# An Agent-Oriented Method for Designing Large Socio-Technical Service-Ecosystems

Alex Norta, Msury Mahunnah, Tanel Tenso, Kuldar Taveter
*Department of Informatics*
*Tallinn University of Technology, Ehitajate tee 5*
*19086, Tallinn, Estonia*
*Email: alex.norta@gmail.com; msurym@gmail.com;*
*tanel.tenso@proekspert.ee; kuldar.taveter@ttu.ee*

Nanjangud C. Narendra
*Cognizant Technology Solutions*
*Bangalore, India*
*Email: ncnaren@gmail.com*

***Abstract*—The development of large socio-technical systems is now more feasible with the availability of service-oriented cloud computing although this results in a higher complexity level with respect to security and dependability in service-ecosystems. Additionally, the communication between humans and software is more complex too. Developing system architectures that comprise proactive agents are a means to tackle that complexity. However, there is a lack of socio-technical design methods for generating agent-based architectures that get deployed on platforms as a service in Clouds. Such a method must be easy to comprehend and enactable in a pragmatic way. To demonstrate the method, the paper uses a running case based on an emergency healthcare-provision scenario in homes for elderly in sparsely populated areas.**

***Keywords*-service; cloud; ecosystem; software engineering; socio-technical system; business-process aware;**

## I. Introduction

With the advent of service-oriented cloud computing [29] feasible scalability opportunities emerge for developing large socio-technical systems. Examples of such systems are individualized healthcare systems in sparsely populated areas. Such systems should be deployable as platforms as a service (PaaS) [9], which promise users savings in time and money via multi-tenancy. Designing such large socio-technical PaaS as service ecosystems poses challenges for existing design methods. Not only are such systems highly decentralized, but with the loose coupling of services, a peer-to-peer (P2P) way of operation is predominant. The resulting technical system complexity affects the quality goals of security and dependability [7], i.e., services that function correctly on their own behave incorrectly when they link to other services in an ecosystem in that they deadlock or livelock in loops. Secondly, in large PaaS-system development, the interactions between human- and software occur within the system rather than across the system boundary and are therefore more complex to capture.

Different researchers show that software agents and multi-agent systems [27], [40] efficiently support a P2P-way of operation. To that end, it is desirable to adopt an agent-oriented software engineering methodology [15], [19], [36] for designing PaaS-ecosystems since agents have the ability to detect changes, reason and subsequently act. In contrast, for socio-technical systems where software agents support humans, the Agent-Oriented Modelling (AOM) approach citesterling2009art includes features similar to the AOSE-methodologies but it is geared towards designing socio-technical systems consisting of humans and software that are respectively termed as human agents and man-made agents. However, also in the case of AOM a gap remains for designing the agent-oriented P2P architecture.

In this paper we address this gap by answering the research question of how to systematically create an architecture for PaaS- ecosystem deployment with a lightweight and simple to follow socio-technical and agent-based design method. We aim to propose a repeatable process for the problem domain analysis and architectural design of large socio-technical systems. To establish complexity-reducing separation of concerns, we deduce the following sub-questions. What are the identified steps of the design method? What is the role of AOM in that new method? What is the differentiating aspect that the new method introduces into AOM? We also explore in this paper the collaboration model that enables the P2P way of business collaboration, and the styles and patterns for a high-quality standard architecture.

The paper is structured as follows. Section II briefly describes the approach for developing architecture types. Section III gives an overview of our running case study from the emergency healthcare systems (EHS) domain. Section IV presents the analysis and platform-independent design of this healthcare scenario by specifying goal models for the socio-technical system, agent- and acquaintance models and knowledge model. Section V bridges the gap towards defining the architecture for PaaS-ecosystem deployment based on agent-oriented analysis and design models by establishing first a discrete collaboration model between the involved parties. Section VI presents an exploratory feasibility study to show with what technological means the architecture is implementable. Finally, Section VIII presents conclusions and provides directions for future work.

978-1-4799-5069-0/14 $31.00 © 2014 IEEE
DOI 10.1109/SERVICES.2014.50

## II. Development Approach

To tackle the complexity involved in EHS design, we aim for a separation of concerns that follows Figure 1. Depicted is a flow for architecture design with the assumption that domain knowledge and suitable styles and patterns lead to the establishment of a high-quality standard architecture for a specific domain. Specific context details about an application scenario lead to a concrete architecture in a sense of a standard-architecture instantiation.

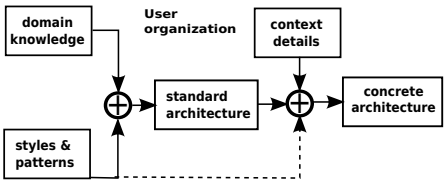


Figure 1. Flow in architecture-types development.

