service providers. A potential service provider must assure to varying degrees the capability of behavior mirroring. For inter-organizationally matching service-consumers and providers, the parties face conceptual-, methodological, and technological differences. Additionally, both parties keep competitive advantages and business secrets hidden from each other.

To manage the collaboration differences, a three-layer framework [50] is a suitable model. Translating this conceptual collaboration model into a formalization, Fig. 2 shows internal layers that comprise legacy systems belonging to the information infrastructure of a respective collaborating party. Organizing legacy systems as orchestrateable services on an internal layer caters towards a heterogeneous system environment.

At the conceptual layer of Fig. 2, business processes are mapped to their respective internal layer to orchestrate wrapped legacy systems. The external layer of Fig. 2 stretches across the domains of collaborating parties. Projected parts of conceptual processes are a subset on the external layer so that business secrets remain hidden internally. The parties first investigate the demands of service consumption and the ability of service provisioning while automatic and dynamic forging characterizes the process-based collaboration. Note that although Fig. 2 shows a bi-directional collaboration, the formalization scales to multiple parties, i.e., a service-consumer's in-house process may comprise multiple consumer spheres that tie in multiple service providers via external-layer projected to and matched contractual spheres.

2.2. Dynamic process-view matching

We consider *eSourcing* [76] focuses on structurally harmonizing on an external layer the intra-organizational business processes of a service consuming and one or many service providing organizations into a business collaboration. Important elements of eSourcing are the support of different visibility layers of corporate process details for the collaborating counterpart and flexible mechanisms for service monitoring and information exchange. Note that differently to Electronic Data Interchange (EDI) [23,55,86] where legacy-system integration happens directly to the detriment of security, process-view matching hides legacy systems behind an internal-, conceptual- and an external inter-organizational system layer.

Starting with the domain of the service consumer in Fig. 2, an in-house process on the conceptual layer adheres to the properties of a Workflow net (WF-net) [4]. Informally, a WF-net has one unique start-state with one token, one unique end-state and all activities always lead to the unique end-state comprising one token while all other states are empty. The in-house process contains a subnet termed a consumer sphere that is visualized with a gray ellipse. At the border of the consumer sphere, interface-places are labeled passive nodes. Only one interface place is *i*-labeled and only one is *o*-labeled. The other interface places are either *in* or *out*-labeled to denote an exchange direction of business-critical information between the in-house process and its contained consumer sphere. Furthermore, the labeling implies whether an interface place has an input arc or an output arc in the sphere. If an interface place is *i* or *in*-labeled, it has one output arc to an active node in the sphere. If an interface place is *o*- or *out*-labeled, it has one input arc from an active node in the sphere.

The in-house process maps to the internal layer of Fig. 2 with legacy systems. The consumer sphere turns the external layer to a contractual sphere after a projection that a service provider enacts. For establishing a contractual consensus [13] between the collaborating parties, the projected contractual spheres must be isomorph. A provider sphere with additional nodes on the conceptual layer, complements the contractual sphere. Hence, the refinement remains opaque for the collaborating counterpart and we refer to [76] for further details about the formal hierarchy relationships between the processes on the respective collaboration layers.

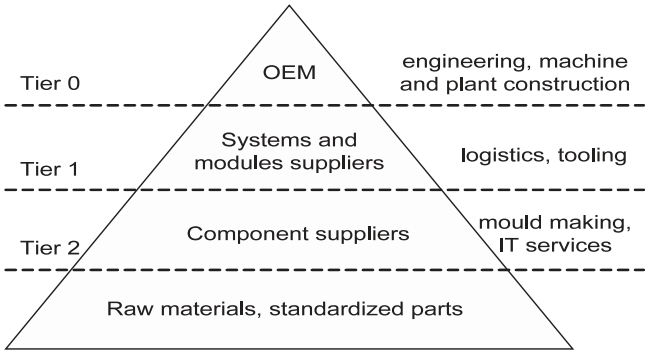

Fig. 1. Supply-chain hierarchy in an enterprise collaboration.

Finally, in Fig. 2, there exist referencing arcs connecting several passive nodes of the respective spheres. These referencing arcs establish a transitive relationship between the consumer sphere and the provider sphere that ensures a correct start and end of service provision during the enactment of an eSourcing configuration. The notations of these referencing arcs of Fig. 2 are part of so called monitorability patterns [77] that also cover the referencing of active nodes contained in the consumer sphere and the provider sphere. This way, a variable degree of enactment-progress monitoring is possible for a service consumer.

After giving a sound and scalable collaboration-model formalization [76] that results from a benchmark B2B industry-case [70], the foundation exists for the next step towards the eSRA reference architecture. Thus, we proceed to giving a set of colaboration-model deduced functional- and non-functional requirements for the eSRA specification.

3. Functional- and non-functional eSRA requirements

As eSourcing is a socio-technical collaboration framework, the requirements of eSRA must reflect this fact [85]. For this context, we follow the steps of requirement development, documentation, and validation. Functional requirements cover specific behavior of systems while non-functional requirements specify criteria for judging the operation of a system, rather than specific behaviors. According to [34,36], a set of characteristics exist for good requirements in system development. First, a requirement must be cohesive in that it addresses one issue only. Completeness refers to a requirement being fully stated without missing information. Consistency means a requirement is not contradictory in itself or in correlation to other requirements. A requirement must be atomic and without conjunctions.

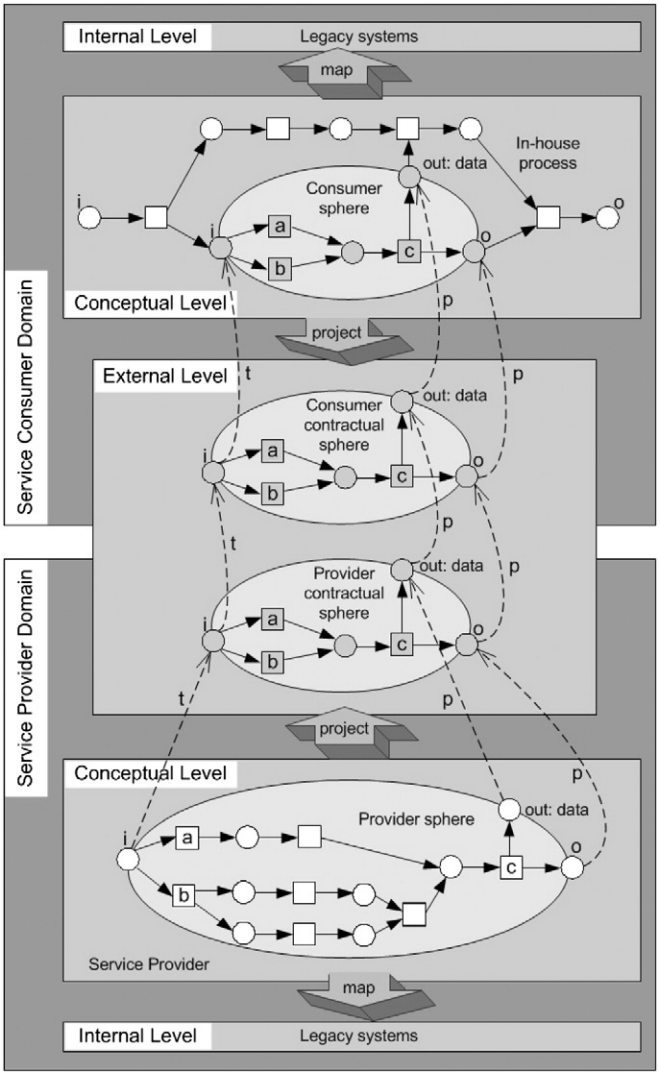

Fig. 2. eSourcing in a three-layer business-process framework and its verification [76].

To satisfy the requirement characteristics for functional requirements in eSRA, we specify a discrete lifecycle for setting up an eSourcing-collaboration as depicted in Fig. 2. As the discrete steps of the setup lifecycle are cohesive and atomic, it serves as a means for deducing in Section 3.1 complete and consistent functional requirements. Based on studies of standard literature [20,99], carefully selected non-functional requirement specifications follow in Section 3.2. For documentation, we translate the requirements in the sequel into UML-component diagrams [46] and subsequently perform a validation with a scenario-based method [12].

3.1. Functional requirements

Based on the eSourcing-collaboration setup lifecycle of Fig. 3, we postulate for eSRA the functional requirements listed in the sequel. Note that the lifecycle assumes a broker [41,79] serves during setup time as a trusted third party between collaborating parties for depositing and matching in a negotiation process service-offers and service-requests. The lifecycle relates to the inter-organizational business-collaboration context of Fig. 2 as follows. For the latter depiction, we give a set of so-called setup-time interaction patterns in [75] for establishing in flexible ways an eSourcing configuration according to two category-dimensions. First, the assignment-dimension is static when the collaborating parties are known before setup, dynamic when an open set of collaborating parties bid in an anonymous market for service provision and/or consumption, and semi-dynamic when a limited number of collaborating parties engage in a setup phase. Second, the direction-dimension comprises categories for the eSourcing direction between the respective conceptual- and external layers of collaborating parties. Consequently, the categories are internal-to-external, in-sourcing, out-sourcing or external-to-internal. The setup lifecycle of Fig. 3 is the generalization of these interaction-dimensions with their respective categories.

The generalized lifecycle in Fig. 3 for the interaction pattern in [75], comprises transitional steps that we specify as follows.

a.) Support for the conceptual formulation of business processes and their accompanying rules.
For the respective conceptual-layer processes of collaborating parties, modeling components must be available. In order to prevent a constant reinvention of business-process constructs, it should be possible to store in and retrieve from a database such conceptually formulated constructs.

b.) Mapping details from the conceptual layer business-process to the internal layer.
In mapping, the tasks of a conceptual-layer process bind to the Web-service ports that wrap legacy systems so that an enactment-time orchestration of the latter is possible. During setup time, the mapping employs service types as preparation for replacement with full service implementations during runtime.

c.) Projecting from the conceptual layer business-process details to the external layer:
For projecting, translation functions to different notation-formats must exist to cater for business-process heterogeneity, e.g., from BPEL on the conceptual layer to BPMN on the external layer.

d.) Brokering capability of projected business processes.
Both the service consumer and the service provider must be able to place their projected business processes into a broker

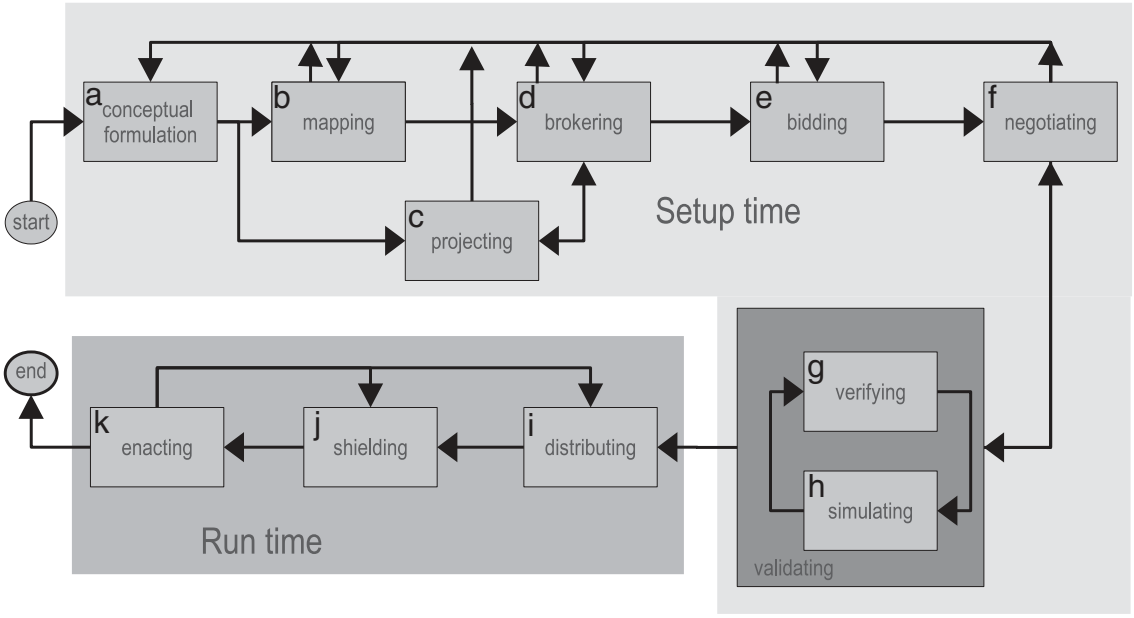

Fig. 3. A lifecycle for setting up an eSourcing configuration.

environment of a trusted third party service. This functionality is important for collaborations in an anonymous collaboration environment. Service offers and service requests must be searchable for potential business partners.

e.) Bidding capability for projected business processes.
The bidding environment is part of a trusted third party service. The collaborating counter-party evaluates and chooses the subjectively best bid. Note that service-level agreements [98] support the decision of what constitutes the best option.

f.) Negotiation support for setting up a collaboration configuration with known collaborating parties.
When collaborating parties have found each other, they need a trusted third party platform for starting the contracting negotiations on the external layer of a collaboration configuration. This negotiation involves the projection of business processes to the external layer until achieving a matching that establishes a consensus between the service provisioning and service consumption. Negotiation also involves an agreement on how many monitorability constructs to include for the collaboration configuration.

g.) Verifying perspectives of a collaboration configuration.
From a dominant control-flow point of view, it is important to verify a collaboration configuration for correct termination [76]. A verification must ensure that a service provisioning internally adheres to the externally promised collaboration behavior. Verification must also cover other perspectives than control-flow, e.g., data- flow [72], resources, transactional properties, and so forth.

h.) Simulation of a collaboration configuration.
Despite verification, errors may still occur during service enactment. Hence, a simulation component for business processes must be available for a-priori enactment simulation.

i.) Distribution of business processes.
For distribution, components on the conceptual-layer interface as data translators between the internal- and external layers. Such ad-hoc configurable translating caters for heterogeneous collaboration situations.

j.) Shielding of business processes and legacy systems on concern-separating layers.
Shielding must ensure the legacy systems that are Web-service wrapped and part of the internal layer, are safeguarded by data-monitoring functionality.

k.) Enactment of a ready collaboration configuration.
After a completed setup phase, the enactment of a collaboration configuration commences. The actual enactment components must be present on an internal layer for orchestrating legacy systems. Additional enactment components on the external layer need to choreograph the internal components of the respective collaborating parties.

3.2. Non-functional requirements

After carefully studying non-functional requirements [20,99] for eSRA-contextual suitability, we choose a subset of suitable requirements based on the industry-partner cases in the CrossWork project [49] with the following distinction between a set of requirements discernible via execution of a system and those that are not. Again, the CrossWork case studies determine the specification-details of these requirements to the context of eSRA-specification.

Studying literature for harvesting eSRA-relevant non-functional requirements [30,99] for applicable requirements, in the discernible case we consider interoperability, performance, security, high automation, flexibility, and usability. In the non-discernible case, the requirements are modifiability and integrability. Finally, we consider requirements on the architecture, namely completeness, feasibility, scalability, and applicability. First, we specify requirements for eSRA that are *not discernible at runtime*. Their effectiveness is not investigatable during the setup time of a collaboration configuration. Note that we consider non-functional requirements for the abstraction layer of eSRA. Thus, requirements for less abstract architectures we don't include below, e.g.; reliability, which means a system performs as intended over a period of time that is only applicable for eSRA implementations.

3.2.1. Modifiability

An eSRA-compliant system changes and adapts during its lifecycle to the business context. Additionally, it harmonizes inter-organizationally heterogeneous system environments comprising of commercial software that undergoes regular updates.

3.2.2. Integrability

An eSRA-compliant system consists of separately developed and integrated components for which the interface protocols between the components must match. Hence, integrability between the components of eSRA must be ensured in the reference architecture.

Next, we specify the system requirements for eSRA that are *discernible during runtime* because their effectiveness is investigatable during the setup and enactment of collaboration configurations.

3.2.3. Interoperability

An eSRA-compliant system must interoperate at runtime with information systems supporting other business functions, e.g., planning, logistics, production, external partner systems. Business-, conceptual-, and technical heterogeneity challenge dynamic interoperability.

3.2.4. Security

Refers to the ability of resisting unauthorized attempts at usage and denial of service while still providing to trusted users with good reputation. To address security, trust- and reputation problems, several architectural strategies are possible. An authentication server checks collaborating parties, monitors, inspects and logs network events. The communication of a system may reside behind a firewall, and so on.

3.2.5. High automation

Companies require systems that provide a high level of automation during the setup, enactment and post-enactment phases. Hence, eSRA-compliant systems must provide for possibilities of a high degree of meaningful collaboration automation that process tedious and repetitive work and allow humans to focus on the remaining action.

3.2.6. Flexibility

Electronic enterprise collaboration is a highly dynamic process that involves the enactment of diverse activities, the participation of diverse partners, and the exchange of diverse data [75]. Hence, an eSRA-compliant system must allow diverse collaboration scenarios and permit the inter-organizational harmonization of heterogeneous concepts and technologies, i.e., multiple collaboration models [75] between business parties.

3.2.7. Usability

Refers to an eSRA-compliant system being easy to use for business collaboration. This requirement we decompose into the following three areas [22]. *Error avoidance* is support to anticipate and prevent common errors that occur during a collaboration configuration. Closely related is *error handling*, to help with system support a user to recover from errors. *Learnability* refers to how quickly users master using the system. For an eSRA-compliant system, inter- and intra-organizational knowledge workers need automation support to conceptualize business processes and project them externally to business counterparts until a collaboration consensus comes into existence. During enactment, employees dynamically slip into specified roles to carry out assigned tasks. During the setup, enactment, and post-enactment phases, experienced parties such as administrators must intervene in cases of exception-handling escalations the system does not solve with specified exception handling and compensation.

Additionally, we consider *requirements on architecture*. *Completeness* is the quality of eSRA-compliant systems comprising the set of components for setting up and enacting a collaboration configuration.

The requirement of *feasibility* means, an eSRA-compliant system is implementable within a limited time framework and work effort.

Scalability refers to the ability of eSRA-compliant systems to combine more than two collaborating parties into one configuration.

Applicability states that eSRA is instrumental for guiding the design of business-collaboration systems.

Portability means, eSRA serves for instantiating business-collaborating systems independent of the industrial domain and collaboration heterogeneity with respect to business-, conceptual-, and technological system infrastructure. Hence, platform-specific considerations architecture layers encapsulate that enable portability with an abstract interface to the environment.

Performance means the computational and communicational strain is low in collaboration configurations for the setup-, and enactment phase. Hence, it is important to ensure that all phases of a collaboration are carried out within a desirable response time and without exponential need for computing power. However, for eSRA the latter two requirements are not meaningful since the reference architecture is aimed at describing functionalities for eSourcing and the design principles for an eSourcing system.

Next, we present the requirements-adhering eSRA-models and specifications.

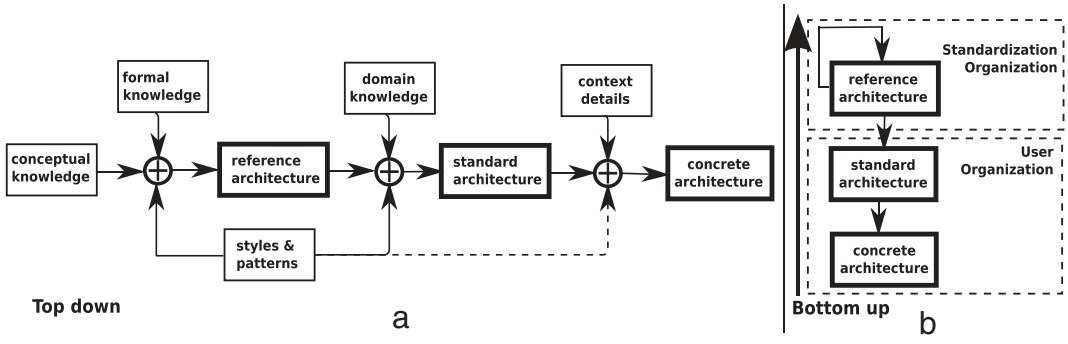

Fig. 4. (a) Top-down architecture-framework process (b) Bottom-up practitioner process.

4. Designing eSRA

The specification of eSRA we closely relate to the requirements of the earlier section. The architecture design is in accordance with the principle of functional decomposition of a system. This part–whole-principle decomposition is also known as "separation operation". Thus, at each refinement level of eSRA, the identified components provide functionalities that do not overlap with the remaining components located on the same layer. To achieve completeness, we design eSRA in a top-down way, i.e., the components on the first level comprise detailing components.

