NEXT GENERATION E-MARITIME SYSTEMS ENGINEERING

Maria A. Lambrou, Department of Shipping, Trade and Transport, University of the Aegean, 2 Korai Str, Chios 82 100, Greece, mlambrou@aegean.gr

Howard Foster, Department of Computing, Imperial College, 180 Queen's Gate, London SW7 2BZ, UK, howard.foster@imperial.ac.uk

ABSTRACT

The increasing adoption of large, distributed and highly dynamic ITS systems calls for effective approaches to ensure high reliability. We consider a SOA based service engineering framework as a robust engineering approach to the elaboration and analysis of functional and quality requirements, as well the formal testing of architectural solutions, in the context of emerging intelligent e-maritime systems development. A case of a maritime transportation planning service behaviour and interactions testing is presented using an open-source environment providing tools for e-services formal specification, verification and validation.

Keywords: Intelligent Transportation Systems, e-maritime, Ambient Intelligence, Service-Oriented Computing, Self-Management, Testing

INTRODUCTION

Intelligent Transportation Systems (ITS) are considered with respect to Ambient Intelligence (AmI) advances that account to adaptive, location and situation-aware transport services and novel service design methods. In this frame, we approach next generation maritime transport systems and enabling techniques for robust service design and service provision.

ITS service engineering techniques are viewed as evolving to interface with emerging software engineering and computer science techniques, such as formal specification techniques for service-oriented computing (SOC). SOC is a new paradigm based on autonomous, platform-independent computational entities (services) that can be described, published and categorized, and dynamically discovered and assembled for developing distributed, interoperable, self-configuring systems and applications.

In our paper, self-management, and service interactions and composition for emerging maritime freight ITS applications are examined and model-driven performance validation and testing techniques for formal analysis of synchronous/asynchronous service invocations of applications are proposed. Scenarios are modelled in UML, in the form of MSCs, and then compiled into the Finite State Process (FSP) algebra to concisely model the required behaviour. The approach is implemented in an open-source environment providing tools for specification, formal modelling, verification and validation of intelligent freight ITS services.

ITS SERVICE ENGINEERING

ITS typically refer to emerging technologies, and services, as well as the planning, operation, and control methods employed for modelling and implementing the transportation of persons and freight (Crainic et al, 2009).

Today, ITS regarding both passenger and freight services comprise a multitude of advanced applications (Wootton et al, 1995; Marchet et al, 2009) for road traffic management, personalized and context-aware services for intermodal travellers and in-vehicle services and intelligent infrastructures applications (vehicle-to-vehicle and vehicle-to-infrastructure applications).

Recently, shipping situation-aware service models are also included, regarding intermodal freight transport environments and advanced ICT enabled intelligent applications for vessel and co-modal vehicle and fleets management as well in particular monitoring, safety and security of goods services (Kia et al, 2000; Evangelista and Sweeney, 2006).

Giannopoulos (2004) identified three main key areas in which ICT can be used in freight transportation systems: freight resource management, terminal and port information management, freight and vehicle tracking and tracing, and back-office logistics.

According to Marchet et al (2008) freight transportation application types may be classified as: transportation management, supply chain execution, field force automation, and fleet and freight management. Freight ITS has also been classified into two broad classes: Commercial Vehicle Operations and Advanced Fleet Management Systems.

In the maritime field, main e-services categories and applications are understood as including port applications, shipping applications, ship and fleet management and transport logistics applications. Advanced ICTs are considered as enabling distributed platforms upon which each port, shipping company, or logistics operator exchange information and perform e-transactions with business partners, authorities and other networks.

According to Crainic et al (2009), freight ITS development proceeds along three major, parallel but complementary directions, namely vehicular and infrastructure developments; electronics, communications as well as the associated information technology and software; also models and algorithms required to process the data and transform it into intelligent advice for advanced system and fleet planning, management, operation and control. The advancement of the ITS field depends on the integration and co-evolution of the above aspects.

The increasing adoption of large, distributed, and highly dynamic ITS systems calls for effective approaches to ensure high reliability. Against this background, we postulate that promoting the intelligence of freight ITS, including maritime freight ITS applications, in view of robust design and testing engineering methodologies is important.

In a parallel world, today, software service engineering entails the consideration and application of a multitude of concepts, models, methods, and tools to design, develop,

deploy, test, operate, and maintain business-aligned and, very commonly, service oriented (SOA) software systems in a systematic and efficient manner (Zimmerman et al, 2004; Papazoglou and Heuvel, 2006; Papazoglou and Heuvel, 2007).
A distinguishing characteristic of service engineering is its holistic engineering approach. Service engineering demands an interdisciplinary approach towards the analysis and re-engineering of business processes, design of supporting software services, implementation, deployment, provisioning, monitoring, and service evolution. To this end, service engineering concepts, models, and methods are integrated and robust service engineering tools interoperable, adhering to open standards and offering integrated support for several stakeholders are employed.
Service engineering based on SOA paradigm principles and artifacts embraces a relatively new style of service design and development; primarily SOA applications are viewed as systemically mapped onto the business processes they realize (Arsanjani et al, 2007)