We assume the EHS is for elderly in homes in sparsely populated areas. Thus, the scenario is of a socio-technical nature where humans and software collaborate in a distributed peer-to-peer way using location-independent means for just-in-time information- and knowledge logistics provision. Next follows the general scenario description.

## III. Collaboration Scenario

We assume our EHS scenario is for elderly in homes in sparsely populated areas. Thus, the scenario is of a socio-technical nature where humans and software collaborate in a distributed peer-to-peer way using location-independent means for just-in-time information- and knowledge logistics provision. There exist several completed and on-going research efforts on the application of multi-agent systems (MAS) in the healthcare domain, e.g., personalized treatment to home care patients [23], decision making on insulin dosages [10], vital-signs monitoring [43]. Our approach is more generic and leverages the use of goal models [42].

Figure 2 shows the challenging information-logistics scenario of providing personalized emergency health care for elderly who live in homes under nurse supervision. A distributed, cloud-based mobile-service system is instrumental for enabling an instantaneous performance of multimedia supported health diagnosis via small wireless units. The computing cloud in the middle is pivotal for orchestrating and choreographing the heterogeneous, distributed ICT-infrastructure. Ensuring instantaneous health care provision in a remote location is challenging if it must be effective and efficient. Effective means, an elder is immediately kept alive and cured in a speedy way while efficient means the remote geographic location should still allow quick help delivery that remains affordable. Instantaneous health care as explained in Figure 2 is also challenging with respect to security and dependability. The analysis and design of

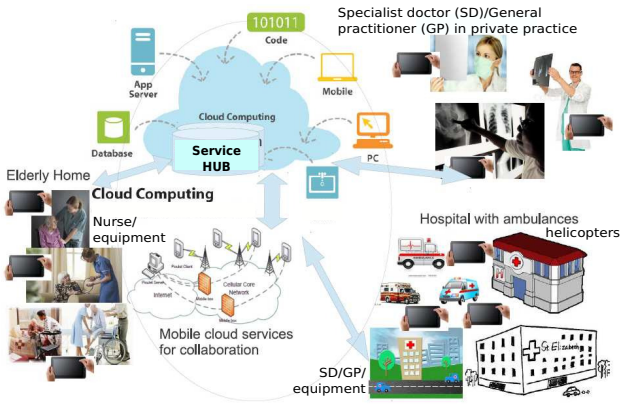


Figure 2. Elderly emergency-healthcare scenario.

this general scenario requires a specific methodology that considers three viewpoint aspects [42], namely, interaction, information and behaviour, as the sequel shows.

## IV. Domain Analysis and Design

In AOM we start analysing the problem domain of the socio-technical system by using a goal model. The objective of goal models is to serve as communication media between technical- and non-technical stakeholders for generating understandable domain knowledge.

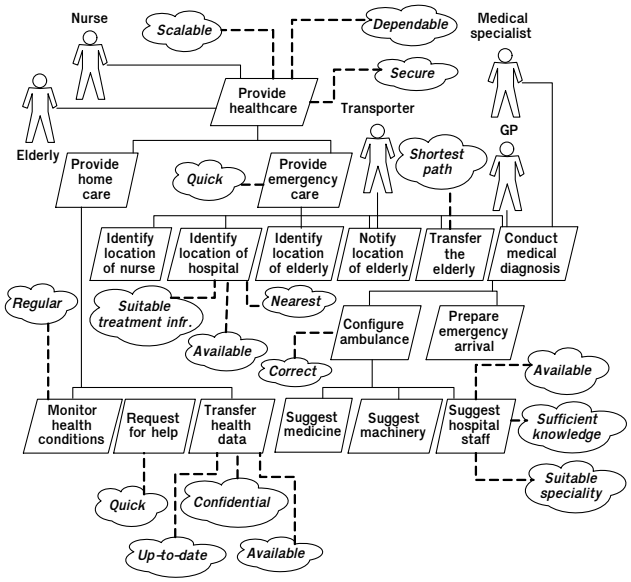


Figure 3. The goal model for the personalized emergency-healthcare system.

In the goal model of Figure 3, we first present the uppermost goal, viz., *provide healthcare* with the attached roles of elderly and nurse. The quality goals *dependable* and *secure* mean in the first case that the system has the ability to avoid

service failures that are more frequent and more severe than is acceptable. The main goal *provide healthcare* splits into two sub-goals: *provide home care* and *provide emergency care* in a *quick* way. The first comprises three goals, viz., *monitor health conditions* in a *regular* manner, *request for help* in emergency cases and *transfer of health data* for the continuous monitoring of elderly health and appropriate emergency setup at the hospital. The quality goal *quick* we attach to the *request for help* from the hospital with the right resources for treatment. Three quality goals *up-to-date*, *confidential* and *available* aim at maintaining the elderly patient's data security.