In [11], the provided classifications allow a positioning according to which eSRA evolves through sound development stages. Following an adaptation in Fig. 4(a), we use a top-down approach starting from a conceptually and formally explored collaboration model [45] to deduce the reference architecture's functional properties. On the other hand, styles and patterns assure the architecture adheres to non-functional goals. The injection of domain knowledge for standard architecture development based on findings from the CrossWork project [49] aims for high quality with equally applying architecture styles and design patterns. The benefit for system architects is they are able to rapidly deduce a concrete architecture instantiation by taking concrete application-context details into consideration. The practitioner way depicted in Fig. 4(b) shows a bottom-up approach where a user organization creates first a standard architecture out of multiple concrete-architecture experience that matures into a reference architecture. In the case of eSRA, the participants from industry and academia in the CrossWork project form the user group that we mature in this paper to a reference architecture for consideration by a standardization organization.

In this research, the first concrete-architecture specification [75] is a facilitator for multiple organizations from industry of the CrossWork-project to take advantage of research results for designing their own inter-organizational business-collaboration architectures. In this paper, we prove with an evaluation against commercial eSourcing systems in the sequel that eSRA is a reference architecture. After these workshop- and documentation-based evaluations, a survey sent to members of industry and academia ensures the validity of the evaluation and caters for final eSRA-specification adjustments. As future work, the latter is a proposal for a standardization organization to transform into a reference architecture.

Given the rise of service-oriented architectures (SOA) for business collaboration, we explain why eSRA is specified in technology-agnostic UML-component diagrams [46]. Strengths of SOA are that publicly accessible services promise increases in efficiency and effectiveness for setting up and enacting business collaborations. A weakness of SOA is that it does not consider internal realizations for services but it focuses on interfaces. Instead, to assure modifiability and loose coupling for eSRA-adhering business-collaboration systems that are heterogeneous in their realization, we use technology-independent architectural styles and design patterns. Thus, we specify a general reference architecture with conceptual refinements to permit the fast communication of essential design ideas, i.e., specific components for data exchanges along semantics.

The remainder of this section shows the three refinement levels of UML-component diagram specified eSRA-models.

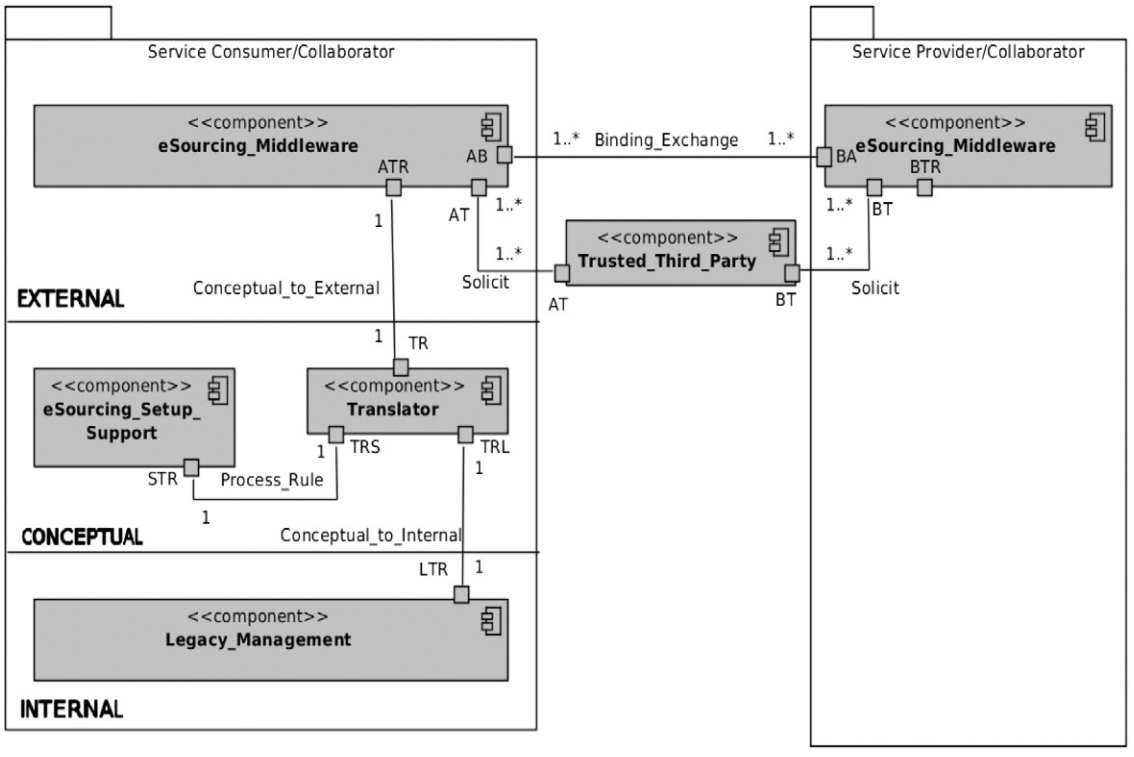

Fig. 5. Highest level of eSRA.

4.1. First refinement level

The highest abstraction level in Fig. 5 shows two parties that contain the same set of components distributed across an external-, conceptual-, and internal layer [50] in accordance with Fig. 2. Note that although Fig. 5 shows a bilateral collaboration case, the architecture scales to more parties [76]. Thus, the package labels in Fig. 5 are service consumer and provider, or collaborator respectively for indicating two existing orthogonal collaboration directions. First is a master–client collaboration type [76] with one service consumer and one or many service providers whom themselves may slip into the role of service consumer if a supply chain has multiple tiers. The second orthogonal collaboration direction is peer-to-peer [64] where several collaborators hold parts of a process that they merge into one, e.g., when a set of small- or medium-sized enterprises who are by themselves too small as providers form one larger business process for a service consumer. The relationship cardinalities between respective components in Fig. 5 express this scalability.

The *eSourcing_Middleware* in Fig. 5 is part of the external layers of every respective collaborating party. This component is the main enabler of interoperability and direct information exchange exists between the *eSourcing_Middleware* components of each collaborating party to synchronize the respective components. Important functions of this component are on the one hand a protection from security breaches in communication by offering a façade to collaborating counter-parties that make the synchronization easier through one communication point in a loosely coupled way. Additionally, the *eSourcing_Middleware* comprises a contracting client that also evaluates the reputation of services or collaborating counter-parties and manages the identity of the latter. Finally, global engines for a coordinated enactment of workflows and rules synchronize through the façade.

Between the collaborating counterparts, a component termed *Trusted_Third_Party* exchanges business-relevant information with the *eSourcing_Middleware*. A *Trusted_Third_Party* is necessary for several reasons that result from satisfying the functional requirements *d* to *g* (see Section 3.1). Firstly, collaborating parties expose service requests or service offerings to the *Trusted_Third_Party* for public evaluation. Secondly, the *Trusted_Third_Party* performs a verification of services and checks quality features of eSourcing configurations before enactment to prevent collaborating parties from revealing business secrets. However, this component is not mandatory during setup when the parties already agree a priori to collaborate with specific services.

The conceptual layer of Fig. 5 depicts two components, namely the *Translator* and the *eSourcing_Setup_Support*. The first component exchanges information after a format transformation between the external- and internal layer for reconciling the business-, conceptual-, and technological complexity of systems involved. As different concepts and technologies exist on the external- and internal layer, bridging them by translation functionality facilitates heterogeneity management. The

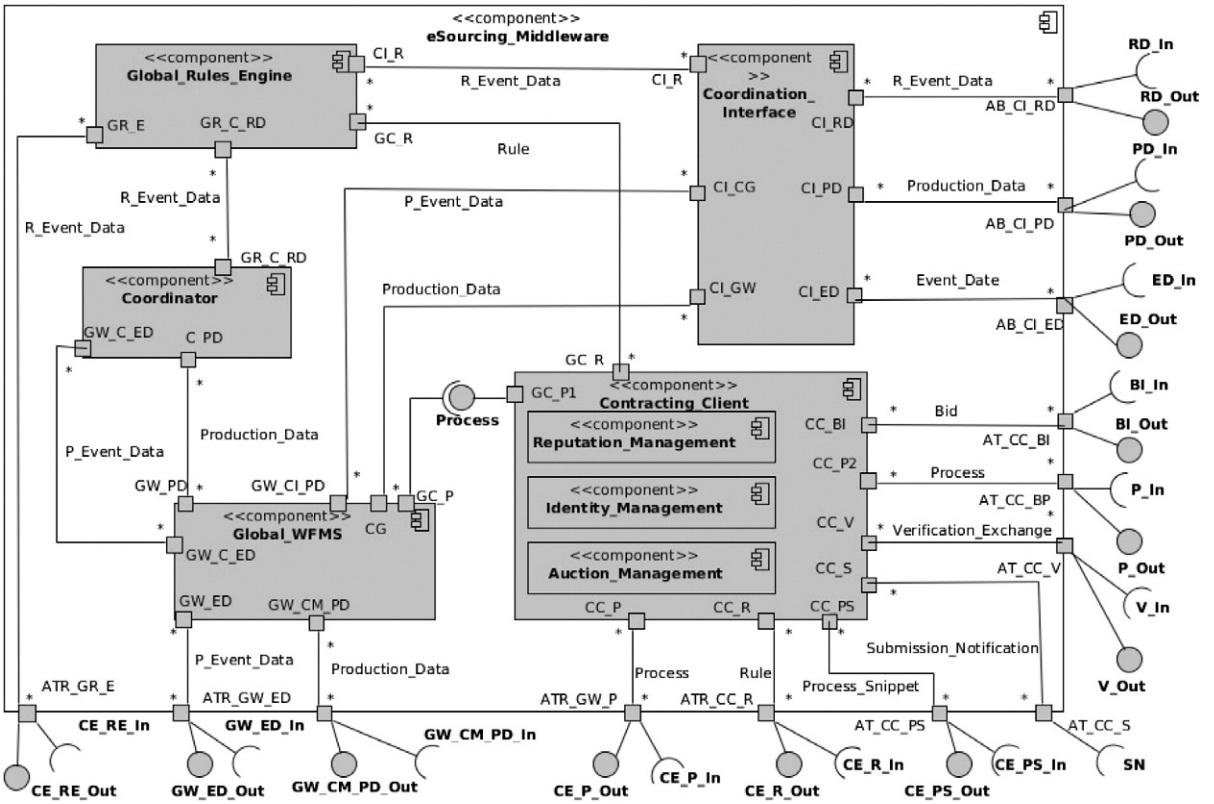

Fig. 6. External-layer collaboration with the *eSourcing_Middleware*.

eSourcing_Setup_Support comprises components for modeling business rules and processes. Libraries with available pre-specified processes and rules speed up the design phase. Finally, a local verifier component checks the soundness of designed processes and rules.

The internal layer depicts a *Legacy_Management* component that interfaces with the *Translator* of the conceptual layer. Inside the former are coordinated local enactment engines for business processes and rules that coordinate via the conceptual layer with the enactment engines of the external layer. Databases deliver data to the respective enactment engines and the latter orchestrate the ports of service-wrapped legacy systems on the external layer.

4.2. Second refinement level

The subsequent UML-component diagrams refine the top-level of Fig. 5. The *eSourcing_Middleware* in Fig. 6 contains a *Coordination_Interface* that is a security façade for protecting a collaborating party from malformed or malicious input from attackers. Consequently, the *Coordination_Interface* comprises special screening routines that assume all input data is harmful until proven otherwise. With extra runtime processing, screenings for *Binding_Exchange* data (see Fig. 5) result not only in delays but exceptions occur when the screening does not result in a security clearance, e.g., for messages with XML-documents, attachments of other formats, and data for enactment, or events communication. Another function of the *Coordination_Interface* is to shield from disclosing internal business process-details or other information when an exception occurs such as data-screening failures. Finally, we infer that the *Coordination_Interface* is the sole communication channel to the counter-party and it prevents accessing the internal domain that may constitute an integrity compromise and violate the loose-coupling principle.

The *Contracting_Client* communicates *Bid*-, *Process*- and *Verification_Exchange* instances with the *Trusted_Third_Party*. The *Contracting_Client* component provides support for the management of an e-contracting process [13] and semi-automatically assembles business processes by using *Process_Snippet* instances from the corresponding database of the *Trusted_Third_Party*. The *Contracting_Client* poses the advantage for eSRA in that multiple contracts for one single service target a specific type of collaborating party respectively. Furthermore, contracts again support loose coupling to internal legacy systems and governance independence with the process views of the external layer being parents of the conceptual-layer processes. The ports at the bottom of the *Contracting_Client* in Fig. 6 interface with the *Translator* component.

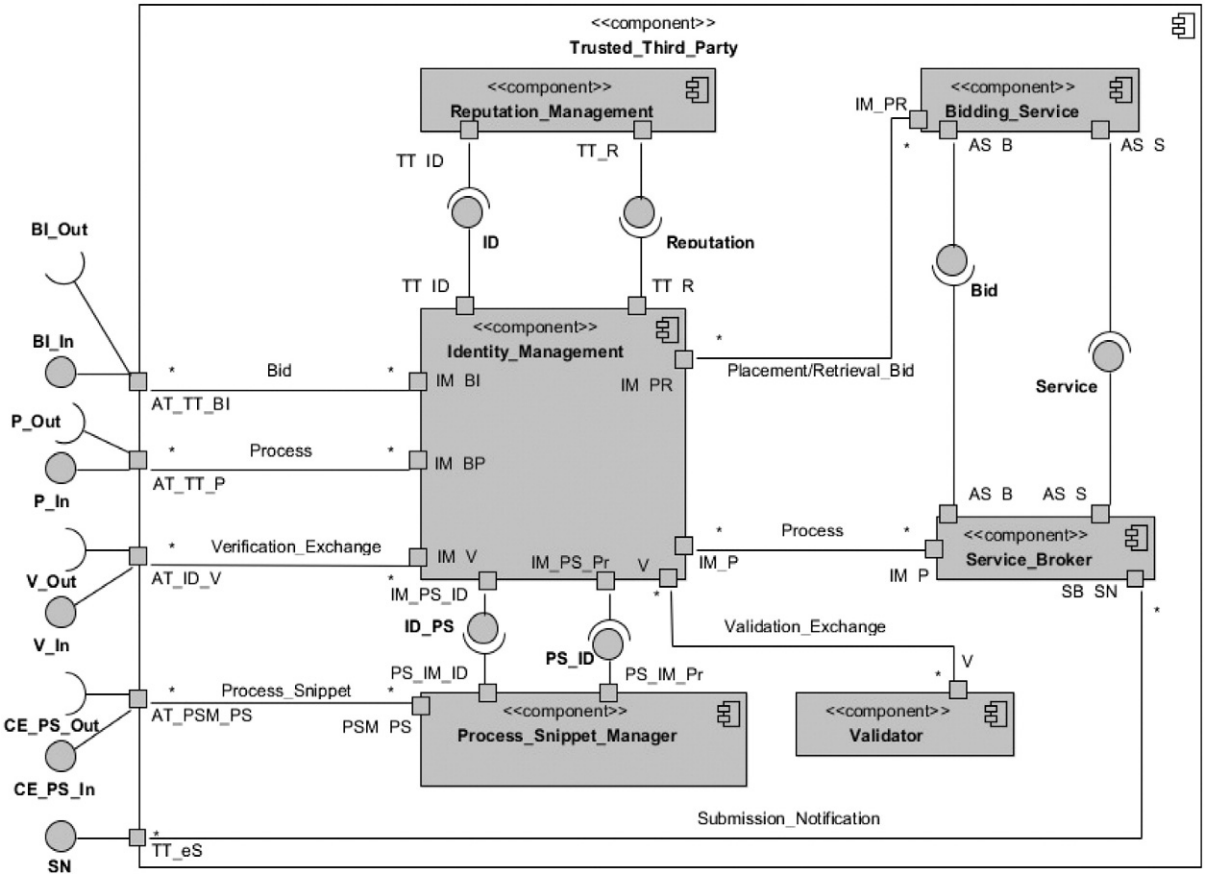

Fig. 7. The *Trusted_Third_Party* component.

After specifying the collaboration configuration for enactment, the *Global_WFMS* enacts the contractual sphere *Process* of a respective party. Using a *Global_WFMS* realizes the management of a process abstraction for governance independence. This abstraction establishes loose coupling to the in-house processes and a flexible compositionality into an eSourcing service. The related *Rule* instances the *Global_Rules_Engine* processes and the *Coordinator* assures during enactment both components remain in sync by communicating event-, production-, and rules data. The *Global_WFMS* and *Global_Rules_Engine* also exchange production-, and event data with each other.

The *Trusted_Third_Party* component in Fig. 7, interacts with the *eSourcing_Middleware* through the contained *Identity_Management* component that uses the *Reputation_Management* component to prevent an exchange with parties who lack trustworthiness and reputation. That way, it is possible to verify credentials of a potential collaborating counter-party if lack of trust exists, or if the consumer requires access to multiple services. A positive effect is the cutting down of communication overhead by a *Trusted_Third_Party* compared to a scenario of directly connecting to every single potential counter-party for the purpose of authentication and issuing a token for accessing the process. We refer to [13] for more details about trust management.

The bids and process proposals in Fig. 7 move to the *Brokering_Service* and *Service_Broker* respectively. The first coordinates the set of trading rules for the exchange of buying and selling services through bids and awarding a sale to the highest bidder. The latter arranges transactions between a buyer and a seller for a compensation that involves the matching of the contractual spheres of counter-parties. Depending on whether a collaborating party slips into the role of a service consumer or service provider, the contracting client submits or retrieves either service offers or service requests respectively from the service broker. The definition of a concerned party triggers the sending of a submission notification about a bid. The protocol between buyers and sellers follows a request for X (RFX) template comprising requests for information, bids, quotes, and proposal. Different types of RFX-protocols exist, e.g., a so-called English auction. As Fig. 6 shows a component for *Auction_Manager* embedded in the *Contracting_Client*, we infer the auctioning happens in a peer-to-peer way with support of the *Trusted_Third_Party* for storing bidding records.

The *Validator* in Fig. 7 caters for simulating and verifying on the one hand, processes for the service broker, on the other hand, the component checks the soundness [76] of an established collaboration configuration to prevent the disclosure of internal business secrets. For enhancing validation efficiency, an option is the abstraction from process details. For example in [43], a graph generated from the actual process-specifications formulated in BPEL allows for a rapid computing of control-flow similarity. Additionally, a partial validation equally enhances efficiency, e.g., an initial focus on the control-flow perspective for pre-screening,

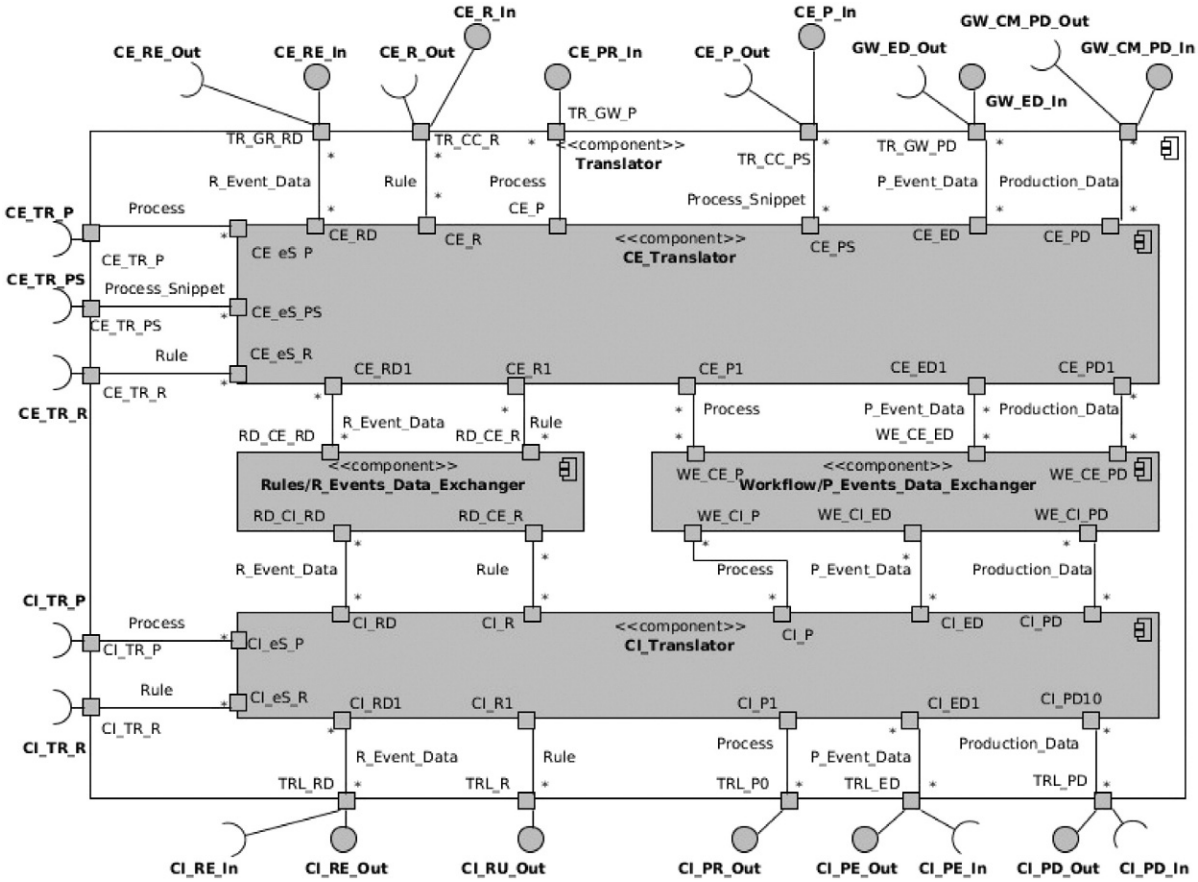

Fig. 8. *Translator* components between external- and internal layer.

followed by yet another partial validation of the remaining process set with a focus on the data-flow perspective. Finally, the *Process_Snippet_Manager* stores process parts for rapid business-process assembly. We refer the reader to [7] for further details.