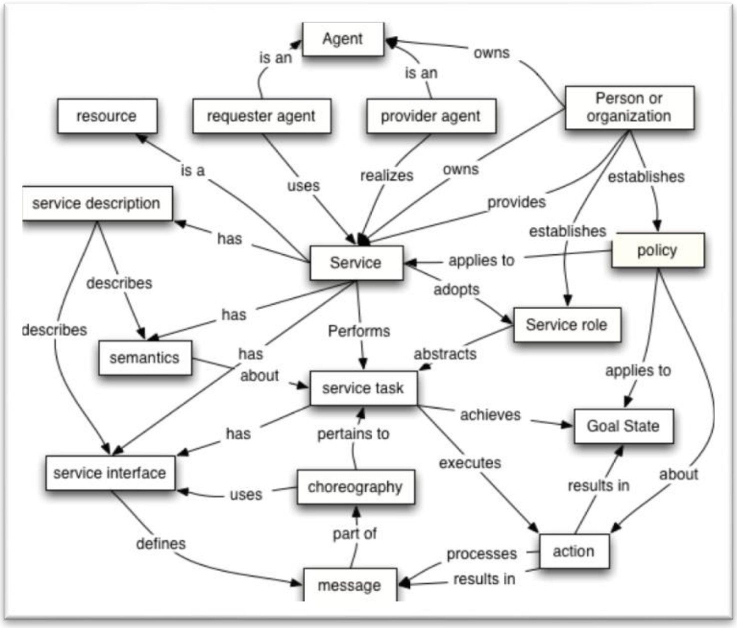

Figure 1 –The W3C Service-Oriented Model

A demanding area of research and practice examines the unification of concepts, artifacts and techniques from Business Process Management (BPM) and software engineering to ensure that applications, including ITS applications do not only meet system level Quality od Service (QoS) criteria, but also perform as specified in certain business process-level Key Performance Indicators (KPIs).
Business rules and semantics can provide an efficient way of expressing business requirements of an application, architected as a SOA; such business rules can then be used to assure that a system meets regulatory compliance conditions and other business policies (i.e. safety and security needs). Service engineering approaches and tasks ensure the efficient integration of such business rules into the overall service engineering lifecycle and programming model.
The service-oriented analysis and design methodology (SOAD) of IBM (Zimmerman et al, 2004) and the service-oriented design and development methodology proposed in Papazoglou and Heuvel, 2006) are representative of current service engineering

frameworks. The EU SENSORIA project also developed an integrated, robust service-oriented analysis, design and testing methodology (Foster et al, 2008).

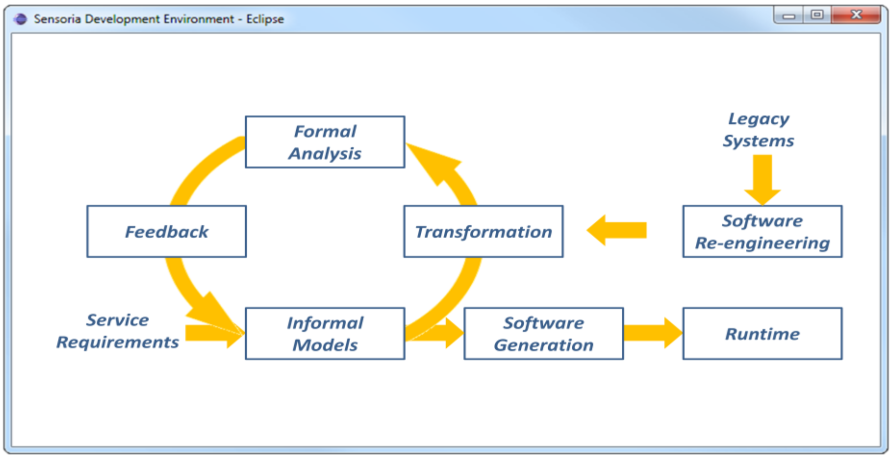

Figure 2 - Service engineering framework

The overall SOA service engineering framework strongly conforms with the standard software engineering approaches comprising the phases of requirement analysis, design, implementation, testing and maintenance. In particular, the following phases are comprised:
Service Analysis: A top-down approach to the application/business domain decomposition activities is undertaken. The business processes are decomposed into their sub-processes using high-level business use cases that enable identifying the required functionalities as services. Complementary, a bottom-up analysis of the existing infrastructure and legacy application systems takes place that serves as a basis for providing re-engineered service functionality.
Service Design: Service design refers to formal service identification, specification and implementation, towards deciding software components that implement the application/business logic required for identified services. In the formal service specification, the internal structure and information flow, the messaging and interaction/event specification, and further service/component interdependencies are given. SOA services are realized and exposed by the provider on a service registry, as also integrated into their business process. Service binding deals with the assignment of service requests to offers, using the request and offer specification are decided.
Testing and Validation: The constructed architecture is evaluated in terms of functionality, as well non-functional or qualitative characteristics, namely robustness, reliability, security, usability, efficiency. While this phase mainly corresponds to the traditional evaluation phase of software engineering, some additional criteria and respective enabling techniques are important, such as testing reusability of services, simulating and testing service efficiency of different providers.
Against this background, we consider a SOA based service engineering framework as a robust engineering approach to the elaboration and analysis of functional and quality requirements, as well the formal testing of architectural solutions, in the context of emerging maritime intelligent transportation systems development. A number of reasons necessitate such an approach: Firstly, service oriented ITS systems, including maritime specific systems are intrinsically distributed and highly dynamic, thus there is a need that functional and QoS requirements are assured for different deployment configurations; emerging maritime