We describe the goal *provide emergency care* in Figure 3 with the sub-goals for identifying the locations of the elderly, nurse and appropriate hospital. The latter must be *available*, *nearest* and with *suitable treatment infrastructure*. The goal *notify location of elderly* we attach to the role *transporter*. The latter receives the address of an elderly home from the nurse role. The use of shortest path is important for the goal *transfer the elderly* from the home to the appropriate hospital. General practitioners (GP) carry out the goal *conduct medical diagnosis*, which may involve medical specialists. The goal also comprises *configure ambulance* with resources for immediate treatment and *prepare emergency arrival* at the hospital. Furthermore, the results of medical diagnosis influence the correct configuration of the ambulance which consists of the three sub-goals *suggest medicine*, *suggest machinery* and *suggest hospital staff*. The latter may involve *emergency nurse*, *emergency GP* and *emergency medical specialist*. For the configuration of the ambulance, we assign three quality goals to suggesting hospital staff in emergency cases, namely, *available*, *sufficient knowledge* and a *suitable speciality*.

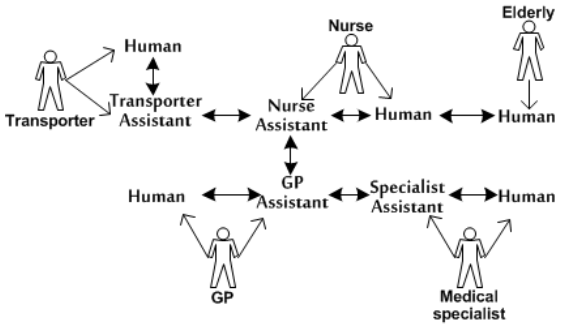


Figure 4. A merged agent and acquaintance model.

The responsibilities of identified roles above are either carried out entirely by man-made agents or both by man-made agents and human agents. The latter is a person such as an elderly, nurse, GP or medical specialist found in the healthcare domain. Man-made agents are intelligent digital assistants that we simply term assistants in the sequel. The agent model specifies the mapping of system roles to human agents or/and man-made agents. The design of interaction pathways between agents of the decided types we capture in the acquaintance model.

Figure 4 represents a merged agent model and acquaintance model. According to the merged model, the role nurse is mapped to nurse assistant and human, meaning some responsibilities of the role nurse are carried out by nurse assistant and other by human. The former perform activities such as regular monitoring of health condition by automatically collecting physiological data from wireless medical sensor devices attached to the elderly while the latter conducts physical activities such as engaging in requests for emergency help through physical interactions with the nurse assistant. The same approach holds for distributing responsibilities for the roles GP and Medical specialist while the role elderly is mapped to human agent only. The mapping of identified roles to relevant agent types completes the agent model. When an emergency occurs, the nurse assistant communicates with GP assistant to transfer up-to-date physiological data of the elderly. Thereafter, the GP assistant alerts the corresponding human GP for the purpose of taking necessary action which may include requesting an appropriately skilled medical specialist for further diagnosis. The actions by corresponding human agents, e.g., touching the button on the interface of a mobile device, trigger transfers of sensitive health data. Furthermore, the identified diagnosis results trigger interactions from the nurse assistant to the transporter assistant. The latter prompts a corresponding human agent to quickly acknowledge receipt of location information.

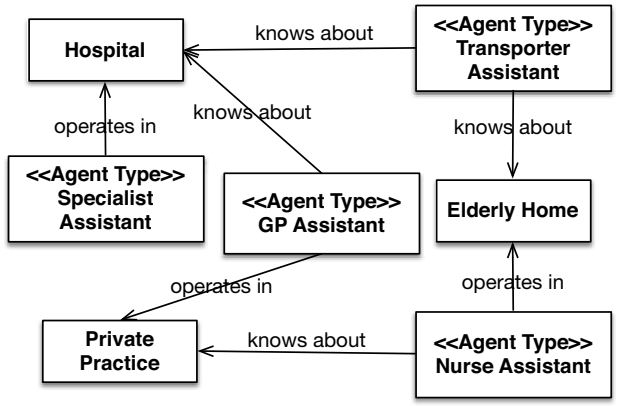


Figure 5. A knowledge model for EHS.