The contracting client sends process snippets and composed processes and contractual rules to the *Translator*. The latter component translates the process snippets and composed processes and contractual rules for the heterogeneous system environment that exists on lower internal layers of a collaborating party. The *Global_WFMS* and *Global_Rules_Engine* are both distributed systems that send data to the *Translator* component depicted as light gray shaded in Fig. 8. Examples for *Global_WFMS* exist, e.g., in [53], the architectural design of a distributed multi-engine workflow system addresses poor scalability and strong coupling between process management and business application using Web services. A distributed workflow in [35] targets redundancies in virtual enterprises in terms of carrying out the same operations, job, or functions. An example for a distributed *Global_Rules_Engine* in [74] named ViDRE manages dynamic business logic by separating the implementation logic with exposing rules as Web services. That way, distributed rules are accessible across various rule engines.

The *Translator* contains two main components for translating data between the external-, conceptual- and internal layer. Dedicated exchanger components for rules and processes bridge the respective translators. The *Translator* ensures proper and targeted transmission of events stemming from rules and processes that occur during enactment. Such transmission is important for ensuring that there is an inter-organizational synchronization between the collaborating domains. For that, a party subscribes to agreed upon monitorability constructs in the inter-organizational business process. As the system environment in eSourcing is very heterogeneous, this also holds for the used protocols of the collaborating parties. Thus, the *Translator* bridges these differing protocols between external- and internal layers with dedicated bridging logics that enhance the messaging reliability. The bridging logics not only transform data such as files or state-changes but also transform models of rules and processes that differ in the collaborating domains.

The *CE_Translator* component of Fig. 8 translates data from the internal-, conceptual- to the external layer and vice versa. The component connects to the rules and process modelers of the *eSourcing_Setup_Support* component. The relationships between the *CE_Translator* and components in the *eSourcing_Middleware* we explain above. Two components exchange data between the *CE_Translator* and *CE_Translator*, namely the *Rules/R_Events_Data_Exchanger* and the *Workflow/P_Events_Data_Exchanger*. The exchanged data includes information about the destination. For example, several instances of WFMS and *Global_Rules_Engine* instances on the external layer and WFMS with *Local_Rules_Engine* instances of the internal layer enact several instances of different eSourcing configurations. The *CI_Translator* component translates data between components of the conceptual- and internal layers. The data-exchanger components bi-directionally translate events-, rules-, and production data to the *Local_WFMS* and *Local_Rules_Engine* on the internal layer. Furthermore, the *CI_Translator* receives contractual rules from the rules modeler and business processes from the process modeler. These rules and processes are translated to the *Local_WFMS* and *Local_Rules_Engine* on the internal layer. We re-visit the local components in the sequel of this paper.

The *eSourcing_Setup_Support* component of Fig. 9 is part of the conceptual layer of eSRA. The component has two core functions, namely, modeling business rules, processes and composing workflows for simulation and verification for correct termination. The *Rules_Modeler* must allow for the specification of different rule types, namely integrity-, derivation-, reaction-, and deontic rules [10]. Integrity rules are assertions that the evolving states and state transitions must satisfy, e.g., a payment must not be higher than 1500. A derivation rule specifies how to obtain the value of an element of interest, e.g., the second phase of the research project completes with the submission of a deliverable. Reaction rules follow a pattern where first an event detection follows a condition evaluation after which follows the invocation of a rule, e.g., when a payment happens (event) that equates to 100 (condition), the service enactment happens (action). Deontic rules determine the assignments of powers, duties in that they indicate the rights, obligations, and prohibitions of a party, e.g., the order manager refuses requests under the price of 100. The *Process_Modeler* must provide recommendation systems at design time for assuring that the processes are sound and specification happens in a fast way.

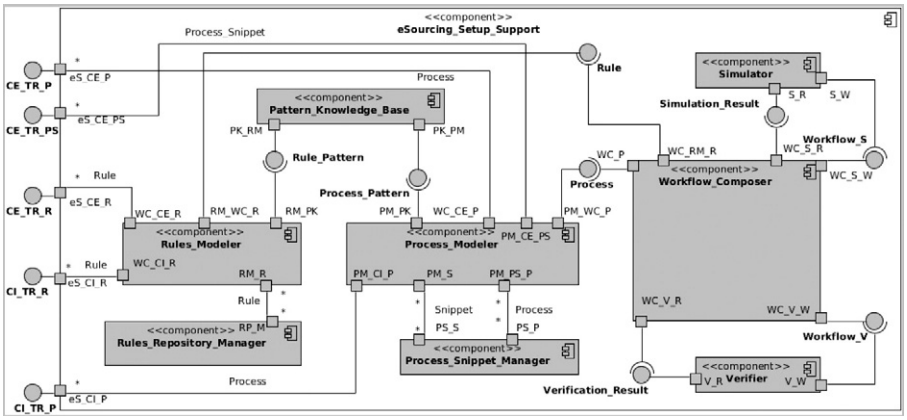

Fig. 9. *eSourcing_Setup_Support* functionality.

The *Rules_Modeler* and the *Process_Modeler* support stem from a *Pattern_Knowledge_Base*. The latter component [78] stores conceptually formulated patterns organized in a taxonomy belonging to different perspectives [6] such as control-flow, data-flow, and exception management. The knowledge base also stores implementations of respective patterns in industry standards such as BPEL [59], BPMN [2], and so on. The *Workflow_Composer* composition uses process snippets or local processes from the *Pattern_Knowledge_Base* that the *Rules_Modeler* and *Process_Modeler* supply. The resulting composition unites the separated perspectives into one business process ready for local verification.

With respect to checks of control flow, the correct termination [105] is important, i.e., soundness [5] violating deadlocks or lack of synchronization. Lack of soundness results in unsatisfied customers, loss of time and money. Secondly, also other perspectives of processes must be correct, e.g., data-flow, resources such as humans or machines, business rules, and so on. Additionally, it is essential to simulate the in-house process of a service consumer and provider processes of an eSourcing configuration. Among other aspects, such a simulation is meaningful for testing how the different perspectives perform together for workflow enactment, e.g., the correct functioning of the Web-service orchestration by processes.

Fig. 10 visualizes a second-level refinement of the *Legacy_Management* component. For using legacy systems to be part of a larger service-based application system [38], loosely coupled services facilitate the establishment of highly dynamic business relations with means of service-oriented computing [84,42]. According to [108], Web-services are self-describing, logical manifestations of heterogeneous physical or software resources that include databases, applications, devices or humans. Processes group and compose Web-services into a set of actions an organization executes. Furthermore, Web-services connect to the internal network of an organization or to the Internet. Uniform Resource Identifiers (URI) identify Web-services and description standards exist such as Web Services Description Language (WSDL) [29], or Representational State Transfer (REST) [47], discovery (Universal Description Discovery and Integration (UDDI) [21]) and communication between Web-services with Simple Object Access Protocol (SOAP) [25] on top of network and transport protocols such as Hypertext Transfer Protocol (HTTP).

In Fig. 10, a *Local_WFMS* and *Local_Rules_Engine* constitute the core components. These components exchange data between each other and are instrumental for coordinating legacy systems. The business rules and processes that the *Local_WFMS* and *Local_Rules_Engine* enact, the *CI_Translator* translates between the internal- and external layer. For enactment, the *Local_WFMS* and *Local_Rules_Engine* use a *Production_Data_Manager*. Furthermore, to coordinate the enactment on an internal- and external layer, the *Local_WFMS* and *Local_Rules_Engine* communicate events, rules, and production data bi-directionally.

Examples for *Local_WFMS* exist from research and industry. A candidate example from research for a local WFMS is the SOA-system termed Yet Another Workflow System (YAWL) [80] that uses a Petri-net based [91,92] business-process modeling notation as a foundation. With such semantic clarity, it is possible to verify processes in advance before enactment for runtime soundness [3], i.e., one can check for control-flow issues such as deadlocks. In YAWL, workflow adaptations are possible through

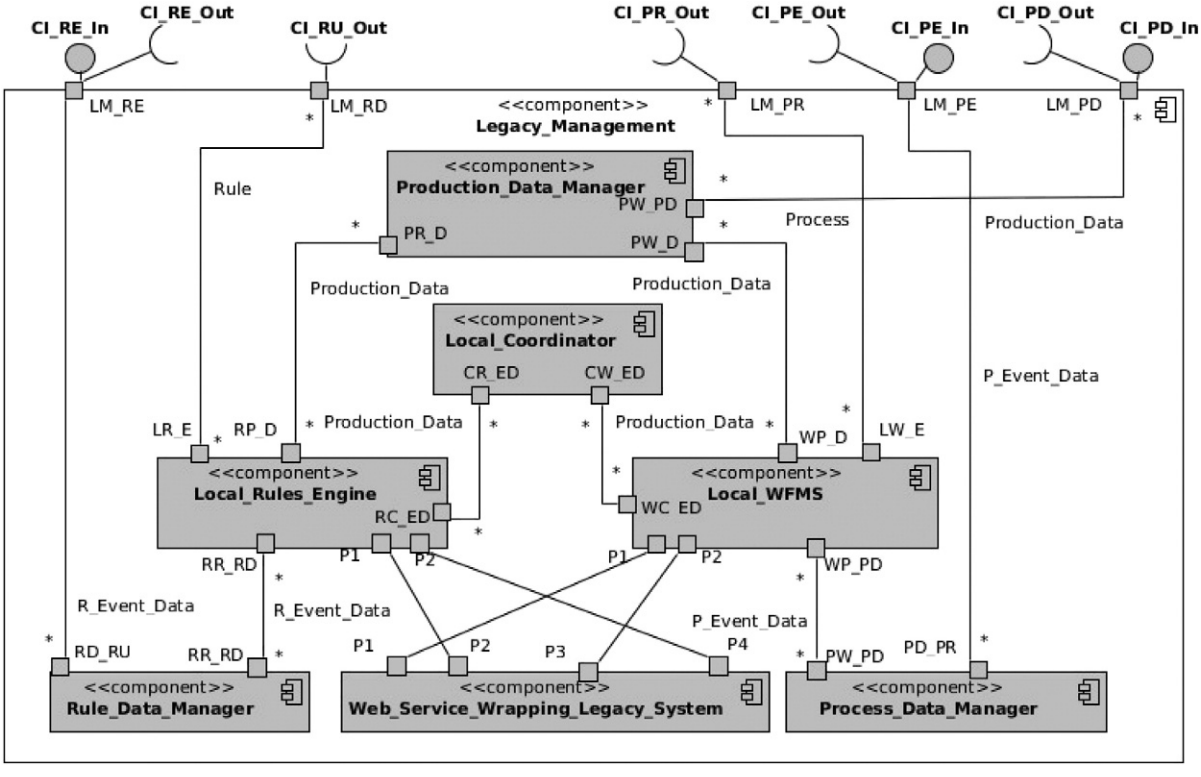

Fig. 10. Connecting to internal systems with *Legacy_Management*.

so-called worklets [8] enactment in a dynamic way based on the evaluation result of Ripple-Down-Rules (RDR) trees. The now industrial AristaFlow system [90] is another example originating from research based on the former ADEPT flow system [89]. With the AristFlow system, it is possible to verify the control flow before enactment. Additionally, the formal notation allows one to change the control flow of business processes at runtime if the enactment context requires that. Finally, examples for *Local_WFMS* from industry are WebSphere [57] and BizTalk [71]. The former WFMS enacts business processes specified in BPEL eBPEL2 that is an industry standard for composing Web-service orchestration. BizTalk enacts business processes formulated in XLANG [102] that is BPEL-compatible. For *Local_Rules_Engine*, again the earlier mentioned ViDRE [74] is a candidate stemming from research while there also exist several Java-based open-source projects such as Drools [87] or Mandarax [58].

4.3. Third refinement level

We continue refining the components from Section 4.2 according to the principles of functional decomposition. The refinement of the *CE_Translator* in Fig. 11 depicts a *CE_Projector* component that transfers data, rules, events, and processes between the conceptual- and external layers. To perform that function, the *CE_Projector* uses a rules database.

Fig. 11 shows several bidirectional arcs to the *CE_Projector*. The *Rules_Modeler* and *Process_Modeler* components exchange contractual rules and process snippets and composed processes via the *CE_Projector* with the contracting client on the external layer. The *Global_Rules_Engine* receives contractual rules from the *CE_Projector* through which rule- and event-data pass down to the *Local_Rules_Engine* on the internal layer.

More detailed exchanges between the *CE_Projector* and components of the *Global_WFMS* are the following. The *Enactment_Engine* receives a *Process* from the service consumer or provider respectively. During enactment, data exchanges take place between the *Enactment_Monitor* and the *Conjointment_Monitor* that we explain below based on depiction Fig. 12. The latter two components exchange events and production data via the *CE_Projector* and the *Workflow/P_Events_Data_Exchanger* down to the local *Enactment_Monitor* and *Conjointment_Monitor* on the internal layer of eSRA.

For realizing the translations of data-, events-, rules-, and process transfers, several technical realizations exist assuming that XML use dominates most transfers. First, for the translation of data, several options exist such as Hadoop [16] that is an open-source project for the processing of large and distributed data sets. The Hadoop Distributed File System provides high-throughput access to application-systems data and Hadoop MapReduce enables reliable parallel processing of largedata sets. Many additional open-source projects on top of Hadoop enhance the data-translation power, e.g., Hive [17], a data warehouse system for easy data summaries and ad-hoc queries in a SQL-like language called HiveQL to analyze large datasets. Also Impala [33] is an open source query engine for massively parallel data processing for Hadoop. Next, the open-source initiative Event Translator [82] comprises four features, namely first, an automated discovery of network services, secondly, receiving events from many different network protocols and also generate events combined with notification management, thirdly, monitoring of service-level agreement assurance with service monitors and finally, performance measures for thresholds or value changes. Assuming that rules- and process specifications use XML, transforming to other rules- and process-notations is possible with

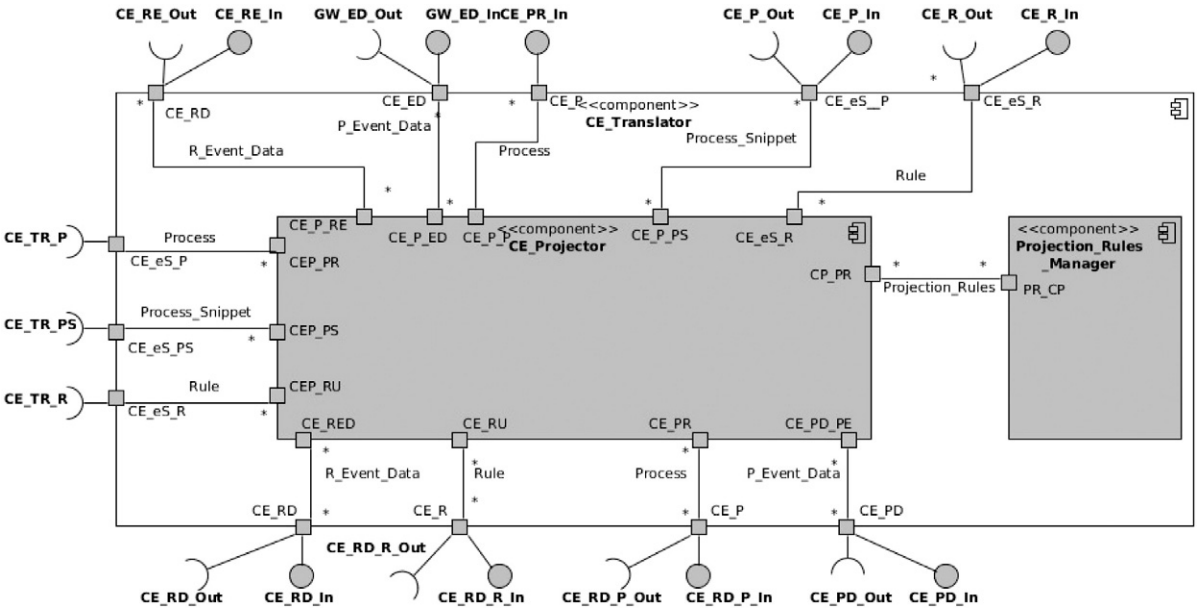

Fig. 11. The *CE_Translator* in detail.

versions of extensible stylesheet language transformations (XSLT) such as SAXON [60], Xalan [14], or Trax [15]. Finally, if content in rules and processes use XML for specifications, there exists an open-source XML content translator [81] called Nikse.

Fig. 12 depicts a refinement of the *Global_WFMS* component of the *eSourcing_Middleware*. It shows an *Enactment_Engine* for the collaborating party's business processes that have the *CE_Translator* as a source. Dedicated databases cater for the retrieval of event- and production data during enactment. In order to support the concept of eSourcing, Fig. 12 shows an *Enactment_Monitor* and *Conjointment_Manager* component. Concretely, the first is responsible for allowing the service consumer to monitor the enactment progress of service provisioning. Both components exchange production- and event data in the domain of the collaborating party via the coordination interface. Conjoinment [77] focuses on the exchange of business information between the domains of the collaborating parties and monitorability covers the way how nodes in the consumer's and provider's processes link to each other. Furthermore, the internal-layer communicates *Production_Data* and *P_Event_Data* via the *CE_Translator* to coordinate local components.

The refinement of the *CE_Translator* in Fig. 13 depicts a similar setup as for the *CE_Translator*. However, the information exchange to neighboring components differs. The *CE_Translator* contains a *CE_Projector* component that projects information between the conceptual- and internal layer. To do so, a *Projection_Rules_Database* exchanges rules with the *CE_Projector* that receives contractual rules from the *Rules_Modeler*, and business processes from the *Process_Modeler*.

The contractual rules are delivered to the local *Global_Rules_Engine* of the internal layer. Furthermore, the business-process specifications are also delivered to the internal layer where a process database stores them until the *Local_WFMS* loads the processes for enactment. To coordinate the *Local_WFMS* and *Local_Rules_Engine* with corresponding components on the external layer, the *CI_Projector* transfers production-, rules-, and events data between the internal- and external layers of eSRA.

The internal-layer refinement of Fig. 14 shows a setup that is comparable to the *Global_WFMS* of Fig. 12. The *Local_WFMS* contains an *Enactment_Engine* that receives business processes. *Production_Data* needed during process enactment is exchanged with the *Production_Data_Database*. *P_Event_Data* exchanges take place with the *Local_Rules_Engine* that carries out contractual rules. Furthermore, the *Enactment_Engine* exchanges data with ports for the orchestration of legacy systems. To coordinate the local enactment progress with the external layer, production data and event data are exchanged with the *Conjoinment_Manager* and the *Enactment_Monitor* respectively. The latter two components exchange events- and production data via the *CI_Translator* with the equally named components located on the external layer.