intelligent transportation systems implement adaptive behaviours, by real time service configurations' modifications and service optimization; thus efficient testing has to deal with those possibly anticipated changing service configurations. The ownership of emerging maritime intelligent transportation systems is shared among different stakeholders. There is also an important trust issue regarding potential service providers operating in transport virtual marketplaces, where it is not possible to guarantee that information of service provides corresponds to QoS levels to be actually delivered. Thus formal and robust testing is required.

INTELLIGENT E-MARITIME SYSTEMS

In emerging ITS environments, inspired by the ambient intelligence (AmI) vision and principles (Weiser, 1991; Weber et al; 2005, Satyanarayanan, 2001), all transport actors, whether people or goods related, can be location-aware and transport mode situation-aware and communicate with each other. Intelligent objects and networks for transport and logistics can be integrated with intelligent mobile systems for people and goods, creating virtual transport services and environments. AmI can enable the formation of virtual enterprises, the smart and adaptive configuration of transport business processes, and the seamless interoperation of underlying information systems. AmI enables transportation companies and transportation third party organizations to participate in different business networks dynamically, based on interconnected and seamless work methods and workflows. Thus, ubiquitous, seamless and context-aware services across transport networks and terminals as well optimised mobility and freight service provision is achieved. While more intelligence is envisaged in the transport systems, in shipping, in particular, advances in surveillance and communication technologies, as well in mobile and wireless networks, electronics and ambient control and context-awareness models and algorithms enable more innovative, efficient and reliable shipping and co-modal transport planning, execution, monitoring and real time optimization of services (Lambrou et al, 2008).

In this context, an e-maritime architecture incorporates administrative applications such as e-customs, security and safety management, legislation and regulation compliance, shipping applications, port applications, and transport logistics applications, providing an inventory of operations and e-services that will be part of and evolve into the next generation e-maritime service infrastructure. We argue that a next generation AmI oriented e-maritime infrastructure architecture should be viewed as a complex, adaptive and self-managed system that will provide not only technological services, but also shared information and knowledge, thus enabling collaboration between public and private parties and advancing e-maritime strategic goals.

An AmI oriented e-maritime SOA is considered to empower maritime stakeholders to define and co-develop e-maritime collaborative environments, in a manner that reflects stakeholders' interests, perceptions and aspirations regarding next generation adaptive, context-aware and anticipatory e-maritime services.

Emerging maritime freight ITS systems set new challenges for the methodologies and technologies for service and application development (Aarts and de Ruyter, 2009). Service architectures, platforms development, tools and enabling techniques, form an environment for the development of innovative freight maritime ITS services. New approaches to existing paradigms, such as architectural frameworks, and autonomic systems can be integrated to engineer new categories of freight ITS services.