To fully gain insight into the EHS domain and link all knowledge together as represented on Figure 2, we need a knowledge model [42] that Figure 5 depicts for the EHS. According to this model, a nurse assistant operates in an elderly home and knows about private practices. The trans-

port assistant knows about the elderly home and hospitals. The GP assistant operates in a Private practice and knows about the hospitals. The specialist assistant operates in the hospital. This kind of knowledge sharing enables agents to communicate between each other as specified in Figure 4. The knowledge diagram is also a basis for the collaboration model.

Next, we further refine the design by focusing on how the parties involved in the EHS meaningfully engage in semi-automated collaboration.

## V. Dynamic Collaboration

Further detailing the EHS system, we must consider a model that is suitable for the P2P collaboration of the parties engaged. According to Section V-A, such a model must have a formal backing as to allow tool-supported verification of its correctness before enactment. In an emergency healthcare scenario, faulty collaboration models may result in a fatal outcome by losing valuable time, e.g., due to deadlocks or livelocks. Next, we deduce a standard architecture in Section V-B for our domain under investigation. Using affiliated styles and patterns discussed in Section V-C, a high-quality concrete-architecture instantiation is feasible.

### A. Collaboration Model

Based on a best-practice industry case [26] we use the structural properties of a formalized master/client collaboration model [30] and adopt it for the P2P scenario of this paper's EHS. In the latter's P2P case, we change the roles of the collaboration-model elements while leaving the formal properties intact. Figure 6 depicts the process elements that overlap with the master/client scenario. Note that the shared knowledge, namely elderly home, private practice and hospital we represent in Figure 5 as services. The depiction shows a business-network model (BNM) [38] that is a collaboration blueprint and resides with a set of other BNM in a Service-HUB [32] (see Figure 2). Each BNM represents the benchmark collaboration workflow for best treating respective emergency cases. Service types with roles depicted as ellipses populate the BNM and via interface protocols data exchange takes place along the P2P control flow. For enacting a BNM, concrete service offers from collaborating parties must match with the service types and roles. Finally, the BNM orchestrates system infrastructure from clouds that involve agents coordinating software as a service (SaaS) [13]; and also devices such as mobile devices, medical equipment, or telematics infrastructure. With respect to the structural properties in Figure 6, we describe them informally below and refer the reader to [30] for the complete formalization. The BNM we depict as a workflow net (WF-net) [2].

The service types with attached roles and service offers from collaborating parties in Figure 6 we depict as ellipses, which are process spheres. They are also WF-nets

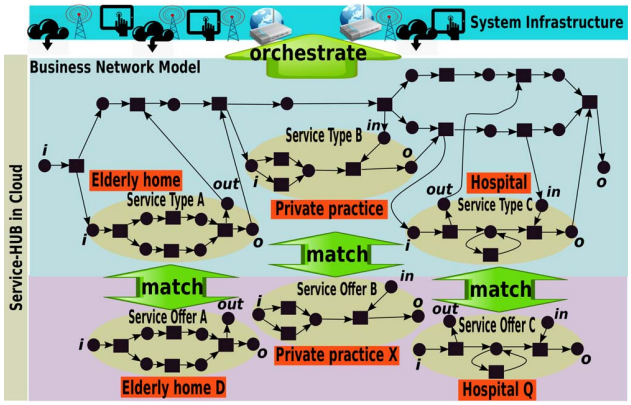


Figure 6. The collaboration model.

however, with optional so-called protocol states assigned (*in*-, *out*-labelled) for interfacing with the remainder of the BNM. A realization option for service types and -offers is to consider them as so-called worklets [3] that are self-contained sub-processes with associated selection rules. Furthermore, while we assume in Figure 6 a isomorphic matching between service types and -offers, it is possible to consider a matching of service offers with concrete services internally of collaborating parties. The latter internal services are subclasses compared to the service offers with respect to the runtime behaviour. For brevity, we omit a depiction in Figure 6 and also a detailed discussion in this paper but refer the reader to [17] for further details. For exploring the soundness [21], i.e., correct termination, of the overall BNM with populated service offers and possible refining concrete services, a collapsing function [30] generates one WF-net by replacing the service types in the BNM with concrete services of collaborating parties. For verifying the soundness of the resulting WF-net, tools such as Woflan [45] exist.

### B. Architecture for the EHS

Additionally to the high-level goal specifications of Figure 3, the collaboration model of Figure 6 suggests a set of functionalities that comprise the service-oriented architecture of an EHS. First, there must exist a service that facilitates the matching of BNMs with service types and roles on the one hand, and service offers from collaborating parties on the other hand. It should also verify soundness of populated BNMs. Second, the collaborating parties house internally a component for the distributed binding and P2P enactment of emergency cases. Third, with tool support, the parties must rapidly develop service offers and concrete services. Fourth, each collaborating party is capable of orchestrating its own internal legacy system for automating the collaboration. Finally, due to the heterogeneity of the collaboration, a translation service must exist for bridging the differences (technical, syntactic, semantic, pragmatic)

between collaborating parties.