The *Service_Broker* refinement within the *Trusted_Third_Party* of Fig. 15 reveals a *Service_Library_Manager* that stores business processes of collaborating parties via the template search engine. The latter component exchanges business-process specifications with the *Contracting_Client* of the *eSourcing_Middleware* on the external layer of the collaborating parties. Furthermore, the *Template_Search_Engine* exchanges data with the *Bid_Manger* component of the *Brokering_Service*. The *Notifier* component checks business-process specifications stored in the *Service_Library_Manager* for data about a collaborating party that needs to be informed. If such facts exist, the *Notifier* informs the specified *Contracting_Client* of the respective parties about the submission of a

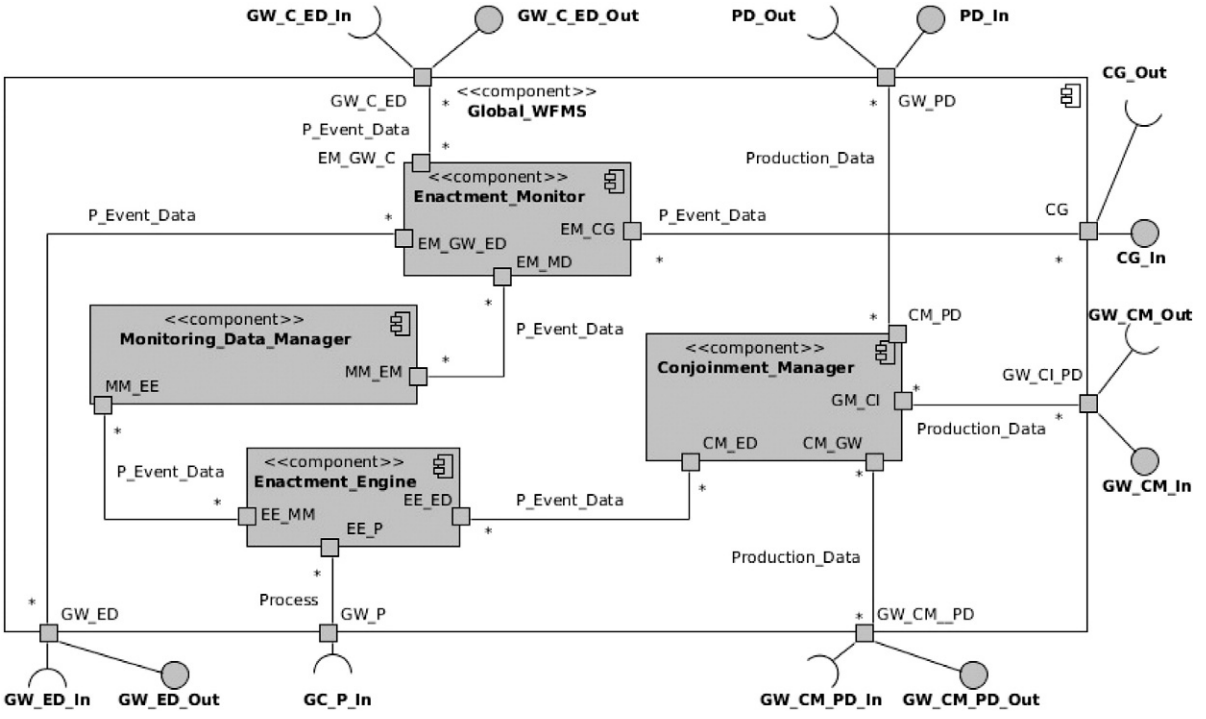

Fig. 12. The *Global_WFMS* in detail.

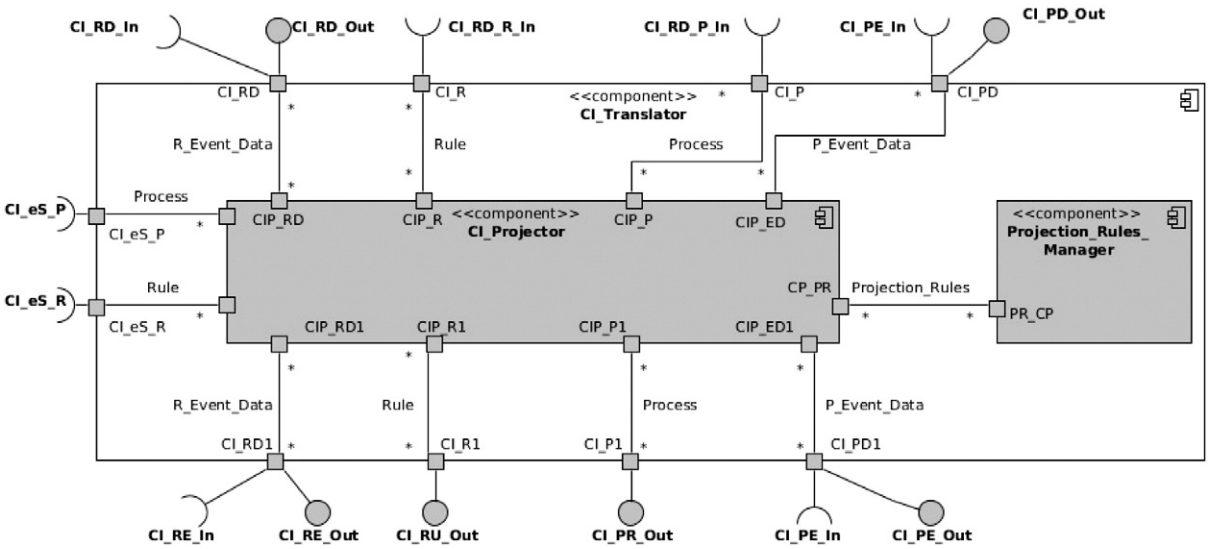

Fig. 13. The *CI_Translator* in detail.

projected consumer request or provider offer. Consequently, informed parties check the stored business-process specifications and either engage in a bidding procedure, or commit to the externalized business process by directly responding with committing a separate process to the *Trusted_Third_Party*. The functionality of an extended version of such a service broker [79] comprises a keyword-based search for services backed by ontologies and an ad-hoc generated mashup of information for background checking to evaluate service quality. Furthermore, automated matching [64] of service-offers with -requests accelerate the collaboration-setup phase.

The refined *Bidding_Service* of the *Trusted_Third_Party* is depicted in Fig. 16. In the *Brokering_Service* component, the contained *Bid_Library_Manager* stores committed bids for retrieval by a *Bid_Manager*. This component communicates with the *Contracting_Client* component that places and retrieves bids from the *Bid_Library_Manager*. As described earlier, the *Bid_Manager* exchanges bid- and service data with the *Template_Search_Engine* component of the *Service_Broker*. Business requests for bids, information, quote, and proposal are part of auctions of differing types. Many auction types exist and we next mention the most common. A so-called English auction [101] focuses on price discovery and is transparent in a sense that bidding buyers know about each other. Potential buyers

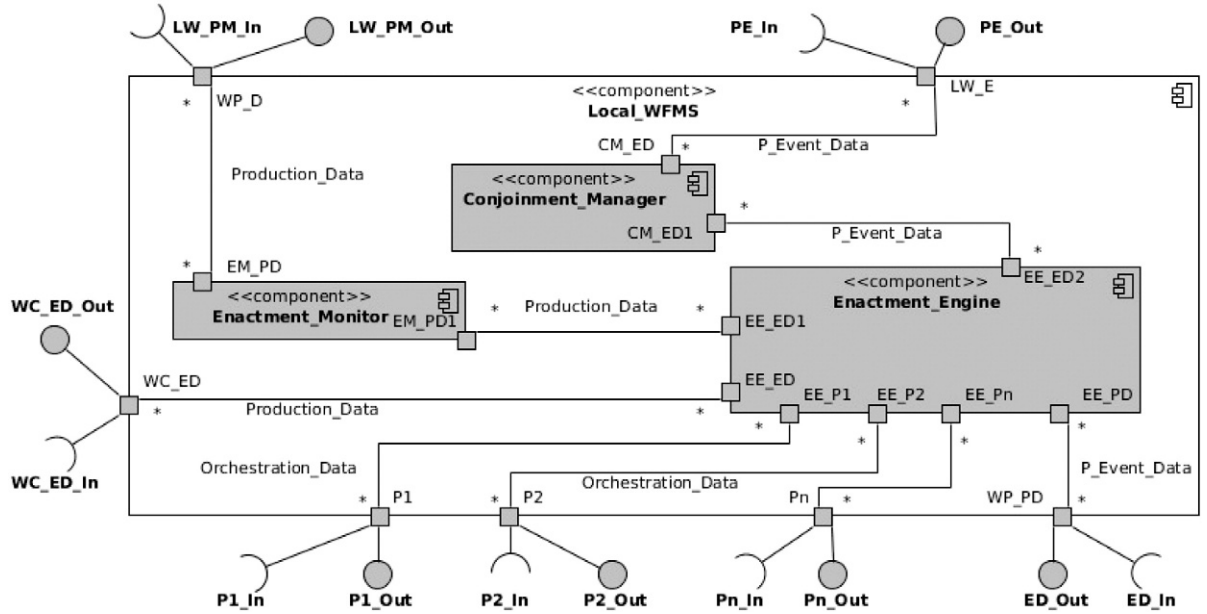

Fig. 14. The *Local_WFMS* in detail.

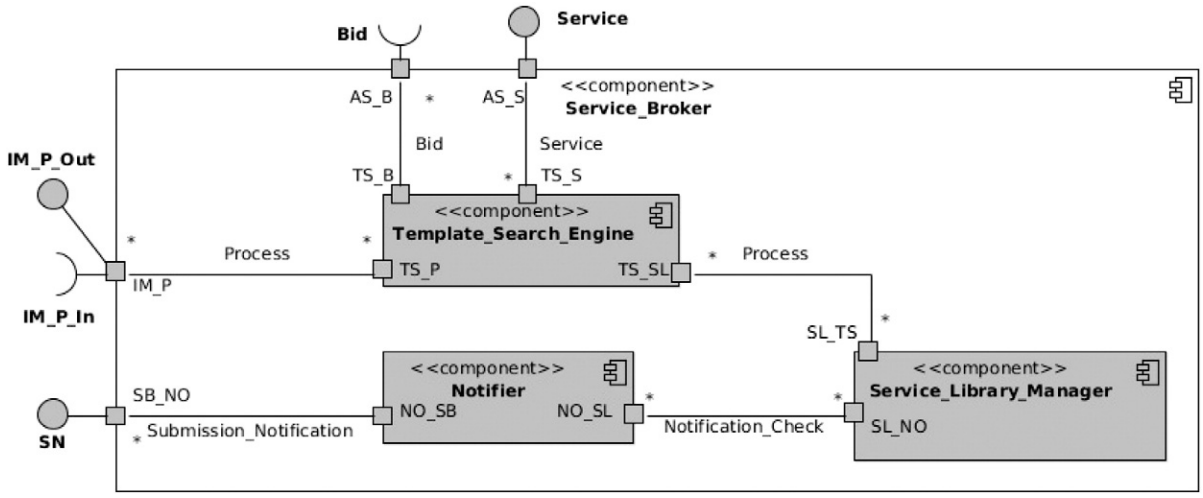

Fig. 15. The *Service_Broker* in detail.

submit progressively increasing bids until nobody is willing to raise the price further. At that time, the auction closes and the highest bidder wins. In an English reverse auction [73] potential suppliers submit progressively decreasing bids, with the lowest bidder winning. In sealed bid auctions [97], buyers bid on an item without knowing the details such as price and quantity of each other's bids. Bidders specify a quantity and a price, and the highest bids receive their quantities until no more units remain. In a sealed bid reverse auction [28], the owner is a buyer and participants are suppliers. Potential suppliers bid to sell the item to the owner, and the lowest bidder or bidders win. After the auction closes to bidders, a selection follows of winning bids based on a set of rules.

The last component of the *Trusted_Third_Party* in Fig. 17 is the *Validator*. The verifier architecture in [75] is suitable for the *Trusted_Third_Party*. In this architecture, the business processes of the collaborating parties are flattened to a P/T-net and consequently verified for correct termination and inheritance relations. In Fig. 17, a *Process_Communicator* component receives a request from the *Contracting_Client* belonging to the domain of a collaborating party to perform a verification of a created eSourcing configuration. The *Process_Communicator* requests the conceptual-layer processes of all collaborating parties and the contractual spheres from the *eSourcing_Middleware*. Next, the collected in-house process, the provider spheres, and the contractual spheres the *Translator* converts into a format the *eSC_To_BW_Mapper* component and *Verifier* can process. The first component delivers the resulting BW-net to a *Collapsor* that creates a net, a component verifies for soundness and projection inheritance. For the latter verification, a comparison follows of the in-house process with the flattened P/T-net. In [76], we give further details about P/T-net details. Finally, with a top-down specification of eSRA based on carefully collected requirements, we must assure in the sequel of the paper that eSRA guides the domain-focused development of standard- and concrete architecture.

5. Evaluation of eSRA

Since eSRA is a reference architecture, it represents a family of B2B-collaboration systems. We first evaluate to which extent eSRA meets the set of requirements stipulated in Section 3, compare eSRA against leading commercial reference architectures and finally, conduct a control-survey with experts from academia and industry. The former evaluation answers the question of

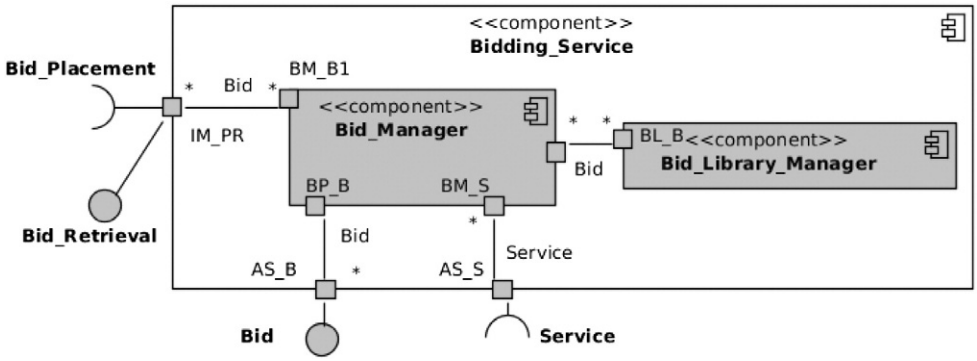

Fig. 16. The *Bidding_Service* in detail.

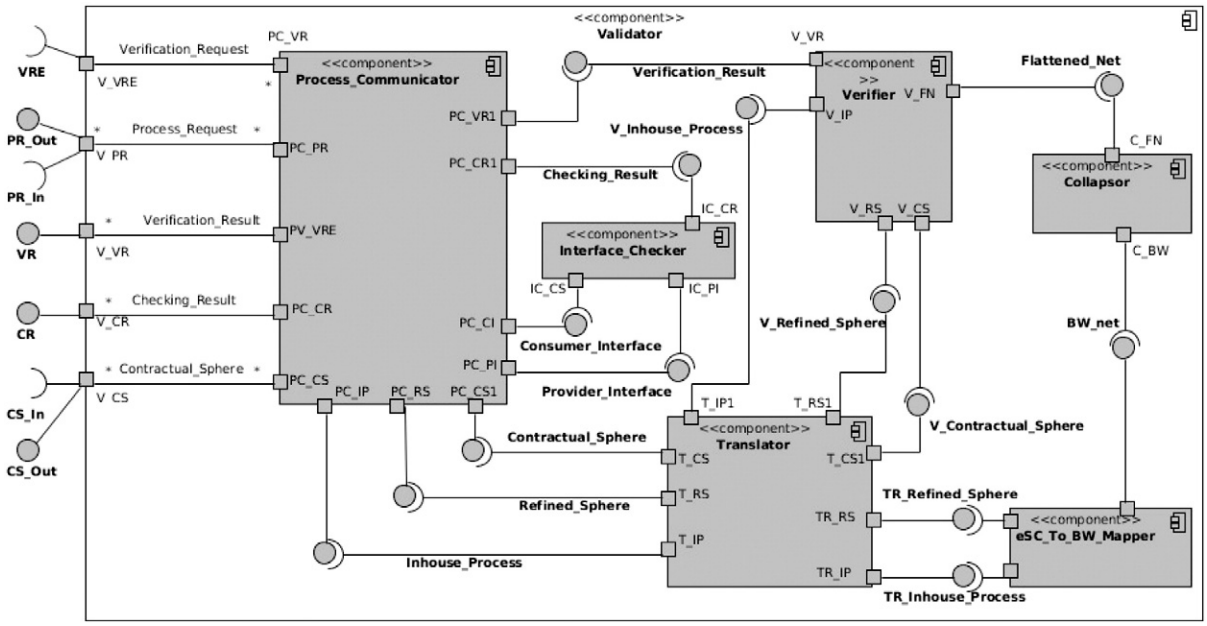

Fig. 17. The *Validator* component in detail.

whether eSRA does provide the required features, while the latter evaluation demonstrates eSRA as an improvement over the commercial reference architectures.

We performed the evaluation in two workshops with participants from industry and academia. During the first evaluation (see Appendix A), two of the evaluators' backgrounds were software engineering and they were researchers and lecturers with an interest in software architecture. Of the two other reviewers, one was a professor with a background in distributed systems for collaborative and interoperable computing and the remaining reviewer had a similar competency background as a member of a research company that carries out applied research with industry. We met in four workshop sessions and collaboratively went through all ATAM-steps as explained below.

For the second evaluation workshop, the number of participants was approximately triple the amount compared to the first time. The participants were mostly from industry but also from public ICT organizations and university research departments for software engineering and socio-technical agent engineering. The workshop's morning comprised an introduction into the ATAM methods and a presentation of the context for eSRA and the reference architecture itself. In the afternoon, the evaluation commenced in an interactive way as prescribed by the ATAM method. We refer the reader to Appendix B for details.

For eSRA, we consider several means to achieve an evaluation. In Section 5.1, we discuss a method for architecture evaluation and the completeness check of Section 5.2 shows that the specified components of eSRA cover all functional requirements. Next, in Sections 5.3–5.8 all non-functional requirements we separately address either with the results of applying the chosen modified scenario-based evaluation method, or we apply reasoning techniques such as exploring existing eSourcing systems from industry and research and a control-survey with members of industry and academia.

5.1. Scenario-based software architecture evaluation methods

For the specification and subsequent evaluation of eSRA, we use a modified version of a scenario-based method, namely, the *A*rchitecture *T*rade-Off *A*nalysis *M*ethod (ATAM) [32,61]. The following discussion gives an overview for the adopted approach.

The *A*rchitecture *T*rade-Off *A*nalysis *M*ethod (ATAM) [32,61] explores quality attributes on the interdependencies with evaluations in several workshop sessions. The initial workshop involves a presentation of the reference architecture together with its business aspects to a group of three to six expert evaluators who then elicit, concretize and prioritize the driving quality attribute requirements. Establishing a quality–attribute–utility tree [61] supports the workshop sessions to the level of scenario-creation, annotated with stimuli for architecture components and their responses, and prioritized scenario-listing. In a follow-up workshop, nine to fifteen related evaluators and project-related personnel evaluate the results from the first workshop with analyzing and specifying in detail the chosen scenarios. During this step, we identify the architectural risks, sensitivity points, and trade-off points. *Risks* are architectural decisions that have not yet been made. *Sensitivity points* are properties of one or more components in the reference architecture that are critical for achieving a particular quality attribute response. *Trade-off points* are properties that affect more than one attribute and that are sensitivity points for more than one attribute.

Note that it is important in ATAM to cover the quality attributes where possible with so-called attribute-based architecture styles [63] (ABAS). Architectural styles [26,96] describe component types and their topology, the design pattern of data and

control interaction among the components, and informally describe their benefits and drawbacks. Architectural styles differentiate classes of designs by offering experiential evidence of how each class has been used, along with qualitative reasoning to explain why each class has certain properties. We define a design pattern as conceptually formulated knowledge that is technology independent. A design pattern for software architecture [26] describes a particular recurring design problem that arises in specific design contexts and presents a well-proven generic scheme for its solution. Just as with architectural styles, in eSRA, we use design patterns for adhering to non-functional requirements.

We use an adjusted version of ATAM to specifically target the evaluation, e.g., emphasis on the establishment of a detailed requirement-attribute hierarchy, selection and communication of stakeholders, selection and usage of scenarios. The results of the ATAM-evaluation of eSRA we present below with summarizing the results for the functional attributes and we continue with non-functional requirements in the sequel. The results of the ATAM-evaluation show that usability is the most critical quality attribute of the attribute tree. Concretely, users must receive intuitively comprehensible verification results while setting up B2B-collaborations. That also comprises real-time setup-guidance with instantaneous notification of design faults and recommendations for corrections, including during run-time. The evaluation also shows that an instantaneous verification- and recommendation support must ensure there is no disclosure of internal details from the collaborating counter-party. Finally, an ad-hoc evaluation of potential collaborating counter-parties must allow for an estimation of trustworthiness and reputability.

5.2. Functional completeness check

In Table 1, we show the results of a paper-based evaluation for functional completeness. If all refining elements inside a respective component are necessary for requirement satisfaction then the table omits them. The eSRA-components in Table 1 are in rows together with supported functional requirement from Section 3.1 and the architectural refinement level in which the components reside. The alphabet in the left column of Table 1 corresponds to the alphabet of the functional requirements in Section 3.1 and it also corresponds to the paragraph alphabet below. Next, we explain how the eSRA-components cover the functional requirements.

5.2.1. Coverage of functional requirements by eSRA-components

With respect to supporting the conceptual formulation of business processes and their accompanying rules, the highest refinement of eSRA comprises the *eSourcing_Setup_Support* component on the conceptual layer. On the second refinement level of the *eSourcing_Setup_Support*, the *Pattern_Knowledge_Base* supplies building blocks for the conceptual formulation of business processes that the *Rules_Modeler*, *Process_Modeler*, *Workflow_Composer* create with the supporting databases.