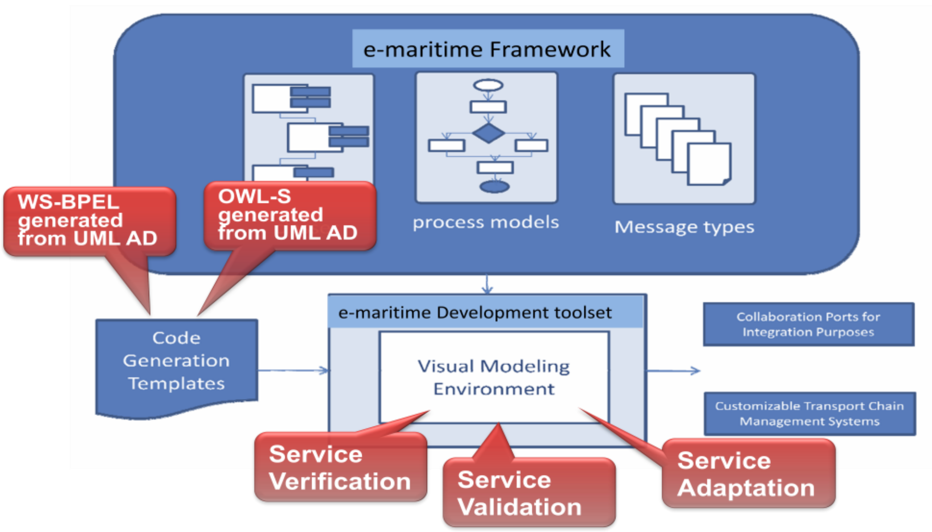

Figure 3- Architectural model of an e-maritime SOA environment

In particular, next generation e-maritime infrastructures will be advanced by enriching e-maritime SOA management with capabilities to obtain self-management (Kreger and Studwell, 2005). Adaptive applications (Kephart and Chess, 2003) have been recognised as viable solutions for large scale, distributed systems. Autonomic and self-managed solutions are being experimented in several application domains, but not yet extensively in service-oriented applications. We consider autonomic, self-managed system design concepts, methodologies and architectures in the context of an e-maritime SOA development. Applicable, novel techniques regarding the classic sense-plan-act control loop for deployment of self-managed, service-oriented maritime e-services are considered in the maritime SOA development. This self-adaptive control loop approach should be customised for each set of maritime e-services to comply with a specific service contract type and respective business policies and rules. To this end, descriptions of maritime e-services' interfaces, capabilities, behaviours, and service composition aspects are considered in the light of autonomic computing principles, as based on policies (business level policies, process level policies, and individual component level policies) (Yu and Lin, 2005).

SERVICE ENGINEERING OF AN E-MARITIME PLATFORM

An e-maritime platform requirements engineering, in particular, consists an important area crossing the boundaries of service analysis and service design and is concerned with the identification of maritime stakeholders' needs of the future system, the specification of services and constraints that satisfy these needs, and the assignment of the resulting requirements to services or/and software components. Functional requirements describe what functions must be provided to satisfy the stakeholders' needs, quality requirements describe how well the functions must be provided to satisfy these needs. Examples of quality requirements include concerns such as security, performance, reliability, availability, maintainability, and scalability as well transport service provider reputation, business relationships with transport service providers, regulatory compliance relationships etc.

Requirements engineering for next generation e-maritime systems design cannot be accomplished through the development of scenarios and the translation of use cases into system requirements, only. Novel service models based on AmI principles that dictate system functionality design can be determined via reliable testing and validation techniques providing proof of concept. Scenarios raise the need to periodically re-test the service designed, to ensure that they still meet functional and non functional requirements. To this end, Nitto et al. (2007) propose to complement service descriptions with a facet providing test cases, in the form of XML-based functional and non functional assertions, that is quality of service and service level agreement attributes that can be negotiated with the potential service users.

An e-maritime SOA combines elements from various related disciplines such as business process modeling and management, software architectures, component-based development, object-orientation, Enterprise Application Integration, distributed computing, and systems management.

SOA architectural design dictates the elicitation of the service requirements during service analysis. E-maritime service requirements analysis include functional requirements captured in narratives, use cases, and business process models, but also non-functional requirements such business policies attributes (Chung et al, 2000). Subsequently, design decisions are made in order to satisfy the service requirements during architectural design. A number of architectural views covering different design aspects are chosen and populated iteratively and incrementally during the architectural design. An e-maritime SOA conceptual architecture can be presented along a view-based approach including a data view, a functional view, a deployment view, a modelling view, and a scenario view. Furthermore, an e-maritime SOA conceptual architecture can be translated into a technical architecture describing how the technical framework is decomposed into e-maritime SOA software components, the main component interfaces, and the composition, scenario workflows among them, that is services and service composition models, methodologies and tools of the actual development and integration of the e-maritime SOA framework.

Enterprise integration patterns can be used to let consumers and providers of software services exchange messages via the SOA middleware. A service registry serves as directory of service providers available to respond to service consumer requests. The service request and response message formats can be specified in the service contract. Workflow concepts can guide service composition issues. The design and configuration of middleware such as ESBs (responsible for request, routing, adaptation, and mediation), workflow and process orchestration engines (facilitating service composition), and service registries (supporting provider lookup) are central parts of the service design phase (Pistore et al, 2005; Medjahed et al, 2003). Detailed service consumers and providers (actors) taxonomies are designed, developed, and instantiated into an e-maritime SOA infrastructure.

During service and architectural evaluation, testing and validation techniques and tools that examine and ensure that the service requirements are satisfied in an optimal manner are employed.

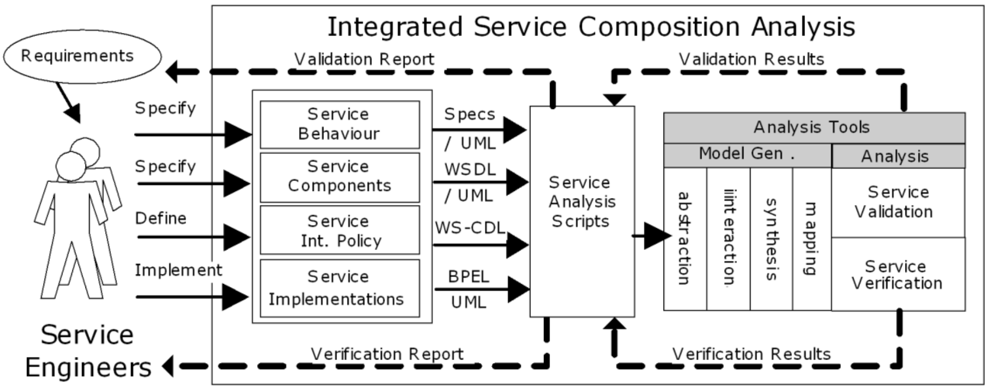

Figure 4—Service engineering of an e-maritime SOA environment

For modelling e-maritime SOA services there are several applicable UML profiles proposed in Ricardo et al, 2005. These profiles generally provide a set of stereotypes that represent features of service artifacts, including a service specification (interface), and orchestrated collaboration (behaviour specifications). What is generally missing from these existing profile approaches is the ability to identify the requirements and capabilities of services and then to elaborate on the dynamic changes anticipated for adaptation or self-management, as stemming from maritime AmI scenarios. Planning techniques, generally with the specification of a guiding policy with goals of service state, can assist the design of dymanic e-maritime service compositions (Kavakli and Loucopoulos, 2005).

