<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *IMBSA 2017*.

Citation for the original published paper:

Chen, D., Lu, Z. (2017)

A Model-based Approach to Dynamic Self-Assessment for Automated Performance and Safety Awareness of Cyber-Physical Systems.

In: Marco Bozzano, Yiannis Papadopoulos (ed.), *Model-Based Safety and Assessment - 5th International Symposium, Trento, Italy, September 11–13, 2017* (pp. 227-240). Springer

LNCS 10437 Lecture Notes in Computer Science series, Springer

<https://doi.org/10.1007/978-3-319-64119-5>

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-215952>

A Model-based Approach to Dynamic Self-Assessment for Automated Performance and Safety Awareness of Cyber-Physical Systems

DeJiu Chen¹ Zhonghai Lu²

¹ Mechatronics, Machine Design, School of ITM, KTH Royal Institute of Technology, SE 10044 Stockholm, Sweden

e-mail chendj@kth.se

² Electronics and Embedded Systems, School of ICT, KTH Royal Institute of Technology, SE 164 40 Kista, Sweden

e-mail zhonghai@kth.se

Abstract.

Modern automotive vehicles represent one category of CPS (Cyber-Physical Systems) that are inherently time- and safety-critical. To justify the actions for quality-of-service adaptation and safety assurance, it is fundamental to perceive the uncertainties of system components in operation, which are caused by emergent properties, design or operation anomalies. From an industrial point of view, a further challenge is related to the usages of generic purpose COTS (Commercial-Off-The-Shelf) components, which are separately developed and evolved, often not sufficiently verified and validated for specific automotive contexts. While introducing additional uncertainties in regard to the overall system performance and safety, the adoption of COTS components constitutes a necessary means for effective product evolution and innovation. Accordingly, we propose in this paper a novel approach that aims to enable advanced operation monitoring and self-assessment in regard to operational uncertainties and thereby automated performance and safety awareness. The emphasis is on the integration of several modeling technologies, including the domain-specific modeling framework EAST-ADL, the A-G contract theory and Hidden Markov Model (HMM). In particular, we also present some initial concepts in regard to the usage performance and safety awareness for quality-of-service adaptation and dynamic risk mitigation.

Keywords: Cyber-Physical Systems, Commercial-Off-The-Shelf, Uncertainties, EAST-ADL, Contract, Probabilistic Inference, Safety, Performance.

1 Introduction

One key technological aspect of a modern automotive vehicle is its embedded E/E (Electrical/Electronic) system, consisting of sensor and actuator devices, ECUs (Electronic Control Units) and communication networks (e.g. CAN, LIN, MOST). By means of digital software and hardware, the embedded system allows both more ef-

fective realization of advanced functionalities and better achievement of performance and flexibilities in a way that is impossible with traditional mechanical and electrical solutions. Currently, the industry shares the view that the embedded system constitutes the most important technology for the advances in sustainability, road safety, and novel traffic solutions. The key innovation areas include ADAS (Advanced Driving Assistant Systems) and AD (Autonomous Driving) [1], with further enhancement by V2V (Vehicle-to-Vehicle) and V2I (Vehicle-to-Infrastructure) communication for cooperative behaviors [2]. Because of such features, an automotive vehicle becomes cyber-physical in nature by having, on one hand, physical dynamics and energy flows under control, and, on the other hand, the corresponding perception, control and cognitive loops within the embedded E/E system.

The evolution of automotive vehicles relies strongly on the technological developments and innovations in the electronics industry that are driven by consumer products. For automotive, the adaptation of generic purpose COTS (Commercial-Off-The-Shelf) solutions, ranging from camera, radar and other sensors for traffic perception, to speech recognition and augmented-reality displays for human-machine interactions, and to wireless and telecommunication services for V2V and V2I connectivity, has been advocated as a necessary means for shortening the innovation loops and enabling efficient product evolution. Unfortunately, there are often big gaps between the quality expectations of automotive vehicles and the quality assurance by generic purpose COTS. One reason for this is that the COTS solutions are developed for different use cases or operational conditions differing from the automotive ones. For example, the reliability targets of automotive components differ considerably from those of consumer products. The expected lifetime for electronics components would be up to 15 years for automotive vehicles, in comparison to a length in 2-5 years for consumer products [3]. Meanwhile, the operational temperature ranges are normally $-40\sim 160^{\circ}\text{C}$ for automotive vehicles and $0\sim 40^{\circ}\text{C}$ for consumer products. Moreover, as a failure of a COTS solution for automotive vehicles could result in system hazards, a systematic management of the risks according to the safety standard ISO26262 becomes necessary [4]. The information of concern typically includes not only a specification of its functional behaviors, but also a complete definition of its assumed failure modes, adopted quality assurance measures, and expected system-wide measures for fault tolerance and treatment. This is not a trivial task for separately developed COTS.