The brokering capability of projected business processes in eSRA the *Trusted_Third_Party* realizes. On the second refinement level, the *Service_Broker* and its connected *Process_Snipper_Manager* allow collaborating parties to submit service requests and service offers. Contained in the *Service_Broker*, the *Template_Search_Engine* allows one to search the *Service_Library_Manager*. Furthermore, the *Notifier* component actively informs a party if she is part of the specification in a service request or offer as a preferred collaboration candidate.

The bidding capability for projected business processes also the *Trusted_Third_Party* comprises with the *Brokering_Service* component. On the third refinement level of eSRA, the *Brokering_Service* allows the placement and retrieval of bids from the *Bid_Library_Manager*.

Table 1
Coverage of functional requirements by eSRA-components.

Functional requirement	Refinement level	eSRA component
a	1	eSourcing setup support
b	1	*Translator*
	2	Cl projector
c	1	*Translator*
	2	CE projector
d	1	Trusted third party
	2	Service broker
e	1	*Trusted_Third_Party*
	2	Bidding service
f	1	eSourcing middleware, trusted third party
	2	Contracting client, *CE-translator*, coordination interface
g	1	*eSourcing Setup_Support*, *Trusted_Third Party*
	2	*Validator*, *Verifiers* (external and internal), *CE_Translator*, *Cl_Translator*
h	1	*eSourcing_Setup_Support*
	2	*Simulator*, *CE_Translator*, *Cl_Translator*
i	1	*eSourcing_Middleware*, *Trusted_Third_Pary*
j	1	*Legacy_Management*
k	1	*eSourcing_Setup_Support*, *Legacy_Management*, *eSourcing_Middleware*, *Translator*

The setup support of an electronic enterprise collaboration involves the *eSourcing_Middleware* and the *Trusted_Third_Party*. On the second refinement level, the *Contracting_Client* together with the *Coordination_Interface* forms a platform to negotiate the setup. We refer to [13] for details about a contracting-client refinement. From the *CE_Translator* component, the contained *CE_Projector* ensures a projection of conceptual-layer processes to the contracting client.

For avoiding the direct linking of business processes and legacy systems, eSRA provides components on the external-, conceptual-, and internal layer. Hence, on the first refinement level of eSRA, the *eSourcing_Middleware* and *Lagacy_Management* cover this functionality requirement. Avoiding direct process linking involves from the second eSRA-level the *Global_Rules_Engine*, *Global_WFMS*, and the *Coordination_Interface* of the *eSourcing_Middleware* while legacy management requires the *Local_WFMS* and *Local_Rules_Engine*. The coordination interface interacts with the *Global_Rules_Engine* and *Global_WFMS* of collaborating parties.

The verification and simulation of electronic enterprise collaboration eSRA caters for with the *eSourcing_Setup_Support* and the *Trusted_Third_Party*. In the first case, internal *Verifier*- and *Simulator*-components allow a local checking of conceptual business processes. When these business processes match inter-organizationally, the *Verifier* of the *Trusted_Third_Party* allows a checking of the overall process without forcing the collaborating parties into revealing business secrets to each other.

For the enactment of a ready collaboration configuration, eSRA provides components from the *eSouring_Setup__Support*, *Lagacy_Management*, *eSourcing_Middleware*, and *Translator*. Hence, from the *eSouring_Setup__Support*, the components termed *Global_Rules_Engine*, *Global_WFMS*, and *Coordination_Interface* support enactment. Furthermore, the *CE_Translator*, *CI_Translator*, and both data exchangers of the *Translator* ensure a functioning link between the external- and internal layer during enactment. From the *Legacy_Management*, the *Local_WFMS* and the *Local_Rules_Engine* components orchestrate the service-wrapped legacy systems.

5.3. Applicability — eSRA comparison with existing leading reference architectures

By comparing eSRA with the functionalities of existing leading commercial applications in the domain of eSourcing, we show the applicability of the reference architecture. The chosen commercial applications are the biggest and most complete systems available, namely the Sourcing platform from SAP [95] and the Sourcing platform from ORACLE [83]. The third commercial application is the SmartSource platform from IASTA [56] that is less complete as the eSRA-comparison shows in the sequel. As the purpose of eSRA is to achieve for system designers a quick understanding of other B2B-collaboration systems in a paper-based form on the one hand, or also to provide guidelines for creating standard- and concrete architectures (see Fig. 4) We use system documentation about the chosen commercial applications for checking eSRA-applicability.

In a SAP context, Sourcing is a Web-based process used to formally select one or several suppliers that provide services. SAP Sourcing encompasses the online process of purchase-expenditure analysis, supplier identification, negotiation events such as requests for Sourcing proposals, or reverse auctioning in which service consumers collect and evaluate price- and bid information of suppliers, supplier-record management, and contract management.

Secondly, we consider the Sourcing application of ORACLE. Sourcing supports online collaboration and negotiation for procurement professionals, business experts, and suppliers from multiple organizations with functionality to electronically exchange information. Furthermore, Sourcing supports bidding and auction processes, and the creation and implementation agreements that reduce sourcing cycle time and create a complete audit trail of supplier commitments.

Finally, the SmartSource platform from IASTA enables companies to communicate with buyers, suppliers and other stakeholders through integrating components for sourcing products and services using request for information, proposal or quotation; survey or auction functionality. With the SmartSource platform, it is possible to perform quick optimization and analyze different award scenarios; track, follow and manage sourcing contracts, and manage supplier relationships.

In Table 2, the cells show to which degree functionalities present in eSRA are part of the chosen commercial applications. The rows list the results for respective commercial applications while the letters in the columns correspond to Section 3.1 that lists the functional requirements for eSRA. For the cells, a plus sign indicates full functionality support while a minus sign indicates no support and both signs in a cell indicate partial support. Next follows an evaluation discussion for the respective commercial applications.

5.3.1. SAP Sourcing and eSRA

In comparison [95], the eSourcing concept (see Section 2.2) matches with SAP Sourcing while the first stresses the importance of the structural harmonization of inter-organizational business processes. eSourcing also puts special emphasis on hiding business internals from the collaboration counterpart that is equally not stressed by SmartSource.

Table 2
Functional-requirementss adherence of commercial applications.

	Functional requirements										
	a	b	c	d	e	f	9	h	1	1	k
SAP Sourcing	+	+	−	±	±		−	−	−	−	±
ORACLE Sourcing	+	+	−	+	+		−	−	−	±	+
IASTA SmartSourcing	−	−	−	±	±		−	−	−	−	−

Studying documentation,[1] SAP Sourcing supports with special components the conceptual formulation of business processes and business rules and there is also a broker present that is not an independent trusted third party but located in the domain of the service consumer. Projection of process details is not supported as only service-proposal requests are exposed. Service providers bid via the broker for proposal requests. SAP Sourcing supports mapping as the system orchestrates internal SAP-legacy systems. However, there is no indication of business-process details exposure to the broker. Bidding abilities exist in SAP Sourcing for service providers and through reverse auctioning, the service consumer checks the price and bid data to pick the best offer. There exists also negotiation support for contract creation, distribution, and compliance management on a centralized platform. The contracts are text based and do not comprise business-process details like in eSourcing. Hence, SAP does not provide automated verifications of contracts.

With respect to distributing and shielding business processes and legacy systems, SAP Sourcing caters for separation layers. However, the text-based contract does not permit an automated extraction of external-layer business-collaboration choreography that is enactable with middleware. The conceptual layer comprises orchestrating business processes, constraints expressed as rules and a layer that shields the legacy-backend systems onto that the business processes are mapped. There is no indication that SAP Sourcing provides for simulation or verification components of collaboration configurations between service consumers and service providers. For the enactment of ready collaboration configurations, SAP Sourcing uses the NetWeaver[2] application located in the domain of the service consumer that integrates with the service provider. NetWeaver has no component as in eSourcing for internal legacy-system enactment. However, as a workaround, a local workflow management system may serve as a legacy application in NetWeaver for enacting processes.

5.3.2. ORACLE Sourcing and eSRA

The evaluation [83] reveals a conformance with the eSourcing definition put forward earlier. Besides relying on powerful database support such as from ORACLE 12c,[3] versatility support of ORACLE Sourcing stems from the openness to a vast spectrum of other ORACLE applications from the Oracle Advanced Procurement suite.[4]

With the openness of the ORACLE suite it is possible to design conceptually processes in a language like BPMN [110] with support for automatic translation into enactable BPEL-code. For projecting and mapping, the same holds as for SAP. Brokering capability ORACLE supports as a component exists between collaborating parties for requesting proposals and quotations. However, there is no explicit projection of process views. Consequently, bidding capabilities are fully present with functionality for power-, proxy-, and surrogate bidding.

Negotiation support is not business-process based but the ORACLE suites try to reduce cycle time with automating the structuring of consolidation requirements, amendments, responses and notifications. On the other hand, no validation functionality exists as a default part of ORACLE Sourcing for shielding distributed legacy systems, Sourcing provides an external collaboration layer for negotiated contract establishment backed by an internal business-process design. The process-supporting legacy systems within the domains of collaborating parties support process activities wrapped into Web-services. Finally, the ORACLE SaaS[5] suite backs Sourcing, a fully developed enactment environment exists.

5.3.3. IASTA SmartSource and eSRA

Compared [56] to eSourcing, Table 2 reveals that IASTA's SmartSource platform provides support for a subset of functional requirements. The explanation for that is the lack of business-process and rule support as SmartSource is a platform that caters for accumulating and aggregating data of suppliers to support external-layer collaboration setup. Consequently, no database stores exist that manage business processes and rules. The evaluation result shows a lack of support for conceptually formulating business processes and rules, mapping and projecting business processes between collaboration layers, distribution and shielding of business processes and legacy systems, and the enactment of ready collaboration configurations.

Partial requirement support of the SmartSource platform exists for brokering, bidding and negotiation. However, full support is impossible because of lacking support of inter-organizational business process and rules design and management. Instead, brokering, bidding and negotiation is driven by requests for information, proposal, tender and quotation. Thus, the SmartSource platform from IASTA requires a combination of traditional ERP-systems to satisfy more functional requirements than brokering, bidding and negotiation.

5.4. Usability of eSRA

For performing the scenario-based evaluation method in Section 5.1, we present eSRA in workshops to experts from industry and academia. Communicating the principles of eSourcing and explain the purpose of eSRA-components on respective refinement levels is simple. The specification detail of eSRA is satisfactory and the terminology comprehensible. Furthermore, as Section 5.2 shows, eSRA is a means for enabling a quick checking of existing eSourcing system features in a paper-based form. This is not

[1] http://www.sap.com/solution/lob/procurement/software/sourcing/index.html.
[2] http://scn.sap.com/community/netweaver.
[3] http://www.oracle.com/us/products/database/.
[4] http://www.oracle.com/us/products/applications/ebusiness/procurement/.
[5] http://www.oracle.com/us/technologies/saas.

possible without quickly understanding the underlying concept, naming conventions, notations and structure of eSRA and being able to compare those to other systems.

5.5. Feasibility of eSRA

The first feasibility property of *buildability* is provable with the existence of application systems as in Section 5.3. One eSRA-component is an exception, namely, the contracting client in Fig. 6 that is partially realized. In [13], the reader finds detailed explanations about the contracting client. A scientific prototype implementation [49] of an eSRA-compliant proof-of-concept application exists from the CrossWork-project.

The second feasibility property of *coherent design and clear structure* in eSRA we achieve with employing architectural styles and design patterns that connect on the one hand the reference architecture to the non-functional requirements and on the other hand, the styles and patterns facilitate a quick design of an eSRA-adhering system instantiation.

In eSRA, a *layering style* [20,26], i.e., we use three concern-separating layers for the domains of a service consumer and service provider because it helps to structure the components of eSRA into groups at a particular level of abstraction. For eSRA, these abstraction levels are the external collaboration layer, the conceptual layer, and the internal layer for managing and orchestrating legacy systems. In eSRA, a layer only communicates with an adjacent layer while a *whole-part pattern* [26] complements the layering style to aggregate the parts of a business collaboration. Finally, on the conceptual layer of eSRA, a *pipes-and-filters pattern* [26] facilitates establishing communication channels between the external and the internal layer via the conceptual layer. The latter design pattern structures the processing streams of data while filter components encapsulate each processing step. Hence, data passes through pipes between adjacent filters from the external layer to the internal layer and vice versa.

5.6. High automation in eSRA

Although it is desirable for efficiency and effectiveness to have a total automation support for eSourcing, reality shows this is not realistic. Typically, an eSourcing scenario is semi-automated by an application system to a point where the users retain control they consider vital for success in real-life business collaborations. Although, such human control requires graphical user interfaces (GUI), eSRA does not explicitly specify them for the reason that the degree of desired human control versus automation degree for an eSourcing-compliant system may vary depending on the business-collaboration domain and situation. The following areas of user involvement may be desirable depending on the context, e.g., for modeling business processes and rules, negotiating a consensus in the contracting phase of an eSourcing collaboration, reasoning about verification results, performing decisions during the enactment phase such as what form of exception handling and compensation to carry out when errors occur, and so on.

5.7. Modifiability, integrability, interoperability, security, flexibility, portability, performance and scalability of eSRA

To check the non-functional requirements of concern, we employ the modified version of ATAM-evaluation method (see Section 5.1). We refer the reader to the appendices of this paper for details about the conducted scenario-based evaluation. The remainder of this subsection first lists eSRA-improvements, followed by results of the structured eSRA-analysis where we relate in a summary architectural decisions and qualities, and the required qualities and conclusions on eSRA-strengths and -weaknesses.

5.7.1. Qualities and architectural decisions

The investigated architectural styles and patterns present in eSRA connect on the one hand, the reference architecture to the non-functional requirements and on the other hand, the patterns help to quickly design an eSRA-adhering business-collaboration systems. However, it is not possible to check all non-functional requirements with the style- and pattern discussion. The adherence of eSRA to the remaining set of non-functional requirements we check in the sequel by evaluating eSRA-adherence of existing business-collaboration systems from research and industry.

Architectural styles [26,96] describe component types and their topology, the design pattern of data and control interaction among the components, and informally describe their benefits and drawbacks. Architectural styles differentiate classes of designs by offering experiential evidence for how to use each class, along with qualitative reasoning to explain why each class has certain properties. A design pattern for software architecture [26] describes a particular recurring design problem that arises in specific design contexts and presents a well-proven generic scheme for its solution. Just as with architectural styles, in eSRA, we use design patterns for adhering to non-functional requirements.

Table 3 gives an overview of the non-functional eSRA-requirements. The top row lists the architectural styles and patterns that are present in eSRA. In Table 3, a plus sign means a particular style or pattern supports a non-functional requirement. A minus sign means that a style or pattern does not support a requirement. Both signs we assign based on the specifications about architectural styles and pattern literature [26,96] that indicate what non-functional requirements they cover.

In eSRA, a *layering style* [20,26] for the domains of a service consumer and service provider structures components into groups at a particular level of abstraction. For eSRA, these abstraction layers are the external collaboration layer, the conceptual layer, and the internal layer to manage and orchestrate legacy systems. In eSRA, the layers assure that there is only communication with each adjacent layer. By using a layering style, eSRA supports the requirements of *modifiability*, *portability*, and *integrability*. Most

importantly *interoperability* prevents collaborating parties from having to link their legacy systems directly. As a trade-off, a layering style results in extra communication overhead that does not support *performance*.

The trusted third party of eSRA uses a *publish/subscribe style* [63] in which the data producers are publishers and the consumers are subscribers. When a publisher submits new data, all subscribers receive a notification and automatically have data delivered. In the trusted third party component, the notifier forms the central component of a star topology where the publishers and subscribers are the leaves. The advantage of this style in a multi-party collaboration environment with large numbers of potential service consumers and service providers, is enhanced system *performance* because of reduced communication overhead and an enhancement of *flexibility* and *integrability* of eSRA.

For the remaining way of data management, eSRA employs an *abstract-data-repository style* [63]. This style keeps the producers and consumers of shared data from having knowledge of each other's existence and the details of their implementations. In the case of eSRA, using a layering style and by interposing an intermediary protocol between the producer and consumers of shared data items also realizes the abstract data repository style. Note, that all domains of collaborating parties replicate the external layer of eSRA and are only accessible through the coordination-interface component. Furthermore, the abstract data repository style requires an abstract interface to the data repository that further reduces the coupling between the data producers and consumers. With this architecture style, eSRA supports the requirements *flexibility*, *modifiability*, *integrability*, and *portability*.

A *whole-part pattern* [26] aggregates the parts of a business collaboration. In eSRA, dedicated components exist on the external- and internal layer in the form of the global and local WFMS and rules engine. Additionally, the conceptual layer differentiates modeling support for business processes and business rules. The global and local WFMS comprise components termed enactment monitor, conjoinment manager, and enactment manager. By using the whole-part pattern, eSRA supports *modifiability*, *flexibility* and *integrability* through modularization while this is detrimental to *performance*.

The *broker pattern* [26] in eSRA the trusted-third-party component realizes between the domains of collaborating parties. A broker is a separate component that interacts with the remainder of the architecture. Its purpose is the redirection and bundling of communicating with many collaborating parties. Hence, since the broker pattern stops parties from having to find, contact and investigate every potential collaborating party separately, it affects *performance* positively. Instead, the broker pattern centrally offers searchable information about collaborating parties and detects the best candidate for service consumption or service provision. In eSRA, collaborating parties may use the trusted third party for submitting service requests and service offers that are centrally searchable and for which it is possible to place bids. Using the broker pattern in eSRA enhances *scalability* as it simplifies the task for service providers and service consumers to collaborate with each other. Most importantly, placing a decoupled component between business domains with which they must register and that enhances the trust between collaborating parties, supports the *security* requirement.

The already mentioned, coordination-interface component on the external layer of eSRA realizes the *façade pattern* [48]. That way, a unified interface offers to a collaborating counterpart access to a set of interfaces of a subsystem, namely the replicated components of the external layer. Hence, this supports the *interoperability* between business parties and enhances the *security* in a collaboration as it shields the legacy systems behind the façade of the coordination interface.

On the conceptual layer of eSRA, a *pipes-and-filters pattern* [26] facilitates establishing communication channels between the external- and internal layer via the conceptual layer. This pattern provides a structure for processing streams of data while filter components encapsulate each processing step. Hence, data passes through pipes between adjacent filters from the external layer to the internal layer and vice versa. With the pipe- and filter pattern, eSRA supports flexibility, *modifiability*, *integrability*, and *portability*. However, pipes and filters for data passing have a negative effect on collaboration *performance*.

5.7.2. Sensitivity, tradeoff points and risks

The detected risks, sensitivity and tradeoff points that result from the eSRA evaluation, for usability focus on supporting the user during the setup and enactment phase of an electronic business collaboration. Simultaneously it is important that business internals remain opaque from the counter-party. The same visibility problem exists for components that verify different perspectives of a business collaboration.

Table 3
Coverage of non-functional requirements by architectural styles and patterns.

			Styles			Patterns			
	Category	Quality	Layering	Publish/subscribe	Abstract-data-repository	Whole–part	Broker	Facade	Pipes-and-filters
System	Design time	Modifiability	+		+	+			+
		Portability	+		+				+
		Integrability	+		+	+			+
	Runtime	Interoperability	+					+	
		Performance	−	+		−	+		−
		Security					+	+	
		Flexibility		+	+	+			+
Architecture		Scalability					+		

For trust management, the risks and sensitivity points revealed during the first evaluation (see Appendix A) reveal a lack in the initial eSRA-specification [75] of components for collecting trust related information about collaboration parties that also include information about reputation and conflict resolution. Fig. 6 of the eSRA specification depicts these additional components in the contracting client and the trusted third party. The components represent a minimum in eSRA for covering trust management [94] for which more mechanisms exist. However, for a complete trust-management coverage it is essential to also consider the way how collaborating parties agree on forming a temporary relationship in which the behavior rules and constraints are set. Such a relationship between collaboration parties has a lifecycle with the stages of an initiation, a set of stages characterized by events such as a party being eliminated or leaving and to be replaced by alternative parties, changes of behavior rules and constraints, and so on. Eventually, the temporary business relationship reaches the end of its lifecycle when there is no further need to carry out new eSourcing transactions.

With respect to tradeoff points, the first ATAM-evaluation concludes that during the development of eSRA-complying systems, primarily the relationship between the performance, usability and modifiability matter. Performance is important because users need an acceptable response time in an electronic business collaboration. However, elaborate components that ensure the usability of an eSRA-compliant system, negatively affect performance. Also modifiability ensured by patterns as a layer architecture, whole-part decomposition, pipes and filters, or abstract data repositories, results in an overhead that is detrimental for performance.

5.7.3. Evaluation results

Out of the four chosen highest ranked scenarios the first evaluation yield in the utility tree (see Appendix A), three are related to the usability and one to the security requirement of eSRA-complying systems. The amount of chosen scenarios for usability shows the importance for adoption of eSRA-compliant systems by users, i.e., usability is essential for setting up and enacting B2B-collaborations. Concretely, the utility tree in Appendix A shows that usability scenarios comprise the sub-factor error avoidance, error handling, and verification. Their importance stems from the need to check B2B-collaborations for soundness before enactment and to have component-support available during enactment to avoid unsatisfied customers, loss of time and money, and so on.