Service Level Agreements for an e-maritime SOA

We recognize that an e-maritime SOA platform development touches upon existing interests, practices, and positions within the organizations which participate in the maritime value chain. We focus on identifying and assessing relevant factors and relationships that influence the successful operation of an e-maritime service infrastructure. We conceive these factors as embedded in the kind of agreements which organizations should develop in order to exchange information and perform electronic transactions over the e-maritime SOA platform: a) policy and administrative agreements referring to the interests and power relationships involved b) technological agreements which refer to (i) the definition of the information (standardized and formalized) to be exchanged, (ii) the use of ICT to support this exchange, and (iii) the management and control of the use of ICT. c) Economic agreements which refer to the specification and allocation of costs and benefits related to the exchange of information and the use of ICT. d) Legal arrangements which refer to specific rights and obligations laid down in rules and regulations, such as security and environmental protection and to more fundamental rights, such as privacy. In an e-maritime SOA environment, the nature and purpose of these agreements (complexity, static/dynamic) as well as the degree of their specification are examined.

The content of a Service Level Agreement (SLA), intended to be a formal agreement between transport service providers and their customers/service users, may vary for different services but typically entails clauses including non-functional and QoS requirements and penalties if QoS requirements are not satisfied (Bhatti, and Knight, 1999). E-maritime services typically expose both functional and non-functional properties (NFPs), as explained above. Important non-functional properties of the e-maritime services include properties such as cost, adherence to standards and rules and regulations and business obligations on the customer and provider side. QoS is traditionally used to refer specifically to infrastructure/platform performance and reliability characteristics; in the context of an e-maritime SOA, a SOA SLA approach is considered so as to allow for provisions for a wide range of maritime-specific business service properties to be formally agreed and negotiated upon (including, e.g. dependability, security, trust, etc.) and over the e-maritime SOA platform (Mukhija et al, 2007). The SLA concept, as considered, refers to properties of system components at different levels of granularity (e.g. infrastructure, process, application). The specification, enforcement and management of SLAs is directly connected to the tasks of modeling, provisioning and managing service related tasks, namely service discovery, composition, negotiation and monitoring based on NFPs (Jin and Wu, 2005; Paschke and Bichler, 2008; Mahbub and Spanoudakis, 2008), as SLAs provide a competitive mechanism that offers the service user an assurance that the services provided by the transportation service provider will operate within an acceptable/agreed range, particularly regarding the NFPs. Likewise, SLAs can serve an important role for the maritime transportation service providers regarding business planning and legal compliance aspects. Enforcing e-contracts by allowing autonomous supervision of service status and management based on efficient e-maritime SLA specification, monitoring and operation is proposed.

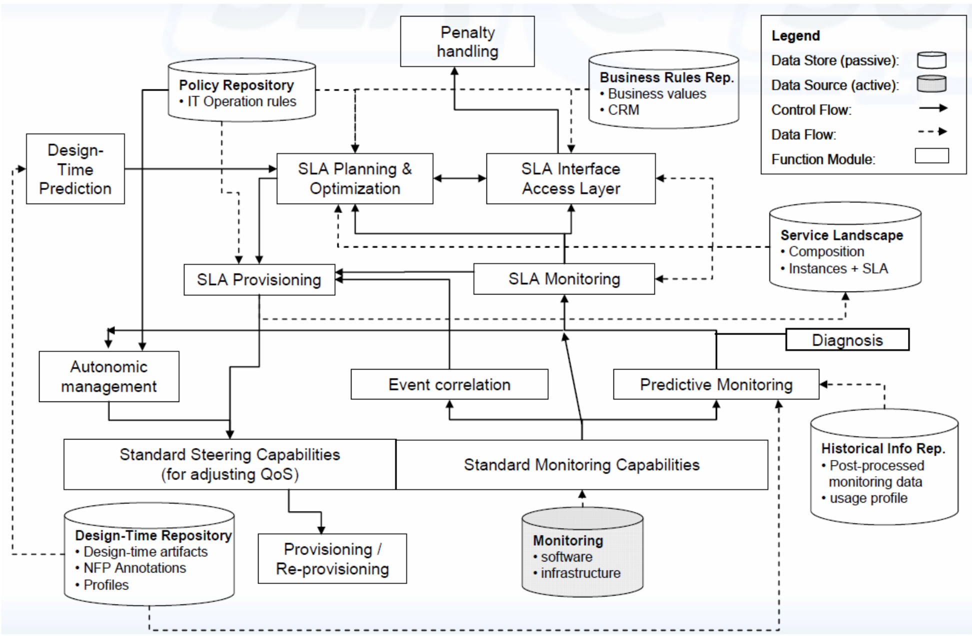

Figure 5- A SOA SLA approach towards robust e-maritime systems (SLA@SOI)