Figure 7 depicts the resulting architecture in UML-component diagram notation that takes into account the above listed requirements. The *Service-HUB* is a trusted third party in the middle that satisfies the first requirement and is suitable for the rapid setup phase in an emergency scenario. Each party has on an external layer a *Collaboration_Middleware* for the technical binding after a successful setup that satisfies the second requirement. During the distributed BNM-enactment, the *Collaboration_Middleware* exchanges data via a security-ensuring gateway with the other parties. Thus, the *Collaboration_ Middleware* also comprises external workflow- and rules-enactment services that coordinate each other not only internally but also via the gateway with other parties. We assume there exists in each party a conceptual layer with a service for *Setup_Support* that satisfies the third requirement and comprises tools for not only rapidly internally designing services and rules with the help of pattern libraries [31], but also includes a local verification- and simulation service. Next, each party has an internal layer with a service for *Legacy_Management* that satisfies the fourth requirement and comprises local workflow- and rules enactment services that coordinate each other for the orchestration of Web-service wrapped internal legacy systems. Finally, the external- and internal enactment services exchange via a *Translator* service on the conceptual layer of each party to bridge the heterogeneous collaboration aspects. The *Translator* satisfies the final requirement and also connects on the conceptual layer with the *Setup_Support* service. Note that we omit in Figure 7 the conceptual- and internal layer with the exception of the *Hospital* due to space limitations.

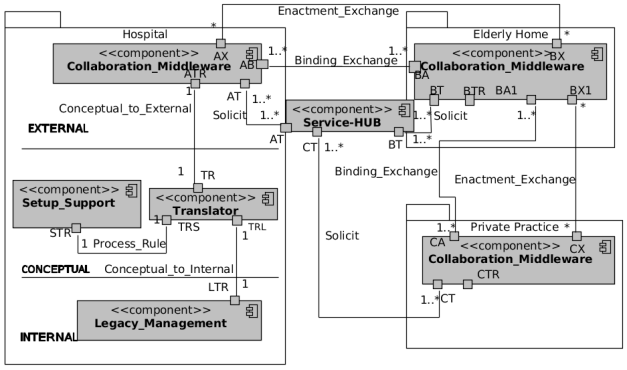


Figure 7. Architecture of EHS.

### C. Styles and Patterns

As presented in Section II, by assigning styles [11], [39] and patterns [11] to the EHS standard architecture, we help a high-quality instantiation into a concrete architecture based on best practices. Styles affect the topology of the entire architecture while patterns only affect parts of it.

A *layering style* [8], [11] for the domains of respective EHS-collaborators structures services into groups at a particular layer of abstraction, i.e., external, conceptual and internal. The style limits communication to each adjacent layer only.

The Service-HUB uses a *publish/subscribe style* [25] in which EHS-collaborators are publishers and subscribers. When a publisher submits new data, all subscribers receive a notification and automatically have data delivered. In the *Service-HUB*, the notifier forms the central component of a star topology where the publishers and subscribers are the leaves.

For the remaining way of data management, EHS-architecture comprises an *abstract-data-repository style* [25] that keeps the publishers and service-subscribers to shared data from having knowledge of each other's existence and the details of their implementations. Using a layering style and by interposing an intermediary protocol realizes the abstract data repository style. Note, that all domains of collaborating parties replicate the external layer of the EHS-architecture and are only accessible through the gateway service.

A *whole-part pattern* [11] aggregates the parts of a socio-technical collaboration. In the EHS-architecture, dedicated components exist on the external- and internal layer in the form of the global and local workflow- and rules engines. Additionally, the conceptual layer differentiates modelling support for business processes and business rules.

The *broker pattern* [11] in the EHS-architecture realizes the *Service-HUB* 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.

The already mentioned gateway service on the external layer realizes the *façade pattern* [18]. 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.

On the conceptual layer of the EHS-architecture, a *pipes-and-filters pattern* [11] facilitates establishing communication channels between the external- and internal layer via the conceptual layer in the form of the *Translator*. This pattern provides a structure for processing streams of data while each processing step is encapsulated in filter components. Hence, data passes through pipes between adjacent filters from the external layer to the internal layer and vice versa.

We next discuss which pre-existing services and artefacts allow for a rapid deployment of the EHS standard architecture into a concrete architecture.