The fourth highly ranked scenario focuses on security and addresses specifically the sub-factor of security termed trust management. In B2B-collaboration, this sub-factor is important because a party desires to estimate in such anonymous markets how trustworthy a potential collaboration party is. Hence, it is important to have mechanisms and components available that on the one hand, permit an evaluation of potential business partners before engaging in a collaboration, and on the other hand, keep track of the performance of individual business partners during setup and enactment time.

The second evaluation of eSRA (see Appendix B) does not result in further change requests to the architecture specification. The highest ranked scenario of collaboration-configuration verification and -simulation is in line with the result from the first evaluation although now listed as a performance issue. The second evaluation yields different sensitivity-, tradeoff points, risks and non-risk (see Appendix). The risks suggest that the use of socio-technical software agents [100] in system instantiation are a means for quickly matching collaborating parties and also for supporting the setup of different collaboration directions that the collaboration lifecycle in [65] supports.

The sensitivity points show that brokering is essential and also important for availability to enable the setup phase. Changing collaboration directions requires additional brokering involvement that is sensitive to availability again. Failing databases for, e.g., stores of prepared business rules and -processes, are sensitive to modifiability. Large collaboration configurations may pose a challenge to performance during verification and simulation. Finally, elaborate background checking of counterparties, services, persons is sensitive to performance and availability when employing very large data sets.

The tradeoff points include brokering in that quick matching is perceived by the workshop participants as detrimental to security when less diligent background checks happen. Modifying large numbers of business rules is detrimental to performance as tools must verify for conflicts with respect to other rules and processes. Finally, the non-risks of the second eSRA-evaluation confirm specification maturity and we refer to the Appendix B for further details.

5.8. eSRA-improvements after evaluation

The evaluation of the initial eSRA-specification [75] shows that the highest level (see Fig. 5) is sound. However, the ATAM-evaluation reveals a need to adjust the second refinement level (see Section 4.2). The risks and sensitivity points lead to an extension with components for collecting information about the trustworthiness and reputation of collaboration parties. Concretely, we introduce additional components for reputation and identity management (see Fig. 6) for covering trust management [94]. Additionally, we introduce a new *Coordinator* component between the *Global_Rules_Engine* and the *Global_WFMS* (see Fig. 6) and the *Local_Rules_Engine* and *Local_WFMS* (see Fig. 10). The third refinement level of the initial eSRA-specification remains unaffected by results of the evaluation.

For implementing eSRA-compliant application systems, the risks, sensitivity- and tradeoff points resulting from the eSRA-evaluation, reveal that the development of suitable graphical user interfaces (GUI) are essential to facilitate the adoption process by collaborating companies. Furthermore, verification and simulation components must ensure the soundness of an eSourcing-configuration before the enactment phase. It is important that such verification- and simulation components are not computationally expensive to the point where it affects the performance of eSRA-compliant system.

6. Related work

Several research projects have investigated inter-organizational business processes. In the WISE project [9,66], a software platform for process-based B2B electronic commerce focuses on supporting a network of small and medium-sized enterprises. WISE employs a central workflow engine to control cross-organizational virtual business processes. In WISE, a virtual business process consists of a number of linked black-box services in a workflow process [9]. An organization offers a service that is part of a WWW catalog of business processes under the control of local workflow-management systems. Although the WISE project succeeds in orchestrating workflows of different collaborating organizations, it does not cater for a flexible degree of mutual visibility of business-process details as eSourcing does. Hence, eSRA is specified in accordance with the requirement of catering for flexible visibility degrees. Differently to WISE, eSRA includes components for collaborating parties to negotiate varying degrees of observability during the enactment phase.

In the CrossFlow project [93], dynamic outsourcing realizes the formation of virtual enterprises. While CrossFlow relies on cooperating pairs of autonomous workflow systems with a peer-to-peer relation, a service matchmaker fits together service offerings and service requests. The resulting electronic contract establishes dynamically a service enactment infrastructure [54] that employs workflow technology. CrossFlow has an external layer that spans across organizational domains in which the process is part of a contract specification. The workflow-specification language of the workflow-management system IBM MQSeries Workflow [1], forms the internal process layer. Different from eSourcing, the shortcoming of CrossFlow is that the service provider is not able to insert tasks that are not a lower process-level refinement, but that extend the business process on the highest level. At the same time a verification-component must ensure that a refinement by inserted tasks does not violate the perceived service behavior a consumer demands. eSRA comprises components for the verification and simulation of B2B-collaborations.

An overview of interoperability principles, concepts, and methods in [106], discusses why earlier electronic-business-collaboration approaches failed and what requirements are important for building eSRA-compliant systems. Key elements for success are the conceptualization of business processes and policies and automating the interoperability with a service-oriented approach.

Additional business-collaboration architectures are available in literature that lend themselves to a check with eSRA of this paper. In [27], the historical development of architectures for enterprise collaboration gives an overview that shows all approaches use multiple layers for shielding the internal legacy systems, for conceptual process formulation and an external inter-organizational collaboration-layer. Accordingly, the approaches bridge the local syntactic features determined by heterogeneous data and technology with a semantic abstraction layer to achieve a common denominator for the inter-organizational collaboration layer governing business pragmatics.

A security architecture for virtual organizations [62], uses Web-services for supporting inter-organizational business applications and policies to ensure collaboration security. The architecture overlaps with eSRA in that an inter-organizational layer exists for the management of virtual organizations. Just as for eSRA, the participating organizations retain their internal autonomy and exchange with each other only as much as necessary. In summary, the focus of this approach lies on a heavy use of policies for ensuring a secure access to the control flow run by distributed workflows.

In [67], a service-oriented multi-channel architecture also uses multiple layers for protecting the internal legacy-applications that are wrapped as services and integrated in an interaction layer. The latter consists of portals and enterprise-resource planning interfaces. The top layer is for enabling collaboration between organizations. However, there is no explicit discussion about conceptualizing business processes and policies that only show necessary details for establishing collaboration.

Earlier efforts in facilitating B2B-integration include ebXML and RosettaNet. ebXML[6] is a family of XML-based interoperability standards for ensuring smooth flow of electronic information across enterprises in a B2B-integration setting. ebXML can be incorporated into our reference architecture as a standard for message exchanges between the *Coordination_Interface* modules of the *eSourcing_Middleware* components of each collaborating party as depicted in Fig. 5. RosettaNet[7] is a non-profit organization establishing standard business processes for sharing B2B-information. To that end, RosettaNet defines a set of Partner Interface Processes (PIPss) that collaborating parties implement for interaction and exchanging information. The PIPs are composed within the *Workflow_Composer* module of our *eSourcing_Setup_Support* component of Fig. 9.

In [68], the discussion for developing industry-wide reference architectures for ultra-large ecosystems show that eSRA adheres to the stated characteristics. First, the expected decentralization of data, development, evolution and operational control eSRA adheres to with employing the eSourcing concept [75,76] for deducing the functional architecture specification. For the processes the parties develop independently, enactment is distributed and collaboration loosely coupled. Control through monitorability and conjoinment ensures loose coupling and configuration flexibility. Secondly, inherently conflicting requirements where parties want complexity to reside in the domains of counterparties of the overall system and want information to be shared, but do not want to share their own information eSRA satisfies with adopting a three-layer framework. Conflicting requirements may be resolved on conceptual-, or internal layers while intersecting requirements can be addressed on the external layer. The three-layer framework also supports a continuous evolution with heterogeneous elements where the whole ecosystem cannot be stopped and re-engineered. Furthermore, adopting workflow concepts allows for cleanly managing the boundaries between people- and system boundaries. As eSRA employs external-layer process views [45], collaborating parties have an option to

[6] http://www.ebxml.org/.
[7] http://www.rosettanet.org/.

influence each other's behavior. Still, the conceptual-layer refinement options limit the possible control. When a black-box process-view projection takes place with limited conjoinment and monitorability, the service interfaces are minimal. Collaboration metadata and -context in service contracts eSRA supports with comprising an explicit contracting component on the external layer. The expected semantic alignment of eSRA we ensure with the formally defined properties of the eSourcing framework. Finally, intermediaries should not be mandatory during a B2B-collaboration. In eSRA that is possible under the assumption that a set of parties already decided a priori to collaborate exclusively with each other and therefore no need exists for a trusted-third-party component to discover services of unknown parties.

Finally, other reference architectures are enclosable in eSRA. The Workflow Reference Model [109] and the reference architecture for Workflow Management Systems [52] are part of the global and local eSRA workflow management systems. The E-Contracting reference architecture [13] populates the contracting client in the eSourcing middleware.

7. Conclusion and future work

In this paper, we evaluate the eSourcing reference architecture for the development and assessment of enterprise-collaborations. A set of functional and non-functional requirements for eSRA result from carefully studied enterprise-collaboration features. From the functional requirements we deduce the reference architecture that comprises components on three refinement levels and their information exchanges. As the appendices confirm in terms of usability and applicability, the eSRA-specification enables designers to efficiently and effectively assess the quality and comprehensiveness of existing systems.

During the validation of eSRA, the discovered risks and sensitivity points from both evaluation sessions we sum up in the appendices, reveal the importance of verification- and simulation components for electronic business collaboration. Such components serve several collaboration perspectives, e.g., control flow, data flow, resource management, transaction management. These components must have user support that allows an instantaneous detection of errors while not revealing business secrets to the collaborating counterpart.

The anonymizing environment of an eSRA-complying system requires the adoption of components for trust management. The collaborating parties must be able to protect their business secrets and the overall performance of an eSRA-compliant system must not impede business collaboration. Besides the need for verification components, the layer architecture, pipes and filters, and whole-part decompositions result in potential performance bottlenecks in eSRA-compliant systems.

Several directions for future work exist. In addition to the already adopted trust-management components, this is an ongoing research issue. We aim for a complete trust solution via the integration with other collaboration concepts. In ongoing research about e-business transactions for electronic business collaboration, we explore the handling of exceptions and compensation as an integral part of enactment. The feasibility of matching service requests with service offers, requires full- or semi-automation. For service matching, different methods characterize themselves with different levels of computational expensiveness and matching quality. Hence, we investigate suitable and complete methods for different stages of service matching. An integral part of eSRA is the trusted-third-party component that reduces the communicational overhead between collaborating parties and ensures scalability. For the cloud-computing domain, the development of a stack for a cloud ecosystem is ongoing. Thus, we explore how to integrate eSRA into the cloud stack for enabling business collaboration in this setting.

Acknowledgments

We thank the experts who participated in the first ATAM-workshop, namely Juha Gustafsson and Joni Niemi from the Software-Engineering department, Lea Kutvonen from the CINCO-group of the University of Helsinki, Jukka Viljamaa from VTT. Their detailed and thorough exploration of eSRA was essential for generating evaluation results that have lead to a deeper understanding for the subsequent validation. Furthermore, we also thank the participants to the second ATAM-workshop from Estonian Kuehne&Nagel, Elion, Estonian Information Systems Authority, IceFire, IoT, Tallinn University Department of Computer Science and Tallinn University of Technology Department of Informatics.

This research was partly conducted in the EU-FP6 research project CrossWork [49], SOAMeS (Service Oriented Architecture in Multichannel e-Services), Trusted Business Transactions (Academy of Finland 129117) and the CloudSoftware[8] project.

Appendix A. First ATAM-Evaluation Results

The tables below show the first set of results from applying the ATAM-evaluation method. The initial group of five participants had either a full-, or applied research background from the Department of Computer Science at the University of Helsinki or the Finnish applied-research organization VTT that matches the top-down eSRA-development process starting from a conceptual- and formal collaboration-model foundation. The workshop results in a refinement of the eSourcing architecture's (eSRA) quality attributes with specific sub-factors. The latter factors create a utility tree together with accompanying expert-ranked scenarios. Further refinements of the top-ranked scenarios incorporate sensitivity points, tradeoff points, risks, and non risks, as the tables below show.

[8] http://www.cloudsoftwareprogram.org/.

A.1. Attribute-characterizing questions

The initial task of the workshop sessions with experts is to clarify the utility requirement of eSRA that expresses the overall "goodness" of the system. Hence, the posed questions establish a better understanding of earlier determined quality attributes.

In Table 4, quality attributes on the left side cluster sets of assigned questions in combination with an additional qualification if they represent for eSRA an external stimuli, an architectural decision, or a response. External stimuli are events that cause eSRA to respond or change. Architectural decisions are the aspects of eSRA that have a direct impact on achieving attribute responses. Finally, responses are measurable or observable quantities.

A.2. Utility tree

All the quality attributes and further specifying sub-factors organize and prioritize the goals for eSRA in a utility tree. The latter is instrumental for a sub-factor assigned creation of scenarios. Next, the perceived difficulty and priority for eSRA determines the ranking of these scenarios.

The left side of Table 5 mentions the most important system requirement, namely the utility. Further refining quality attributes for utility are a subset of the non-functional requirements in Section 3.2. The subset of attributes results from a discussion with the expert group about what to consider for further questionings, as Table 4 shows. Quality attributes for which the experts cannot find questions are not considered in the ATAM-evaluation as it implied that eSRA provides coverage. Next, we deduce the sub-factors in Table 5 from the questions in Table 4 and refine the quality attributes. In effect, the sub-factors tell a deeper probing of eSRA must take place.

The scenarios to the right of sub-factors carry identification numbers. At the very right-hand of Table 5, the scenarios-assessment uses their perceived difficulty and the priority to realize. The gray shaded rows contain scenarios that score highly in both categories and require further investigation. Tables 6–8, give justifications for the ranking of scenarios.

Table 4
Attribute-characterizing questions.

Quality attributes	Stimuli	Architectural decision	Response	Questions
High automation	X			Is It possible to define examples that state what is automatic or not automatic?
		X		Is manual interaction required?
	X			Does the legacy system influence decision (special input required)?
		X		Is it possible to perform a highly automated bidding (Agents involved, or humans involved)?
		X		What is the target degree of automation
			X	How can the degree of automation be controlled?
Usability		X		Does the system cover also operational time observability and breach detection?
			X	Is there a discrepancy detection during bidding involved?
			X	How about handling errors during enactment phase?
		X		Is it possible to test run the entire eSourcing configuration?
	X			Based on which information is the decision made to eSource?
			X	When a system provides an eSourcing configuration, how much interoperability does that guarantee?
Modifiability		X		Are three levels enough or are more needed?
		X		What are the expected rules of expressing yourself In each level (structuring rules)?
			X	What happens when the modeling languages are changed?
		X		What happens when a totally new component with new functionality is added?
		X		Does the architecture forbid certain extensions?
Scalability		X		Does only one trusted third party component turn into a bottle neck?
		X		Is it possible to have multiple collaborating parties in an eSourcing configuration?
		X		Can multiple modeling languages be used on different levels?
	X			How many enactment parties are anticipated?
			X	Does replication imply effects on the protocols?
		X		Are dedicated components required to support replication?
			X	How does scalability affect performance?
Performance			X	Is verification a bottleneck during the setup phase?
			X	How does the system ensure the response time is acceptable during enactment?
Flexibility		X		To what extent does system facilitate different contracting protocols?
	X			Is multi-phase contracting supported?
	X			Is iterative contracting supported?
			X	How does eSRA respond when a trusted-third parties are replicated?
			X	How does eSRA respond when a contracting component is replicated?
Integrability		X		Which components are envisaged that need to be integrated?
Security		X		How is trust management realized in eSRA?
		X		How are the legacy systems shielded from denial of service attacks?
		X		How is sensitive data exchange shielded?
		X		How are process details hidden from collaborating parties?
			X	What are the different trust levels between different eSRA integration options?

Table 5
ATAM utility tree.

	Quality attributes	Sub–factors	#	Scenarios	Difficulty	Priority
Utility	Modifiability	Adding new components	1	A fourth collaboration –level is added for enabling inter–organizational e–business transaction management without triggering modifications to components contained in other eSourcing levels.	Low	Medium
		Adding functionalities	2	An existing component is extended , e.g., the global WFMS, with additional functionality without triggering knock–on modification that goes further than immediately bordering components. These knock–on modifications are localized to the same collaboration level where the component is embedded that is extended with more functionality.	Low	Low
		adding new SW–versions	3	The global rules engine is updated to a new version without triggering update requirements to the local rules engines of respective collaborating parties .	Low	Low
	Usability	Response time	4	When a task is accepted on the service–provider side, that change is propagated within 5 s through to the side of the service consumer and visible on the enactment–progress monitor.	Low	Low
		Error avoidance	5	For a modeled eSourcing configuration , tools are available that detect control–flow errors in a way that enable respective parties to perform an internal modification of the process model without revealing business secrets to the counterparts.	High	High
		Learnabilily	6	Users are able to setup an eSourcing configuration after 1month of training.	Medium	Medium
		Error handling	7	An error that occurs during the enactment of an eSourcing configuration is caught and managed without requiring stopping the enactment.	High	High
		Verification	8	eSA provides the com ponents required to verify the perspectives of data–flow, control·flow, resource management before enactment.	High	High
	Intero perability	Legacy–system integration	9	A SAP inventory management system can be integrated in the selling up of an eSourcing configuration.	Low	High
		Between systems of participating parties	10	An eSourcing configuration allows an ORACLE and SAP accounting system to exchange securely monetary transaction data.	Low	High
		Heterogeneous business functions	11	Provided one collaborating party internally uses BPMN and the counterpart uses EPC, it is still possible to set up an eSourcing configuration.	Low	High
	Performance	Verification	12	The control–flow verification of an eSourcing configuration completes in less than 1 h.	Medium	Medium
		Response time during enactment	13	The reaction of an eSA–based system never requires more than 5 s for responding to a user interaction .	Medium	Medium
	Scalability	Broker	14	The eSA domain of a collaborating party allows the integration of severeal service brokers for different industrial domains, e.g., a broker for telecom industry, a different broker for forrest industry.	Low	Medium
		Bidding	15	A reasonably high number of service providers may bid for a particular request from a service consumer.	Low	High
		Enactment	16	eSA–based systems support the integration of more than one service provider into the enecatment of one eSourcing configuration.	Low	High
	Security	Legacy system	17	The integration of an ORACLE resource–management system does not endanger its operation.	Low	High
		Data exchange	18	Data that is exchanged in an eSourcing configuration does not originate from and is not diverted to an unauthorized destination.	Low	High
		Trust management	19	A collaboration does so badly that it is not desirable any more to repeat the collaboration.	High	High
		Business–process details	20	Internal business–process details of a collaborating party are not revealed during the setup and enactment of an eSourcing configuration unless they are proiected to an external level.	Low	High
		Trusted third party	21	IA denial–of–service attack is launched against the broker which does not result in accessibility problems.	Low	High

Table 6
Justifying the ranking of scenarios (part 1).

Scenario	Justification	Rank	
1	Since eSRA comprise a part-whole pattern and a layered architecture, adding new components is not difficult.	Low	Difficulty
	As an example for an open extension issue in eSRA that is not trivial, e-business transaction (eBT) management is required to safeguard B2B collaboration. For now eBT is an ongoing research topic and evolving.	Medium	Priority
2	Although component changes should remain isolated, it is realistic that minor changes need to propagate to surrounding components.	Low	Difficulty
	Given the inclusion of a layer style and a whole/part pattern, sufficient provisions are in place to not consider adding functionality as a challenge that requires high priority.	Low	Priority
3	No challenge, provided the update does not result in the adoption of rule-types that are not understood by the local rule engines.	Low	Difficulty
	Same justification as for Scenario 2.	Low	Priority
4	No challenge, provided the hardware infrastructure and the legacy systems are responding well.	Low	Difficulty
	Since it is assumed that powerful and modern hard- and software and well designed legacy systems are applied, response time is considered assured.	Low	Priority
5	Tool support for error avoidance implies a considerable research effort for powerful algorithms that support users in error avoidance. Creating a suitable GUI for pinpointing mistakes in a collaboration model is not trivial.	High	Difficulty
	Without the ability of verifying and evaluating the correctness of an electronic B2B collaboration the success of the enactment phase cannot be assured. Most likely the enactment will run into problems that are followed by penalty payments, unsatisfied customers, and so on because of unintended contractual violations.	High	Priority
6	Provided the users have a background that unites business. IS. and computer-science knowledge, applying an eSRA-compliant system can be learned in an acceptable time. A highly skilled consultant would be required to back up a 'regular' user.	Medium	Difficulty
	If the system is so complicated that layman users cannot set up and enact an electronic B2B collaboration despite the help of an assisting consultant, the adoption of eSRA-compliant systems is in doubt.	Medium	Priority
7	The question is not answered how inter-organizational enactment errors should be handled in electronic B2B collaborations. Since enactment-error management should also comprise advanced inter-organizational e-business transaction management concepts that are only now scientifically investigated.	High	Difficulty
	Mechanisms must be in place that contribute as far as possible and feasibly to a successful completion of the enactment phase.	High	Priority

Table 7
Justifying the ranking of scenarios (part 2).