This paper presents a novel model-based approach to system operation monitoring and self-assessment in regard to the operational uncertainties and thereby automated performance and safety awareness. The approach emphasizes an integration of several modeling technologies, including the domain-specific modeling framework EAST-ADL for modularity design, the A-G contract theory for module specification, and Hidden Markov Model (HMM) for probabilistic inference of operation conditions. The ultimate goal is to support effective integration and management of separately developed services and components in safety critical CPS by means of quality management services for post-deployment operation- and life-cycle management. The rest of this chapter is structured into the following sections: Section 2 provides an overview of the overall methodology. Section 3 introduces the modeling support for modularity design and module specification for successful management of COTS solu-

tions, including the EAST-ADL modeling framework and the A-G contract theory. Section 4 focuses on the dynamic self-assessment through probabilistic inference based on Hidden Markov Model (HMM), while describing the related software services and design considerations for self-assessment and self-management. An overview of related technologies is given in Section 5. The paper concludes with Section 6 by elaborating the future research directions.

2 Overview of the methodology

The overall methodology is shown in Fig. 1. Following the *knowledge-in-the-loop* paradigm proposed in [5], the approach emphasizes an integration of the models that describe system requirements, design solutions, verification and validation cases, and the software services that are embedded for operation monitoring, *compositionality* and composability assessment, diagnostics and anomaly treatment.

Fig. 1. The overall methodology for a model-based contract and service for self-managed system service and components. ① Step I – Architecture design ; ② Step II – Partitioning and Assertion Specification; ③ Step III – Isolation and Contract Specification; ④ Step IV – Service-based Self-Management of System Services and Components. R- Requirement; C- Contract; OC-Operation Contract; SC-Safety Contract.

As shown in Fig. 1, the methodology consists of the following four main steps with the activities ranging from modularity design to automated post-deployment services:

- **Step I – System architecture design:** This step consists of work tasks that focus on the *overall design* of a particular cyber-physical system. The issues of concern include the expected operational environments, the functional and technical constituent units, the expected behaviors and quality constraints, the corresponding

verification and validation cases, etc. As the outcome, it provides a system view that stipulates the design-space with compositional variants and restrictions.

- **Step II – Partitioning and assertion specification:** This step consists of work tasks for system *modularity design*, which is centric on partitioning the target system into modules and thereby refining the system design-space for external COTS solutions. In effect, the design results in a restructuring of the target system so that some of its constituent units can be developed and managed independently. As a self-contained unit, each system module should not violate the functional and technical requirements of the corresponding services and components being composed. We specify such top-level module constraints as the *module assertions* (e.g., the accepted types and resolutions of input and output signals, the allowed range of execution time, and the criticalities of failure modes).
- **Step III – Isolation and Contract Specification:** This step consists of work tasks that provide the *interface design* for the preferred modules. By *isolation*, we refer to the process of defining and configuring the target system in regard to preferred module properties in accordance with the module assertions. This allows each module, as a self-contained system unit, to be isolated from changes or variations in the rest of target system and in the particular COTS providing the module implementation. The results of such module interface are specified as *contracts*. Each contract specifies formally some preferred module assertions. In particular, such assertions can also cover the operational behaviors of concern, including the modes for diagnostics and maintenance. Such contract extension is referred to as Operation Contract (OC) and Safety Contract (SC) in Fig. 1. A Safety Contract (SC) further extends the operation contract by declaring the states and transitions for fault tolerance and treatment, such as failure modes, safe states, and the related transitions for error handling and fault treatment.
- **Step IV – Service-based Self-Management of System Services and Components:** This step consists of the design and execution of *quality management services* for post-deployment monitoring, assessment and adaptation of system and components for quality-of-service control and safety management. These quality management services take the module contracts as formal specifications of the parameters to be monitored, the conditions of conformity to be judged, and the adaptations to be conducted. They estimate thereby the system operation conditions by probabilistic inference with Hidden Markov Model (HMM). The adaptation planning is centered on the condition-action reasoning based on some deterministic or probabilistic decision models. For the service realization, a key issue is related to how to enable the run-time monitoring and control through integration with system run-time environment and maintenance infrastructure.