## VI. Feasibility Evaluation

In accordance with Section II, there exist several technical instantiation options for concrete architectures. Starting with the assumption for the EHS that the BNM-enactment takes place in a distributed P2P manner, we refer the reader to [20] about a distributed multi-engine workflow system that is scalable and supports strong coupling between process management and business application using Web services. A distributed workflow in [14] targets redundancies in virtual enterprises in terms of carrying out the same operations, job, or functions. Furthermore, a distributed rules engine in [28] named ViDRE separates the implementation logic with exposing rules as Web services for accessibility across various rule engines.

For using legacy systems to be part of a larger service-based application system, loosely coupled services facilitate the establishment of highly dynamic business relations using service-oriented computing [16]. A pattern repository in [31] describes the storage of conceptually formulated patterns organized in a taxonomy belonging to different perspectives such as control-flow, data-flow, exception management. The knowledge base also stores implementations of respective patterns in industry standards such as BPEL [24], BPMN [1] for service specification, and so on.

Examples for internal workflow enactment exist from research and industry. A candidate example from research for a local WFMS is YAWL [33] that uses a Petri-net based business-process modelling notation as a foundation. With such semantic clarity, it is possible to verify processes for runtime soundness [2] in advance before enactment, i.e., one can check for control-flow issues such as deadlocks. The now industrial AristaFlow system [37] is another example originating from research. With the AristFlow system it is possible to verify control flow before enactment. Additionally, the formal notation allows to change the control flow of business processes at runtime if the enactment context requires so.

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 [5] 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 large data sets. Many additional open-source projects on top of Hadoop enhance that data-translation power, e.g., Hive [6], a data warehouse system for easy data summaries, ad-hoc queries in a SQL-like language called HiveQL to analyse large datasets. Also Impala [12] is an open source query engine for massively parallel data processing for Hadoop. Next, the open-source initiative Event Translator [35] comprises four features, namely first, an automated discovery of network services, secondly, receive 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 versions of extensible stylesheet language transformations (XSLT) such as Xalan [4]. Finally, if content in rules and processes use XML for specifications, there exists an open-source XML content translator [34] called Nikse.

## VII. Related Work

The need for formally backing socio-technical collaboration exists, as related work shows. To enhance the development of virtual enterprises (VE), the citation [22] presents a policy-based multi-agent management system. A hierarchical policy specification controls the behaviour of agents. A detection algorithms and resolution strategies describe the conflicts that hierarchical policies may cause. A policy administration tool simplifies the operations of these policies. Each enterprise registered in this tool has an agent assigned that execute policies of enterprise.

We view [22] as complementary to our work as they consider agents for VE-collaboration. Also the citation [41] discusses an approach to multi-agent system requirements engineering. So-called responsibility models serve as a basis to identify stakeholders and information requirements.

The citation [44] offers a service-oriented architecture for ontology-based multi-agent system negotiations in the context of a VE. A series of semantic ontology matching methods achieve interoperability among agents. Finally, an extended contract-net protocol provides basic guidelines for agents to achieve successful negotiations.

## VIII. Conclusion and Future Work

In this paper, we address the gap in socio-technical and agent-oriented methods for developing architectures that are deployable as large service-ecosystems. The method we demonstrate by using a running case for emergency healthcare provision in homes for elderly in sparsely populated areas. The method comprises a set of steps that involve specifying goal-, agent-, acquaintance and knowledge models followed by establishing first a discrete collaboration model between the involved parties for subsequently deducing a standard architecture. The latter reaches a high quality level by assigning best practice styles and patterns. We additionally partially provide context details with a technical focus that indicate how to rapidly deploy multiple concrete architectures that vary depending on case-specific societal context.

The agent-oriented design method comprises the following steps. It commences with capturing domain knowledge

by posing a socio-technical scenario of general formulation and of a high complexity-degree. Additional domain knowledge capture happens by modelling first the goals together with roles for subsequently deducing the merged agent- and acquaintance model. The formal, discrete collaboration model among participating parties completes the domain knowledge. The latter in combination with an assigned set of best-practice styles and patterns lead to the specification of a high quality standard architecture. This allows a pragmatic deducing of concrete architectures that tightly match with the needs of their respective detailed context.

The specific role of AOM in the method is to capture human-oriented aspects of engaging with the socio-technical system. The facts in the goal-, agent-, acquaintance and knowledge models guide the development of the dynamic collaboration model among participants. Specifically, the goal model captures the functional requirements, quality goals, roles of the socio-technical systems and the dependability among them. In combination with the captured quality goals, this is also instrumental for deducing a virtual-enterprise architecture of context-specific utility.