Scenario	Justification	Rank	
8	Same reason as with Scenario 7.	High	Difficulty
	As many perspectives of an electronic B2B-collaboration must be verifiable before enactment, control-flow. data-flow, resource perspective are a minimum.	High	Priority
9	Any legacy system can easily be integrated if it is wrapped as a service that can be orchestrated by local WFMS and rules engines.	Low	Difficulty
	It is essential that all legacy systems can be integrated in an electronic B2B-collaboration. Otherwise the purpose of an eSRA-compliant systems is defeated.	High	Priority
10	On the external level the exchange of data between opposing collaborating parties is managed by a coordination-interface component that represents a facade pattern. Hence, the coordination interface can ensure secure data-transmission. Extra security is given since the legacy systems of opposing collaborating parties are only indirectly linked in an electronic B2B-collaboration.	Low	Difficulty
	The data exchanges between legacy systems in an electronic B2B-collaboration must never be compromised in any situation. Any data format must be exchangeable.	High	Priority
11	Different business-collaboration formulation languages are used on separate eSRA levels, which allows the projection of one language to another language on the external level. The real problem is the extent to which different business-collaboration formulation languages share semantically similar language constructs.	Low	Difficulty
	It cannot be assumed collaborating parties use identical concepts, notations, technologies. It is very important the differences can be inter-organizationally reconciled by an eSRA-compliant system.	High	Priority
12	E.g., tools for control-flow checking require much time and computing power. With heuristics, the verification of control-flow. Data-flow, resource management, transaction management, and so on, will be within an acceptable limit.	Medium	Difficulty
	If verification is computationally too expensive, the results of the setup phase cannot be checked and the enactment phase is in doubt before it even starts. Also ad-hoc verifications during enactment time are impossible if that is too expensive.	Medium	Priority
13	Given the three levels of eSRA and the multiple components involved that may be geographically dispersed, a 5 second response time might be challenging.	Medium	Difficulty
	Essential for supporting the adoption of eSRA-com pliant systems by corporations. However, with powerful, modem hardware and technology and the use of well designed legacy systems, response time is not a problem.	Medium	Priority
14	Broker scalability merely requires separate instantiations of the broker. Additionally, the trusted-third-party component of eSRA-compliant systems must be designed in a way to handle several brokers.	Low	Difficulty
	Broker scalability is a relevant scenario that enhances the adoption of eSRA-com pliant systems, however, it is easy to achieve due to the modularity of eSRA.	Medium	Priority

Table 8
Justifying the ranking of scenarios (Part 3).

Scenario	Justification	Rank	
15	Provided the server on which the broker process runs is powerful, bidding scalability is assured.	Low	Difficulty
	An eSRA-compliant system must allow all potential collaborating parties to take part in bidding. Otherwise it is not ensured that the best deal is realized.	High	Priority
16	In [31], it is shown that eSourcing allows many service providers in an electronic B2B-collaboration. The whole-part pattern used in eSRA supports such a scenario.	Low	Difficulty
	Real-life business collaboration is often not limited to only one service provider. Hence. eSRA-compliant systems must allow the establishment of an electronic B2B-collaboration with multiple opposing parties.	High	Priority
17	Legacy system security is assured as they are not directly exposed to collaborating counterparts. Instead legacy systems remain securely on the internal level under the protection of several eSRA levels.	Low	Difficulty
	eSRA-compliant systems must not violate the security of other legacy systems that are vital for the day-to-day business of companies.	High	Priority
18	The collaborating parties first need to contact each other via the trusted-third party if they have not agreed a priori on an electronic B2B-collaboration for long term. Hence, in eSRA-compliant systems, prior contact is established before a collaboration is set up.	Low	Difficulty
	It is important that the collaborating parties can trust the origin of exchanged data.	High	Priority
19	Trust management is an ongoing research issue that is not under exploration for electronic B2B-collaborations. Hence, realizing comprehensive electronic trust management in eSRA is not trivial.	High	Difficulty
	In an anonymized market that is driven by electronic services, it is not clear how much the counterpart can be trusted. Thus, mechanisms must be in place that allow to fill this knowledge gap about a potential collaborating counterpart.	High	Priority
20	eSRA-compliant systems have a conceptual and external level to which subsets of the conceptual processes are projected.	Low	Difficulty
	When the engagement in an electronic B2B-collaboration results in the unintended disclosure of important business secrets. The existence of a company is in Jeopardy.	High	Priority
21	This standard problem has been technically solved.	Low	Difficulty
	If the trusted-third party does not function, it is not possible for an eSA-compliant system to support anonymized markets with automated bidding and other services like inter-organizational verification components. Thus, only parties that already agree a-priori to collaborate, can engage in setting up an electronic B2B-collaboration that cannot be verified in a satisfactory way.	High	Priority

A.3. Scenario specification

Since there is limited time available in an ATAM-evaluation, the workshop participants scrutinize only the highest ranked scenarios for further refinement. In the case of this evaluation, the highest ranked scenarios of Figs. 18 to 21 fall into the category of use cases, i.e., they describe the user's intended interaction with the completed, running system.

All the scenarios in Figs. 18 to 21 use the same template for their specification. A relevant part of the specifications are the architectural decisions in eSRA for dealing with the scenarios. The left sidelists the decisions and to the right whether they constitute so-called sensitivity or tradeoff points, risks or non-risks (see [61] for details). Table 9 lists the sensitivity- and tradeoff points, risks and non-risks.

Appendix B. Second ATAM-evaluation results

In this round of ATAM-workshop, the group of seventeen participants comprises primarily practitioners from industry. Five members from the Estonian branch of Kuehne&Nagel have a background in system-architecture development and software engineering. The same or similar background have the two participants from the Estonian telecom provider Elion, two from the Estonian Information Systems Authority, two from the startup firm IceFire and one from the startup firm IoT. Additionally, two PhD-candidates participate from the Tallinn University Department of Computer Science and the Tallinn University of Technology Department of Informatics, respectively. Finally, from the latter department also one professor is a participant with a research focus in software agents and business-process management.

The agenda of this second ATAM-workshop commences with an introduction to ATAM, followed by a presentation of the eSourcing framework that is the top-down input for developing the reference architecture. Next, the eSRA-specification is coupled with an explanation of the results from the first workshop that the participants acknowledge and approve. In the second part, the actual eSRA-evaluation takes place with first deciding on a user perspective for the meta-goals of the utility tree.

B.1. Second attribute-characterizing questions

After the presentations in the morning, the afternoon session is interactive with the participants actively engaging in discussions and generating contributions. Table 10 sums up the attribute-characterizing questions posed to the workshop coordinator that is a means for participants to understand eSRA in detail. We provide a discussion that reflects the given answers to the listed questions that are mostly qualitative in nature as a reference architecture is not domain-, or context specific as in the case of standard-, or concrete architectures (see Fig. 4). Note that the participants gave the quality attributes with an intuitive understanding without agreeing on academic definitions and many questions address concrete application-system instantiations.

Scenario 5:
For a modeled eSourcing configuration, tools are available that detect control-flow errors in a way that enable respective parties to perform an internal modification of the process model without revealing business secrets to the counterparts.

Attribute(s) usability (error avoidance)

Environment eSRA-compliant system

Stimulus created business-collaboration model

Response
A verification tool points out errors in the model only to the party concerned with correction measures while the opponent only receives a message that a mistake exists.

Architectural decisions	Sensitivity	Trade off	Risk	Nonrisk
Verification component in the trusted-third party.	S1			
GUI with separate views for collaborating parties			R1	
Recommendation component for fixing mistake			R5	
Dedicated server assigned to verification		T1		
Verification of collapsed net				N1

Reasoning
Inter-organizational verification ensures correct termination and must be checked before enactment.
The concerned collaborating party must be able to find the mistake and know how to correct it so that the overall configuration can terminate.
Still, business secrets must remain hidden from counterpart (see R1).

Architectural diagram
See the verification components in Figure 9 and Figure 17 of this paper.

Fig. 18. Specification of highly ranked Scenario 5.

B.1.1. Reliability

Yields three questions of which the first one is answered in an architectural-decision way. Consequently, a reliability goal of the trusted third party is to provide the components for validating collaboration configurations. The maximum downtime as a response requires zero tolerance as the trusted third party is essential for the setup phase. Downtime itself may be total when the entire third party fails or partial when, e.g., the collapsing- or, validation-component fails while service-search is still possible.

B.1.2. Maintainability

We understand this quality factor as who maintains eSRA in its documentation depending on various application domains and instantiation contexts. eSRA is ideally standardized comparable to the workflow management reference architecture [52] and maintained by a standardization coalition.[9] Rules and processes are part of the setup face as they are either specified or chosen from available libraries for re-use. During the enactment phase the rules and business processes are carried out and participating organizations of a collaboration configurations. As specifications for rules and business processes are machine readable, many change adjustments are possible with specification changes. With adopting a layering style and whole-part decomposition pattern in eSRA, changes to the components are easy to introduce.

B.1.3. Performance

Predictable bottlenecks of eSRA-compliant systems are the trusted third party component as it is essential during collaboration setup. Also the verifications before enactment are potentially computationally expensive, e.g., if verification tools use Petri-net mappings with large state spaces. eSRA as a reference architecture performs in that it allows quick paper-based assessments of other architectures and system-instantiations.

[9] http://www.wfmc.org/.

Scenario 7:
An error that occurs during the enactment of an eSourcing configuration is caught and managed without requiring stopping the enactment.

Attribute(s) usability (error handling)

Environment eSRA-compliant system

Stimulus data-flow: wrong data sent

Response
Depending on the severity of the error, either exception handling follows till system recovery, or inter-organizational business-process compensation is started.

Architectural decisions	Sensitivity	Tradeoff	Risk	Non risk
Global versus local WFMS and Rules Engines.	S2			
A reliable coordination component.			R2	
Distributed WFMS and rules engines.				N2
Level architecture and global WFMS and rules engine replication.		T2		

Reasoning
Errors and compensations have several aspects, which can be grouped as business, conceptual, and technological. These different groups should be handled on different eSRA levels to achieve a separation of concerns that reduces complexity.
For example, the WFMS and rules engine on the external level are concerned with business and conceptual exceptions and compensations, while the internal WFMS and rules engines deal with conceptual and technical exceptions and compensations.

Architectural diagram
See global and local WMFS and rules engines in Figure 6 and Figure 10 of this paper.

Fig. 19. Specification of highly ranked Scenario 7.

Scenario 8:
eSRA provides the components required to verify the perspectives of data-flow, control-flow, resource management before enactment.

Attribute(s) usability (verification)

Environment eSRA-compliant system

Stimulus Ready modeled eSourcing configuration

Response
Results of the verification, i.e., either errors are detected or there are no errors.

Architectural decisions	Sensitivity	Tradeoff	Risk	Nonrisk
Localize verification as far as possible.	S3			
Idem.		T3		
An evaluation component is part of eSRA.			R3	
Verification components on conceptual level.				N3

Reasoning
It is important that an eSourcing configuration is thoroughly verified in all perspectives to ensure a correct enactment phase. When enactment fails, penalty payment must follow, customers are unsatisfied, etc., which is all undesirable.

Architectural diagram
Figure 9 and Figure 17 of this paper.

Fig. 20. Specification of highly ranked Scenario 8.

Scenario 19:
An electronic business collaboration experiences an exception, e.g., business-rule violation. Consequently, it is not desirable to continue the collaboration.

Attribute(s) security (trust management)

Environment eSA-compliant system

Stimulus A business party detects deviating business-collaboration behavior compared to what has been agreed.

Response Update of the reputation record about badly collaborating counterparts and possibly exception handling and compensation.

Architectural decisions	Sensitivity	Trade off	Risk	Nonrisk
Integrate additional trust components in trusted third party.	S4			
Integrate appropriate components.		T4	R4	
Refinement of trusted third party.				N4

Reasoning
In an anonymized electronic collaboration environment, it is necessary to integrate mechanisms for evaluating the reputation and trustworthiness of potential collaboration counterparts.

Architectural diagram
In Figure 6&7 of this paper, components for identity- and reputation management are part of the eSourcing middleware and trusted third party.

Fig. 21. Specification of highly ranked Scenario 19.

B.1.4. Usability

As eSRA is based on pre-existing peer-reviewed scientific literature about collaboration models, users of eSRA save time when they either assess other collaboration systems or need to develop their own standard- or concrete architecture. This paper with detailed eSRA assessment and the pre-existing literature facilitates the guidance of use. Complexity removal by eSRA use happens as the core concepts are explained and how they map to the architecture.

B.1.5. Security

Security breaches eSRA tackles with a coordination-interface component on the external-layer of collaborating parties. This component serves as a façade to protect internal details and filter out events and data that are obscure. Additionally, the

Table 9
Risks, sensitivity- and tradeoff points, non-risks.

Risk	Sensitivity	Tradeoff	Nonrisk	Explanation
	S1			GUI of verification tool that protects process details while allowing fault pinpointing.
R1				GUI that separates process views for respective collaborating parties for error detection.
		T1		Usability and performance during the setup phase are potentially diminished by bad GUI.
			N1	Support of correct-termination check for inter-organizational eSourcing configuration.
	S2			A coordination between global and local exception handling and compensation is essential.
R2				The global WFMS and rules engines are replicated and require coordination.
		T2		Due to replicated WFMS and rules engines, performance, usability, and modifiability are affected.
			N2	Locally occurring mistakes can be managed by global WFMS and rules engine.
	S3			Inter-organizational verification in many perspectives may be computationally too expensive.
		T3		Performance problems unless pragmatic heuristics are used. Usability issues because of GUI. Security is compromised if verification results are wrongly disclosed to collaborating counterpart.
R3				Even with verified perspectives, there is no guarantee they work together.
			N3	When certain rules are followed, control-flow verification can be successfully performed locally alone.
	S4			Trusted-third-party component lacks components for trust management.
		T4		Trust-management components may prolong the setup and post-enactment phase.
			N4	Trusted-third-party component can be refined with components to enable trust management.
R4				Trust, reputation, conflict components are missing in eSRA.
R5				Without a recommendation component, fixing problems might be only possible for experts.

Table 10
Attribute-characterizing questions.

Quality attributes	Stimuli	Architectural decisions	Responses	Questions
Reliability		X		What are reliability goals of the trusted third party?
			X	What is the maximum acceptable downtime per year?
	X			What is downtime?
			X	Who is responsible for reliability?
Maintainability			X	Who maintains the system?
		X		Are business rules part of setup or enactment?
		X		Who is responsible for changing the rules?
	X		X	What triggers changing the rules/processes?
		X		How to cope with legacy-system changes?
Performance		X	X	Where is the bottleneck?
		X		How does RA compare to alternative standard architecture?
Usability			X	How to convince user to adopt eSRA?
		X		How to guide the user through system use (problems occur)
		X		Does eSRA remove complexity?
Security		X	X	How to deal with security breaches?
		X		How to recreate trust?
		X		How to enforce legal compliance to standards, practices, agreements?
Integrity				Who governs the changes in the TTP?
Flexibility	X	X	X	How to handle changes in the business context?
				How to manage SLAs?
			X	How to handle a situation where the business requirements for the architecture change during design time?
	X			Who evolves an instantiation of eSRA?
	X			How to predict increase in data volume?
Observability				What is the system status?
				How to get and analyze feedback?
				How to monitor event flows?
Privacy		X		How to cater for privacy?
		X		Can eSRA be applied in totally anonymous markets?
Interoperability		X		Does the architecture prescribe explicit or implicit ontology?
		X		How does eSRA relate to P2P?

three-layered architecture secures the internal legacy systems behind a conceptual- and external layer. Trust management eSRA caters for by comprising components for reputation-, identity- and trust management. Once trust-breach happens, the committing party must demonstrate trustworthy conduct over time to statistically repair reputation and trust. Additionally, each breach of trust requires checks as a business rule can force a collaborating party into behavior perceived as not trustworthy without involving malintent. Legal compliance is best assured with the specification of business rules and processes.

B.1.6. Integrity

For the trusted third party, the assumption is to be independent as privacy compromises occur when one collaborating party holds that component in his domain.

B.1.7. Flexibility

Changes in the business context again the machine-readable specifications of business rules and processes address while the management of service-level agreements happens on the external layer and in the trusted third party. Business requirement changes are domain- and context specific that do not affect the reference-architecture level. The same holds for predictions in

Table 11
Utility tree from second ATAM-evaluation adopting a user perspective.

	Quality attributes	Sub-factors	Nr	Scenarios	Importance	
utility	Performance	Validation	1	Balanced elimination of livelocks and deadlocks.	H	H
		Brokering	2	Quick matching with collaborating counterparty.	H	L
	Modifiability	Of rules	3	20 new rules are created in a week per party	H	L
		Of business context	4	Collaborators move from M/C- to P2P-collaboration.	M	H
	Availability	Downtime	5	Failing DB must be replaced on the fly.	H	L
		Uptime	6	Sound termination of the collaboration transaction.	H	M
	Security	Breach prevention	7	Public IP-address cannot be determined.	H	L
		Trust creation	8	Trust-relevant factors captured and feasible to process	H	M

Table 12
Second set of sensitivity- and tradeoff points, risks and non-risks.

Scenario	Risk	Sensitivity	Tradeoff	Non-risk	Explanation
1				N5	As validation tools are in trusted third party and respective external layers, danger averted of disclosing business secrets by accident.
2	R6	S5	T5		Sensitive to availability, risk avertable by socio-technical software agents, tradeoff to security with less counterparty check.
3			T6	N6	Tradeoff to performance if visual tools used, Non-risk as modeling components on conceptual level.
4	R7	S6			Risk averting by using socio-technical software agents for complexity management, sensitive to availability if trusted third party fails.
5		S7		N7	Sensitive to modifiability when rules and process DBs disappear, non-risk as abstract-data repository style proposed.
6		S8		N8	Sensitive to performance if validation is computationally expensive, non-risk as validation components present.
7				N9	Non-risk because of the coordination-interface component on the external layer.
8		S9, S10			Sensitive to performance depending on how complete the data sets are, also sensitive to availability as the trusted-third-party component is essential.

data-volume. Thinkable is a concrete architecture instantiation in CPN[10] for carrying out laboratory-type tests on data-volume distribution. System architects are candidates for evolving an instantiation of eSRA.

B.1.8. Observability

The translator- and monitorability components of eSRA allow for are aware of collaboration progress. Additionally, flexible observability the underlying collaboration model permits with adjustable monitorability-specifications.

B.1.9. Privacy

The three-layer style of eSRA allows for using process views that are subsets of the conceptual-layer business processes. With the trusted-third-party component in the middle, collaborating parties have the option to remain very anonymous. Also the coordination-interface component of the external layer allows for obscuring internet addresses.

B.1.10. Interoperability

Ontology-relating decisions are relevant for concrete architectures, or system instantiations and not relevant for a reference-architecture level. Finally, peer-to-peer collaboration models versus the standard assumed master-client collaboration are possible with the same eSourcing framework if the roles of elements involved change. Thus, for peer-to-peer collaboration, the service consumer's in-house process does not exist but is replaced with a business-network model [64] that serves as an eCommunity collaboration blueprint. The trusted third party [79] extends in that it manages the business-network models too. The latter comprise service types instead of sourcing spheres that collaborating parties match with service offers on the external layer.

B.2. Second utility tree

A user perspective the workshop participants adopt for the utility tree in Table 11 by focusing on the non-functional eSRA goals of performance, modifiability, availability and security. The sub-factors result from a discussion of the workshop-participants and the approach taken follows the procedure from the first round of ATAM-evaluation.

The results in Table 11 show that the validation of collaboration configurations is a critical factor performance. The workshop participants consider it essential to eliminate faults before the enactment phase and realize this is a hard problem to solve for all perspectives. Furthermore, modifiability of collaborations from traditional master–client towards peer-to-peer is also of high importance but not perceived as a challenge as the collaboration model [45] that is input for deducing eSRA also scales to a peer-to-peer direction, as we explain above. A sound termination of collaborations is also of high importance and relates to the highest rated validation goal. Finally, the capturing and meaningful processing of trust-relevant factors is important as an eSRA-compliant system brings together potentially unknown organizations, services and persons into the same collaboration.

B.3. Sensitivity- and tradeoff points, risks and non-risks

As the scenarios of the second ATAM-evaluation yields a similar top-ranked scenario as in the first round while being this time under the non-functional goal of performance instead of usability, we omit re-specifying scenarios as in Fig. 18. Instead, we give the set of collected sensitivity- and tradeoff Points, risks and non-risks. Note, the first column in Table 12 list the scenario numbers from Table 11.

[10] http://cpntools.org/.