In particular, service contracts that capture e-maritime business and system goals and constraints (pre- and post-conditions and invariants) are necessary. Enriching the service interfaces with additional semantic information such as scenarios allows a more robust and stable service composition (behavioural contract). Designing service contracts with SLAs between transport service consumers and transport service providers which allow service consumers to express the expected service attributes and service providers to specify the available policy contracts is thus proposed. Machine-readable contracts allow the ESB and service composition middleware to collaborate and provide efficient service realization. A SOA component may expose such a contract. One of the key elements of SOA service engineering techniques is to use the principles of built-in testing allowing for services to contain their own test specification and enabling their run-time verification. Since each abstract service in a workflow can be bound to a set of possible concrete services (equivalent from functional point-of-view, but with different non-functional characteristics), there might be particular combinations of service bindings that can cause SLA violations; thus SOA testing for SLA is important.

Approach

To achieve this analysis, we consider an approach to input service requirements and capabilities by building service architecture specifications in a higher level architectural notation (such as UML). Attributed to these service architecture specifications are protocols for each of the service components used, detailing their required and provided services and the interface specification. The inputs mentioned previously are then transformed to architecture and behavioural models. Service architecture specifications are transformed modelling each service component, their required and provided services and their bindings between service instances. A Labelled Transition System (LTS) is obtained as result of transformation and compilation of both architecture and behaviour model transitions. Also properties to use as correctness checks against the system models are generated.

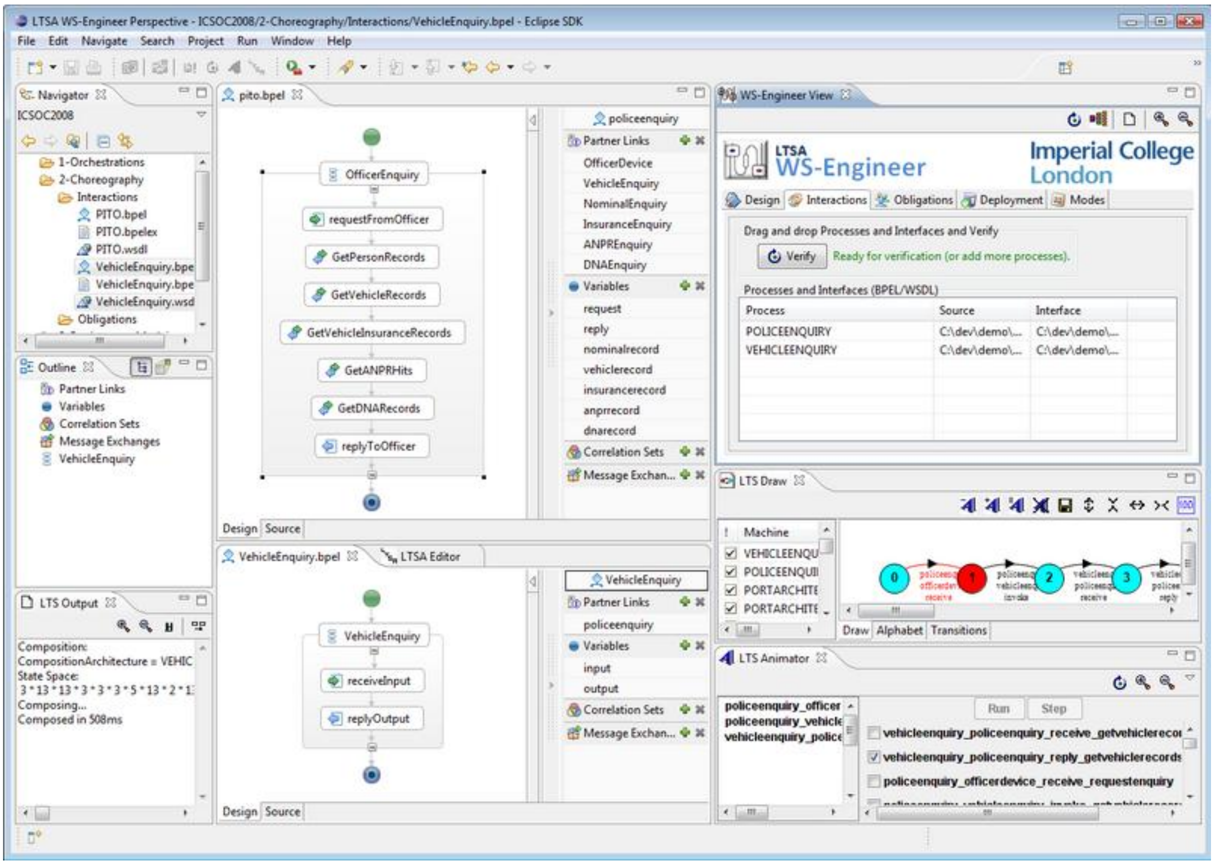

Figure 6 – Tool support for e-maritime service testing and validation

Scenarios are modelled in UML, in the form of MSCs, and then compiled into the Finite State Process (FSP) algebra to concisely model the required behaviour. The approach is implemented in the LTSA WS-Engineer environment providing tools for specification, formal modelling, verification and validation of e-maritime services.