As future work, several options exist. First, the integration between domain knowledge with given styles and patterns must be further investigated as the AOM models currently influence primarily the discrete collaboration model. Secondly, the domain knowledge comprises an informal description of a collaboration lifecycle. It is important to formalize these collaboration lifecycles for building concrete service-ecosystems that are dependable and secure. Finally, there is an option to study the method by completing the context details with varying case-specific societal context for concrete-architecture generation. Thus, future work may use multiple cases for implementing an emergency healthcare system on different continents with deviating regulations, hi-tech infrastructure, skills sets, and so on, to study how they affect concrete architectures.

## ACKNOWLEDGEMENT

This work is partially supported by the IT Academy Programme for Information and Communication Technology Research of Tallinn University of Technology (13-09-00-1);

## REFERENCES

[1] *Business Process Modeling Notation (BPMN), Version 2.0*. Object Management Group, 2011. http://www.bpmn.org/spec/BPMN/2.0/.

[2] W.M.P. van der Aalst. Structural Characterizations of Sound Workflow Nets. Computing Science Reports 96/23, Eindhoven University of Technology, Eindhoven, 1996.

[3] M. Adams, A.H.M. Hofstede, D. Edmond, and W.M.P. Aalst. Worklets: A service-oriented implementation of dynamic flexibility in workflows. In Robert Meersman and Zahir Tari, editors, *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*, volume 4275 of *Lecture Notes in Computer Science*, pages 291–308. Springer Berlin Heidelberg, 2006.

[4] Apache. *Apache Xalan Project*. https://xalan.apache.org/, 2013.

[5] Apache. *Hadoop*. https://hadoop.apache.org/, 2013.

[6] Apache. *Hive*. https://hive.apache.org/, 2013.

[7] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33, 2004.

[8] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison-Wesley, 1998.

[9] M. Boniface, B. Nasser, J. Papay, S.C. Phillips, A. Servin, Xiaoyu Yang, Z. Zlatev, S.V. Gogouvitis, G. Katsaros, K. Konstanteli, G. Kousiouris, A. Menychtas, and D. Kyriazis. Platform-as-a-service architecture for real-time quality of service management in clouds. In *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*, pages 155–160, 2010.

[10] Stefano Bromuri, Michael Ignaz Schumacher, Kostas Stathis, and Juan Ruiz. Monitoring gestational diabetes mellitus with cognitive agents and agent environments. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 02*, pages 409–414. IEEE Computer Society, 2011.

[11] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-Oriented Software Architecture: A System of Patterns*. Chichester, UK: Wiley, 1996.

[12] Cloudera. *Impala*. http://www.cloudera.com/content/cloudera/en/products/cdh/impala.html, 2013.

[13] Michael Cusumano. Cloud computing and saas as new computing platforms. *Commun. ACM*, 53(4):27–29, April 2010.

[14] R. Davidrajuh. Distributed workflow based approach for eliminating redundancy in virtual enterprising. *The Journal of Supercomputing*, 63(1):107–125, 2013.

[15] Scott A DeLoach. Analysis and design using mase and agenttool. Technical report, DTIC Document, 2001.

[16] H. Erven, G. Hicker, C. Huemer, and M. Zaptletal. The web services-businessactivity-initiator (ws-ba-i) protocol: an extension to the web services-businessactivity specification. In *In 2007 IEEE International Conference on Web Services (ICWS 2007)*, page 216224, 2007.

[17] R. Eshuis and A. Norta. A framework for service outsourcing using process views. In *Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International*, pages 99–108, 2010.

[18] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Professional Computing Series. Addison Wesley, Reading, MA, USA, 1995.

[19] Paolo Giorgini, John Mylopoulos, and Roberto Sebastiani. Goal-oriented requirements analysis and reasoning in the tropos methodology. *Engineering Applications of Artificial Intelligence*, 18(2):159–171, 2005.

[20] Z. Haibei and Y Xu. The architecture design of a distributed workflow system. In *Distributed Computing and Applications to Business, Engineering Science (DCABES), 2012 11th International Symposium on*, pages 9–12, 2012.

[21] Kees Hee, Natalia Sidorova, and Marc Voorhoeve. Soundness and separability of workflow nets in the stepwise refinement approach. In W.M.P. Aalst and E. Best, editors, *Applications and Theory of Petri Nets 2003*, volume 2679 of *Lecture Notes in Computer Science*, pages 337–356. Springer Berlin Heidelberg, 2003.

[22] Jun Hu, Yinghui Song, and Ye Sun. Multi-agent oriented policy-based management system for virtual enterprise. *JSW*, 7(10):2357–2364, 2012.