References

[1] IBM MQSeries workflow, http://www-4.ibm.com/software/mqseries/workflow.
[2] Business Process Modeling Notation (BPMN), Version 2.0. Object Management Group, 2011. (http://www.bpmn.org/spec/BPMN/2.0/).
[3] W.M.P. van der Aalst, Structural characterizations of sound workflow nets, Computing Science Reports 96/23, Eindhoven University of Technology, Eindhoven, 1996.
[4] W.M.P. van der Aalst, Verification of workflow nets, in: P. Azéma, G. Balbo (Eds.), Application and Theory of Petri Nets 1997, Lecture Notes in Computer Science, vol. 1248, Springer-Verlag, Berlin, 1997, pp. 407–426.
[5] W.M.P. van der Aalst, The application of petri nets to workflow management, J. Circ. Syst. Comput. 8 (1) (1998) 21–66.
[6] W.M.P. van der Aalst, A.H.M. ter Hofstede, N. Russell, Workflow patterns, http://http://workflowpatterns.com/ 2013.
[7] M. Adams, Facilitating flexibility and dynamic exception handling in workflows, (PhD thesis) Faculty of Information Technology, Queensland University of Technology, Brisbane, Australia, 2007.
[8] Michael Adams, Arthur H.M. Hofstede, Wil M.P. Aalst, David Edmond, Dynamic, extensible and context-aware exception handling for workflows, in: Robert Meersman, Zahir Tari (Eds.), On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, Lecture Notes in Computer Science, vol. 4803, Springer, Berlin Heidelberg, 2007, pp. 95–112.
[9] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, N. Weiler, WISE: business to business e-commerce, Proc. of the 9th International Workshop on Research Issues on Data Engineering, 1999, pp. 132–139, (Sydney, Australia).
[10] S. Angelov, Foundations of B2B electronic contracting, (Dissertation) Technology University Eindhoven, Faculty of Technology Management, Information Systems Department, 2006.
[11] S. Angelov, P. Grefen, D. Greefhorst, A classification of software reference architectures: analyzing their success and effectiveness, Sep. 2009. 141–150.
[12] S. Angelov, J.J.M. Trienekens, P. Grefen, Extending and adapting the architecture tradeoff analysis method for the evaluation of software reference architectures, 2014. http://purl.tue.nl/396031470035009.pdf (forthcoming).
[13] Samuil Angelov, Paul Grefen, An e-contracting reference architecture, J. Syst. Softw. 81 (November 2008) 1816–1844.
[14] Apache, Apache Xalan Project, https://xalan.apache.org/ 2013.
[15] Apache, Apache Xalan XML Project, https://xml.apache.org/xalan-j/trax.html 2013.
[16] Apache, Hadoop, https://hadoop.apache.org/ 2013.
[17] Apache, Hive, https://hive.apache.org/ 2013.
[18] A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, Basic concepts and taxonomy of dependable and secure computing, IEEE Trans. Dependable Secure Comput. 1 (1) (2004) 11–33.
[19] O. Barbosa, C. Alves, A systematic mapping study on software ecosystems, Proceedings of the Workshop on Software Ecosystems, 2011.
[20] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, Addison-Wesley, 1998.
[21] T. Bellwood, L. Clment, D. Ehnebuske, et al., UDDI version 3.0, published specification, http://uddi.org/pubs/uddi-v3.00-published-20020719.htm 2003.
[22] P.O. Bengtsson, Architecture-level modifiability analysis, (PhD thesis) Department of Software Engineering and Computer Science, Blekinge Institute of Technology, Sweden, 2002.
[23] R.I. Benjamin, D.W. de Long, M.S. Scott Morton, Electronic data interchange: how much competitive advantage? Long Range Plan. 23 (1) (1990) 29–40.
[24] J. Bosch, From software product lines to software ecosystems, Proceedings of the 13th International Software Product Line Conference, SPLC '09, Carnegie Mellon University, Pittsburgh, PA, USA, 2009, pp. 111–119.
[25] D. Box, D. Ehnebuske, G. Kakivaya, et al., Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/SOAP/ 2003.
[26] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal, Pattern-oriented software architecture: a system of patterns, Wiley, Chichester, UK, 1996.
[27] David Chen, Guy Doumeingts, François Vernadat, Architectures for enterprise integration and interoperability: past, present and future, Comput. Ind. 59 (7) (2008) 647–659.
[28] Chi-Bin Cheng, Solving a sealed-bid reverse auction problem by multiple-criterion decision-making methods, Comput. Math. Appl. 56 (12) (2008) 3261–3274.
[29] R. Chinnici, M. Gudgin, J.J. Moreau, Sanjiva Weerawarana, Web Services Description Language (WSDL) version 1.2, http://www.w3.org/TR/2003/WD-wsdl12-20030611 2003.
[30] Lawrence Chung, JulioCesarSampaio Prado Leite, On non-functional requirements in software engineering, in: Alexander T. Borgida, Vinay K. Chaudhri, Paolo Giorgini, Eric S. Yu (Eds.), Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science, vol. 5600, Springer, Berlin Heidelberg, 2009, pp. 363–379.
[31] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, J. Stafford (Eds.), Documenting Software Architectures: Views and Beyond, Second edition, Addison-Wesley Professional, 2010.
[32] P. Clements, R. Kazman, M. Klein, Evaluating Software Architectures: Methods and Case Studies, Addison Wesley, 2002.
[33] Cloudera, Impala, http://www.cloudera.com/content/cloudera/en/products/cdh/impala.html 2013.
[34] IEEE Computer Society, Software Engineering Technology Committee, Committee, Institute of Electrical, and Electronics Engineers, IEEE Recommended Practice for Software Requirements Specifications, IEEE Std. Institute of Electrical and Electronics Engineers, 1994.
[35] R. Davidrajuh, Distributed workflow based approach for eliminating redundancy in virtual enterprising, J. Supercomput. 63 (1) (2013) 107–125.
[36] Alan M. Davis, Software Requirements: Objects, Functions, and States, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.
[37] Gero Decker, Oliver Kopp, Frank Leymann, Mathias Weske, Interacting services: from specification to execution, Data Knowl. Eng. 68 (10) (2009) 946–972.
[38] Elisabetta Di Nitto, Carlo Ghezzi, Andreas Metzger, Mike Papazoglou, Klaus Pohl, A journey to highly dynamic, self-adaptive service-based applications, Autom. Softw. Eng. 15 (3–4) (2008) 313–341.
[39] R.P. dos Santos, C.M.L. Werner, A proposal for software ecosystems engineering, Proceedings of the Workshop on Software Ecosystems 2011, 2011.
[40] J. Ebert, G. Engels, Observable or invocable behaviour you have to choose! Leiden University, Technical report 94–38, 1994.
[41] ebXML Technical Architecture Project Team, ebxml technical architecture specification, 2013.
[42] G. Hicker, C. Huemer, H. Erven, M. Zaptletal, The web services-business activity-initiator (WS-BA-I) protocol: an extension to the web services-business activity specification, 2007 IEEE International Conference on Web Services (ICWS 2007), 2007, pp. 216–224.
[43] R. Eshuis, P. Grefen, Structural matching of BPEL processes, ECOWS '07: Proc. of the Fifth European Conference on Web Services, IEEE Computer Society, Washington, DC, USA, 2007, pp. 171–180.
[44] R. Eshuis, A. Norta, A framework for service outsourcing using process views, Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International, 2010, pp. 99–108.
[45] R. Eshuis, A. Norta, O. Kopp, E. Pitkaenen, Service outsourcing with process views, IEEE Trans. Serv. Comput. 99 (2013) 1 (PrePrints).
[46] R. Soley, et al., Unified modelling language, http://www.uml.org 2012.
[47] Roy Thomas Fielding, Architectural styles and the design of network-based software architectures, (PhD thesis) University of California, 2000.
[48] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design patterns: elements of reusable object-oriented software, Professional Computing Series. Addison Wesley, Reading, MA, USA, 1995.
[49] P. Grefen, R. Eshuis, N. Mehandjiev, G. Kouvas, G. Weichhart, Internet-based support for process-oriented instant virtual enterprises, IEEE Internet Comput. 13 (6) (2009) 65–73.
[50] P. Grefen, H. Ludwig, S. Angelov, A three-level framework for process and data management of complex E-services, Int. J. Coop. Inf. Syst. 12 (4) (2003) 487–531.
[51] P. Grefen, H. Ludwig, A. Dan, S. Angelov, An analysis of web services support for dynamic business process outsourcing, Inf. Softw. Technol. 48 (11) (2006) 1115–1134.

[52] P. Grefen, R.R. de Vries, A reference architecture for workflow management systems, Data Knowl. Eng. 27 (1) (1998) 31–57.
[53] Z. Haibei, Y. Xu, The architecture design of a distributed workflow system, Distributed Computing and Applications to Business, Engineering Science (DCABES), 2012 11th International Symposium on, 2012, pp. 9–12.
[54] Y. Hoffner, H. Ludwig, C. Gülcü, P. Grefen, Architecture for cross-organizational business processes, Procs. 2nd Int. Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, 2005, pp. 2–11, (Milpitas, CA, USA).
[55] C.L. Iacovou, I. Benbasat, A.S. Dexter, Electronic data interchange and small organizations: adoption and impact of technology, MIS Q. 19 (4) (December 1995) 465–485.
[56] IASTA, SmartSource Platform, 2014.
[57] IBM. WebSphere. http://www-01.ibm.com/software/websphere/ 2013.
[58] J. Dietrich, J. Hiller, A. Kozlenkov, Mandarax, http://mandarax.sourceforge.net/ 2013.
[59] D. Jordan, J. Evdemon, A. Alves, A. Arkin, Business process execution language for web-services 2.0, http://www.oasis-open.org/committees/download.php/10347/wsbpel-specification-draft-120204.htm 2007.
[60] M. Kay, SAXON, http://www.saxonica.com/welcome/welcome.xml 2013.
[61] R. Kazman, M. Klein, P. Clements, ATAM: method for architecture evaluation: ATAM — architecture trade-off analysis method report, http://www.sei.cmu.edu/ata/ata_method.html 2002.
[62] Florian Kerschbaum, Philip Robinson, Security architecture for virtual organizations of business web services, J. Syst. Archit. 55 (4) (2009) 224–232.
[63] M. Klein, R. Kazman, Attribute-based architectural styles, Technical, Report CMU/SEI-99-TR-022 ESC-TR-99-022, 1999.
[64] L. Kutvonen, A. Norta, S. Ruohomaa, Inter-enterprise business transaction management in open service ecosystems, Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International, Sept 2012, pp. 31–40.
[65] Lea Kutvonen, Alex Norta, Sini Ruohomaa, Inter-enterprise business transaction management in open service ecosystems, EDOC, 2012, pp. 31–40.
[66] A. Lazcano, H. Schuldt, G. Alonso, H. Schek, WISE: process based E-commerce, IEEE Data Eng. Bull. 24 (1) (2001).
[67] C. Legner, The evolution of B2B e-services from first generation e-commerce solutions to multichannel architectures, J. Electron. Commer. Organ. 6 (2) (2008) 58–77.
[68] Z. Liming, M. Staples, V. Tosic, On creating industry-wide reference architectures, Enterprise Distributed Object Computing Conference, 2008. EDOC '08. 12th International IEEE, 2008, pp. 24–30.
[69] Duen-Ren Liu, Minxin Shen, Workflow modeling for virtual processes: an order-preserving process-view approach, Inf. Syst. 28 (6) (September 2003) 505–532.
[70] N. Mehandjiev, P. Grefen (Eds.), Dynamic Business Process Formation for Instant Virtual Enterprises, Springer, 2010.
[71] Microsoft, BizTalk, https://www.microsoft.com/en-us/biztalk/default.aspx 2013.
[72] S. Moser, A. Martens, K. Gorlach, W. Amme, A. Godlinski, Advanced verification of distributed WS-BPEL business processes incorporating CSSA-based data flow analysis, Services Computing, 2007. SCC 2007. IEEE International Conference on, July 2007, pp. 98–105.
[73] S.V. Nagabhushan, K.N. Subramanya, G.N. Srinivasan, Modeling and analysis for single item multi-attribute reverse auction, Advance Computing Conference (IACC), 2013 IEEE 3rd International, 2013, pp. 1533–1539.
[74] C. Nagl, F. Rosenberg, S. Dustdar, VIDRE-A distributed service-oriented business rule engine based on RuleML, Enterprise Distributed Object Computing Conference, 2006. EDOC '06. 10th IEEE International, 2006, pp. 35–44.
[75] A. Norta, Exploring dynamic inter-organizational business process collaboration, (PhD thesis) Technology University Eindhoven, Department of Information Systems, 2007.
[76] A. Norta, R. Eshuis, Specification and verification of harmonized business-process collaborations, Inf. Syst. Front. 12 (2010) 457–479, http://dx.doi.org/10.1007/s10796-009-9164-1.
[77] A. Norta, P. Grefen, Discovering patterns for inter-organizational business collaboration, Int. J. Coop. Inf. Syst. (IJCIS) 16 (2007) 507–544.
[78] A. Norta, M. Hendrix, P. Grefen, A pattern-knowledge base supported establishment of inter-organizational business processes, in: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, and ODBASE, Lecture Notes in Computer Science, vol. 4277, LNCS Springer, Montpellier, France, October 2006, pp. 834–843.
[79] A. Norta, L. Kutvonen, A cloud hub for brokering business processes as a service: a rendezvous platform that supports semi-automated background checked partner discovery for cross-enterprise collaboration, IEEE SRII Global Conference (SRII), 2012 Annual, 2012, pp. 293–302.
[80] Queensland University of Technology, YAWL home page, http://www.yawlfoundation.org/ .
[81] N.L. Olsson, XML content translator, http://www.nikse.dk/XmlContentTranslator/ 2013.
[82] openNMS, Event translator, http://www.opennms.org/wiki/Event_Translator 2013.
[83] ORACLE, Oracle Sourcing, http://www.oracle.com/us/products/applications/ebusiness/procurement/053985.html .
[84] Mike P. Papazoglou, Dimitrios Georgakopoulos, Service-oriented computing, Commun. ACM 46 (10) (2003) 24–28.
[85] P. Parviainen, M. Tihinen, M. Lormanms, R. van Solingen, Requirements Engineering: Dealing with the Complexity of Sociotechnical Systems Development, IGI Global, 2005.
[86] G. Premkumar, K. Ramamurthy, Sree Nilakanta, Implementation of electronic data interchange: an innovation diffusion perspective, J. Manage. Inf. Syst. 11 (2) (September 1994) 157–186.
[87] M. Proctor, K. Verlaenen, E. Tirelli, Drools, https://www.jboss.org/drools 2013.
[88] P. Reed, Reference architecture: the best of best practices, http://www.ibm.com/developerworks/rational/library/2774.html 2002.
[89] Manfred Reichert, Stefanie Rinderle, Peter Dadam, Adept workflow management system, in: WilM P. Aalst, Mathias Weske (Eds.), Business Process Management, Lecture Notes in Computer Science, vol. 2678, Springer, Berlin Heidelberg, 2003, pp. 370–379.
[90] Manfred Reichert, Barbara Weber, Aristaflow BPM suite, Enabling Flexibility in Process-Aware Information Systems, Springer, Berlin Heidelberg, 2012, pp. 441–464.
[91] W. Reisig, G. Rozenberg (Eds.), Lectures on Petri Nets I: Basic Models, Lecture Notes in Computer Science, vol. 1491, Springer-Verlag, Berlin, 1998.
[92] W. Reisig, G. Rozenberg (Eds.), Lectures on Petri Nets II: Applications, Lecture Notes in Computer Science, vol. 1492, Springer-Verlag, Berlin, 1998.
[93] IBM Research, Crossflow architecture description, Technical Report, ESPRIT Crossflow EP 28653, 1999.
[94] S. Ruohomaa, L. Kutvonen, Making multi-dimensional trust decisions on inter-enterprise collaborations, Proceedings of the Third International Conference on Availability, Security and Reliability (ARES 2008), IEEE Computer Society, Barcelona, Spain, March 2008, pp. 873–880.
[95] SAP, SAP sourcing application, http://scn.sap.com/community/sourcing 2014.
[96] M. Shaw, D. Garlan, Software Architecture: Perspectives on an Emerging Discipline, Prentice-Hall, Upper Saddle River, NJ, 1996.
[97] W. Shi, A sealed-bid multi-attribute auction protocol with strong bid privacy and bidder privacy, Security and Communication Networks, 2013.
[98] James Skene, Franco Raimondi, Wolfgang Emmerich, Service-level agreements for electronic services, IEEE Trans. Softw. Eng. 36 (2) (2010) 288–304.
[99] Ian Sommerville, Gerald Kotonya, Requirements Engineering: Processes and Techniques, John Wiley & Sons, Inc., New York, NY, USA, 1998.
[100] L.S. Sterling, K. Taveter, The Art of Agent-Oriented Modeling, The MIT Press, 2009.
[101] Xiao-han Sun, A secure english electronic auction protocol, Proceedings of the 2012 International Conference on Convergence Computer Technology, IC3T '12, IEEE Computer Society, Washington, DC, USA, 2012, pp. 223–225.
[102] S. Thatte, XLANG: Web Service for Business Process Design, 2003.
[103] Wei-Tek Tsai, Xin Sun, J. Balasooriya, Service-oriented cloud computing architecture, Information Technology: New Generations (ITNG), 2010 Seventh International Conference on, 2010, pp. 684–689.
[104] J. van Angeren, J. Kabbedijk, S. Jansen, K.M.S. Popp, A survey of associate models used within large software ecosystems, Proceedings of the Workshop on Software Ecosystems, 2011.
[105] H.M.W. Verbeek, T. Basten, W.M.P. van der Aalst, Diagnosing workflow processes using Woflan, Comput. J. Br. Comput. Soc. 44 (4) (2001) 246–279.
[106] F.B. Vernadat, Interoperable enterprise systems: principles, concepts, and methods, Annu. Rev. Control. 31 (1) (2007) 137–145.

[107] Yi Wei, M.B. Blake, Service-oriented computing and cloud computing: challenges and opportunities, Internet Comput. IEEE 14 (6) (2010) 72–75.
[108] Gerhard Weikum, Hans-J Schek, Concepts and applications of multilevel transactions and open nested transactions, Database Transaction Models for Advanced Applications, Morgan Kaufmann Publishers Inc, 1992, pp. 515–553.
[109] WFMC, Workflow reference model, Technical report, Workflow Management Coalition, Brussels, 1994.
[110] S.A. White, et al., Business Process Modeling Notation (BPMN) Specification, Version 1.0, Object Management Group, 2006. (http://www.bpmn.org).

Alex Norta is currently a research member at the Faculty of Informatics/TTU and was earlier a researcher at the Oulu University Secure-Programming Group (OUSPG) after having been a post-doctoral researcher at the University of Helsinki, Finland. He received his MSc degree (2001) from the Johannes Kepler University of Linz, Austria and his PhD degree (2007) from the Eindhoven University of Technology, The Netherlands. His PhD thesis was partly financed by the IST project CrossWork, in which he focused on developing the eSourcing concept for dynamic inter-organizational business process collaboration. His research interests include business-process collaboration, workflow management, e-business transactions, service-oriented computing, software architectures and software engineering, ontologies, mashups, and social web. At the IEEE EDOC'12-conference, Alex won the best-paper award for his full research paper with the title "Inter-enterprise business transaction management in open service ecosystems".

Paul Grefen has been a full professor in the School of Industrial Engineering at Eindhoven University of Technology (TU/e) since 2003, where he has been the chair of the Information Systems subdepartment since 2006. He received his Ph.D. in 1992 from the University of Twente. From 1992 until early 2003, he held assistant and associate professor positions in the Computer Science Department at the University of Twente. He was a visiting researcher at Stanford University in 1994. He has been involved in various European research projects as well as various projects within the Netherlands. He is a member of the editorial board of the International Journal of Cooperative Information Systems. He is an editor of the books on the WIDE and CrossWork projects, and has authored books on workflow management and e-business. He is a member of the Executive Board of the European Supply Chain Forum. His current research interests include architectural design of business information systems, inter-organizational business process management, and service-oriented business design and support. He teaches at the M.Sc. and Ph.D. levels at TU/e and at the executive level via the TIAS-NIMBAS business school.

Nanjangud C Narendra is a Chief Architect at Cognizant Technology Solutions, Bangalore, India. His research interests are in software engineering, Web Services, SOA and Cloud Computing. He has over 20 years R&D experience in Cognizant, IBM, Hewlett Packard and Motorola. He is the co-author of over 100 papers in international conferences and journals. He has also been a program committee member for several well-known international conferences, and a reviewer for several international journals. He is also member of Editorial Board of Service-Oriented Computing and Applications journal. He is a Senior Member of IEEE and ACM. He obtained his BTech from IIT Madras, India, and PhD from Rensselaer Polytechnic Institute, USA.