Scenario

Service adaptation and constraining changes to e-maritime architecture and services, identifies both functional and non-functional variants on the specification. An SLA oriented QoS Profile can be used to describe the required SLA when connecting a particular service partner (of a particular type and offering similar specifications of usage). Architectural constraints may be specified in the Object Constraint Language (OCL) or another constraint based language. Service Behaviour requirements are attributed to each of the service components in each architectural version. In addition to the interface specification assigned, we can specify what behaviour the service fulfils. This describes the behaviour of required and provided interfaces, in that the sequence of the interface protocol is directly given. An example of a maritime transportation planning service behaviour and interactions testing is given.

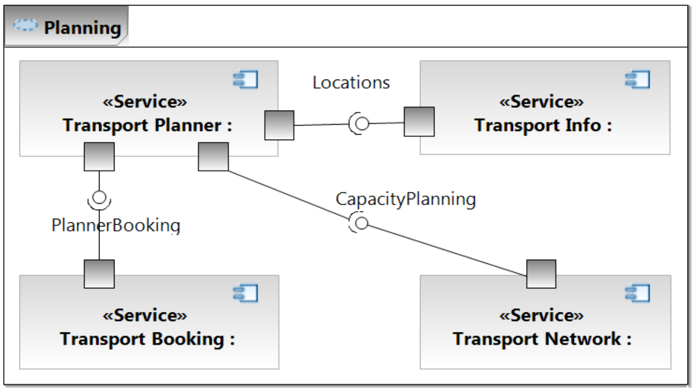

Figure 7 - e-maritime architecture testing: Component Structure Diagram of the Planning service

SUMMARY AND DISCUSSION

A necessary precursor to the success of a large-scale e-maritime system is a thorough analysis of the needs of all affected agencies, organizations and business groups. A strategic alignment approach to the expectations of each of these stakeholders groups defines the robustness of the design of a large scale e-maritime SOA compliant platform. Stakeholders' motivation to collaborate in an emerging intelligent e-maritime system encompass goals concerning increased benefits or cost reductions for their respective organizations. In this context, in a requirements service engineering phase, perceived benefits and costs should be addressed, including intangible (positioning, authority and common interests) or tangible ones (return on investment, efficiency) to be realized in the short or long term. Transactional, lower-level goals as well transformational and strategic goals involving interorganizational structures and operational processes should be also assessed against maritime stakeholders' views, perceptions and expectations. Thus, in a requirements service engineering phase, next-generation business models and services scenarios and respective stakeholders' incentives for participating in a e-maritime system within varying maritime and multimodal transport, public and private settings should be identified. Policy incentives include complying with foreseen legislative or regulatory requirements; requests from external oversight bodies; norms within maritime organizational cultures; maintaining key relationships; and responding to crises and other critical events. Technical incentives refer to concerns about system architecture, hardware, software, data management, standards, and sourcing criteria. Operational incentives derive from expected improvements in organizational and interorganizational processes, and economic motivation includes reducing costs or realizing economies of scale. These applicable incentive categories are considered to assess the alignment potential of stakeholder motivation within an intelligent AmI inspired collaboration environment. The related technical vision is to use the paradigm of SOA for managing a complete e-maritime service environment in correlation with SLAs which are defined at business and technical level. With SOA solutions, the difficulty of thoroughly testing a system increases as service-oriented organizations face considerable changes, both at the system and the business/organization model level. With automated service discovery and composition the exact configuration of a system is not completely known. Reliability and testing challenges derive primarily from the intrinsically distributed and highly

dynamic nature of emerging e-maritime systems, where a multitude of stakeholders and roles between service users, providers, and system owners arise.
Developing and using robust formal testing and validation techniques for the emerging e-maritime architectural solutions, consist a critical dimension for their future market adoption.

ACKNOWLEDGEMENTS

This work has been partially funded by the SENSORIA EU FP6 IST project, and by the SLA@SOI EU FP7 Collaborative Project. Also, partially supported by the eFREIGHT 7FP DGTREN Project. The approach and concept has been long and creatively discussed also with Prof. Michael G.H. Bell and Dr. Panagiotis Angeloudis of PORTeC, Imperial College.

REFERENCES

Aarts E.H.L., de Ruyter, B (2009). New research perspectives on Ambient Intelligence. Journal of Ambient Intelligence and Smart Environments, 1 , 5–14.

Arsanjani, A., Zhang, L.-J., Ellis, M., Allam, A. and Channabasavaiah, K. (2007). Design an SOA Solution Using a Reference Architecture. IBM developerWorks, IBM Corporation, http://www.ibm.com/developerworks/library/ar-archtemp/.