[23] David Isern, Antonio Moreno, David Sánchez, Ákos Hajnal, Gianfranco Pedone, and László Z Varga. Agent-based execution of personalised home care treatments. *Applied Intelligence*, 34(2):155–180, 2011.

[24] D. Jordan, J. Evdemon, A. Alves, and 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.

[25] M. Klein and R. Kazman. Attribute-based architectural styles. *TECHNICAL REPORT CMU/SEI-99-TR-022 ESC-TR-99-022*, 1999.

[26] N. Mehandjiev and P. Grefen, editors. *Dynamic Business Process Formation for Instant Virtual Enterprises*. Springer, 2010.

[27] R. Merino, L. Lopez, F. Fernandez, A. Cholvi, and Vicent. Dante: A self-adapting peer-to-peer system. In Sam Joseph, Zoran Despotovic, Gianluca Moro, and Sonia Bergamaschi, editors, *Agents and Peer-to-Peer Computing*, volume 4461 of *Lecture Notes in Computer Science*, pages 31–42. Springer Berlin Heidelberg, 2008.

[28] C. Nagl, F. Rosenberg, and S. Dustdar. Vidre–a distributed service-oriented business rule engine based on ruleml. In *Enterprise Distributed Object Computing Conference, 2006. EDOC '06. 10th IEEE International*, pages 35–44, 2006.

[29] N.Antonopoulos and L. Gillam, editors. *Cloud Computing: Principles, Systems and Application*. Springer, 2010.

[30] A. Norta and R. Eshuis. Specification and verification of harmonized business-process collaborations. *Information Systems Frontiers*, 12:457–479, 2010. 10.1007/s10796-009-9164-1.

[31] A. Norta, M. Hendrix, and P. Grefen. A pattern-knowledge base supported establishment of inter-organizational business processes. In R. Meersman and Z. Tari, editors, *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, and ODBASE*, volume 4277 of *Lecture Notes in Computer Science*, page 834843, Montpellier, France, October 2006. LNCS Springer.

[32] A. Norta and 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. In *IEEE SRII Global Conference (SRII), 2012 Annual*, pages 293–302, 2012.

[33] Queensland University of Technology. YAWL Home Page. http://www.yawl-system.com.

[34] N.L. Olsson. *Xml Content Translator*. http://www.nikse.dk/XmlContentTranslator/, 2013.

[35] openNMS. *Event Translator*. http://www.opennms.org/wiki/Event_Translator, 2013.

[36] Lin Padgham and Michael Winikoff. Prometheus: A methodology for developing intelligent agents. In *Agent-oriented software engineering III*, pages 174–185. Springer, 2003.

[37] M. Reichert and B. Weber. Aristaflow bpm suite. In *Enabling Flexibility in Process-Aware Information Systems*, pages 441–464. Springer Berlin Heidelberg, 2012.

[38] S. Ruohomaa, P. Kaur, and L. Kutvonen. From subjective reputation to verifiable experiences augmenting peer-control mechanisms for open service ecosystems. In Theo Dimitrakos, Rajat Moona, Dhiren Patel, and D.Harrison McKnight, editors, *Trust Management VI*, volume 374 of *IFIP Advances in Information and Communication Technology*, pages 142–157. Springer Berlin Heidelberg, 2012.

[39] M. Shaw and D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline.* Upper Saddle River, NJ: Prentice-Hall, 1996.

[40] Onn Shehory. Robustness challenges in peer-to-peer agent systems. In G. Moro, C. Sartori, and M.P. Singh, editors, *Agents and Peer-to-Peer Computing*, volume 2872 of *Lecture Notes in Computer Science*, pages 13–22. Springer Berlin Heidelberg, 2005.

[41] Ian Sommerville, Russell Lock, and Tim Storer. Information requirements for enterprise systems. *CoRR*, abs/1209.5246, 2012.

[42] Leon S Sterling and Kuldar Taveter. *The Art of Agent-Oriented Modeling*. The MIT Press, 2009.

[43] Alberto Tablado, Arantza Illarramendi, Miren I Bagüés, Jesús Bermúdez, and Alfredo Goñi. An intelligent system for assisting elderly people. In *Foundations of Intelligent Systems*, pages 466–474. Springer, 2005.

[44] Xiaohuan Wang, T.N. Wong, and Gong Wang. Service-oriented architecture for ontologies supporting multi-agent system negotiations in virtual enterprise. *Journal of Intelligent Manufacturing*, 23(4):1331–1349, 2012.

[45] Haiping Zha, WilM.P. Aalst, Jianmin Wang, Lijie Wen, and Jiaguang Sun. Verifying workflow processes: a transformation-based approach. *Software and Systems Modeling*, 10(2):253–264, 2011.