Bhatti, S.N., Knight, G. (1999). Enabling QoS adaptation decisions for Internet applications. Computer Networks, 31 (7), 669–692.

Chung L., Nixon B.A., Yu E., Mylopoulos J (2000). Non-functional requirements in software engineering. Kluwer, Dordrecht

Crainic T.G. et al. (2009). Intelligent freight-transportation systems: Assessment and the contribution of operations research. Transportation Research Part C, 17, 541–557

Evangelista, P. and Sweeney, E. (2006). Technology usage in the supply chain: the case of small 3PLs. The International Journal of Logistics Management, 17 (1), 55-74.

Foster, H., Uchitel S., Magee J., and Kramer, J. (2008). Leveraging modes and uml2 for service brokering specifications. in Proceedings of 4th Model-Driven Web Engineering Workshop at MoDELS 2008.

Giannopoulos, G.A. (2004). The application of information and communication technologies in transport. European Journal of Operational Research, 152, pp. 302-20.

Jin, H., Wu, H. (2005). Semantic-enabled Specification for Web Services Agreement. International Journal of Web Service Practices, 1(1), 13-2.

Kavakli P., Loucopoulos P. (2005). Goal modeling in requirements engineering: analysis and critique of current methods. In: Krogstie J, Halpin T, Siau K (eds) Information modeling methods and methodologies (Advanced Topics of Database Research), IDEA Group

Kephart J. O., Chess D.M. (2003). The Vision of Autonomic Computing. IEEE Computer, 36 (1), 41-50.

Kia, M., Shayan, E. and Ghotb, F. (2000), “The importance of information technology in port terminal operations”, International Journal of Physical Distribution & Logistics Management, Vol. 30 Nos 3/4, pp. 331-44.

Kreger H., and Studwell, T. (2005). Autonomic Computing and Web Services Distributed Management. IBM white paper. http://www.ibm.com/developerworks/autonomic/library/ac-architect/

Lambrou, M.A., Fjørtoft, K.E., Sykas, E.D., and Nikitakos, N. (2008). Ambient intelligence technologies in support of shipping markets' operations. Telematics and Informatics. 25(2), 72-83.

Mahbub K., and Spanoudakis, G. (2005). Run-time Monitoring of Requirements for Systems Composed of Web Services: Initial Implementation and Evaluation Experience. In Proceedings IEEE International Conference on Web Services 2005.

Marchet G., Perego A., and Sara Perotti S. (2009) An exploratory study of ICT adoption in the Italian freight transportation industry. International Journal of Physical Distribution and Logistics Management, 39 (9), 785-812.

Marchet, G., Mizzi, M., Perego, A. and Perotti, S. (2008). Improving performances through ICT in the freight transportation industry. In Proceedings of the Logistics Research Network, 2008, Liverpool.

Medjahed, B., Bouguettaya, A., and Elmagarmid, A.K. (2003) Composing web services on the semantic web", VLDB Journal, 12 (4), 333–351.

Mukhija, A., Dingwall-Smith, A., and Rosenblum, D. S. (2007). Qos-aware service composition in dino. In Proceedings of the Fifth European Conference on Web Services, Washington, DC, USA, 2007.

Nitto, Di E., Penta, Di M., Gambi, A., Ripa, G., Villani, M.L.(2007). Negotiation of service level agreements: An architecture and a search-based approach. In: Kraamer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, 4749, 295–306. Springer, Heidelberg.

Papazoglou, M. P., and Heuvel, (2006). Service-oriented Design and development Methodology. International Journal of Web Engineering and Technology, 2(4), 412 – 442.

Paschke, A., Bichler, M. (2008). Knowledge representations concepts for automated SLA Management. Decision Support Systems, 46 (1), 187-205.

Pistore, M., Marconi, A., Bertoli, P., and Traverso, P. (2005). Automated composition of web services by planning at the knowledge level. in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2005.

Ricardo, P.M., Machado, J., Fernandes, J.M., and Rodrigues, H. (2005). Transformation of uml models for service-oriented software architectures. in In Proceedings of the 12th IEEE International Conference and Workshops on Engineering of Computer-Based Systems, Washington, DC, USA, 2005.

Weiser M. (1991). The Computer for the Twenty-First Century. Scientific American 165 (3), 94-104.

Weber, W., Rabaey, J., and Aarts, E. (Eds.) (2005). Ambient intelligence. Berlin: Springer

Wootton, J.R., Garcia-Ortiz, A. and Amin, S.M. (1995). Intelligent transportation systems: enabling technologies> Mathematical and Computer Modelling, 22 (4), 11-81.

Yu, T., and Lin, K. J. (2005). Service Selection Algorithms for Web Services with End-to-end QoS Constraints. Journal of Information Systems and e-Business Management, 3 (2), 103-126.

Zimmermann, O., Krogdahl, P., and Gee C. (2004). Elements of Service-Oriented Analysis and Design - An Interdisciplinary Modeling Approach for SOA Projects. http://www-128.ibm.com/developerworks/library/ws-soad1/. IBM developerWorks.