3 Modeling Support for Modularity Design and Module Specification

3.1 EAST-ADL for Modularity Design

ADL (Architecture description language) is a modeling technology for structuring and

managing the engineering information of a system in particular in regard to the overall system design. As a domain-specific approach, the EAST-ADL represents a key European initiative towards a standardized description of automotive embedded E/E systems [6, 7]. In our approach, the EAST-ADL system model, together with the associated requirements and constraints, constitutes a basis for effective but still very flexible modularity design. That is, given the models capturing the system wide interdependencies, a particular modularization task can be driven at any specific abstraction level in accordance with the particular preferences of modules. For example, the modules given by features (*EAST-ADL Vehicle Level*) provide a structuring of the externally visible functionalities of the target system and allow a service-oriented composition of external functions. Meanwhile, module definitions can also be done based on software components (*EAST-ADL Implementation Level*) to allow a component-based engineering of software solutions. In both cases, the EAST-ADL system model constitutes a useful means for systematically reasoning about the corresponding module coupling and cohesion, both in regard to the functional interactions (which are given by communication links in the same level of abstraction) and technical implications (which are given by realization links across the abstraction levels).

3.2 A-G Contract for Module Specification

Following the contract theory defined in [8, 9], we use formal contracts to specify the expected functional and technical properties of a system module. Fundamentally, a system module M is defined by its variables, behaviors as well as some related quality constraints as follows

$$M = (\mathbb{V}, \mathbb{F}, \mathbb{Q}) \quad (1)$$

with $\mathbb{V}$ for all variables $\mathbb{V} = \mathcal{U} \cup \mathcal{X} \cup \mathcal{Y} \cup \mathcal{K}$, where $\mathcal{U}$ denotes the input variables, $\mathcal{X}$ the internal state variables, $\mathcal{Y}$ the output variables, and $\mathcal{K}$ the configuration variables; $\mathbb{F}$ for all functional behaviors over the variables $\mathbb{F} = \llbracket \mathcal{F}(\mathcal{U}, \mathcal{Y}, \mathcal{X}, \mathcal{K}) = 0 \rrbracket$; and $\mathbb{Q}$ for all quality constraints over the functional behaviors $\mathbb{Q} = \{\mathbb{P}, \mathbb{R}, \dots\} = \{\llbracket \mathcal{P}(\mathbb{F}) = 0 \rrbracket, \llbracket \mathcal{R}(\mathbb{F}) = 0 \rrbracket, \dots\}$ with $\mathbb{P}$ for all performance constraints and $\mathbb{R}$ for all reliability constraints, etc. For successful system integration, all COTS solutions implementing a module implementation need to satisfy the expected module properties given by M , under certain environmental conditions. Such a requirement is formally defined by module contract $\mathcal{C}$ in terms of the pair:

$$\mathcal{C} = (\varepsilon_c, \mathcal{M}_c) \quad (2)$$

with $\mathcal{M}_c$ for all COTS solutions satisfying the M ; and ε_c for all the legal environmental conditions. A contract is consistent if $\mathcal{M}_c \neq \emptyset$ and compatible if $\varepsilon_c \neq \emptyset$. A component $M_{\text{comp}} \models \mathcal{C}$ if and only if $M_{\text{comp}} \in \mathcal{M}_c$; An environment $E \models \mathcal{C}$ if and only if $E \in \varepsilon_c$.

An *A-G* (Assume-Guarantee) contract is a formalism for the description of contract $\mathcal{C}$ without directly referring to the actual COTS solutions $\mathcal{M}_c$. Normally, it is used for behavioral specifications of components, with A for the constraints on acceptable behaviors of the environment and G for the guarantees in terms of the corresponding component behaviors (as in e.g. [9]). Here, we use the A-G contract formalism in a

more generic sense for all constraints that are characterized by logical or technical causalities from some environmental conditions (A) to some consequential module properties (G). Formally, an A - G module contract $\mathcal{C}$ is a pair of assertions (A , G) over some module variables, defined as:

$$A \Rightarrow G \quad (3)$$

with A for the constraints on the acceptable environment conditions; and G for the corresponding constraints on the properties to be guaranteed by module implementations. There is a logical or technical causality between A and G . That is, a module M satisfies the contract $\mathcal{C}$ if it has the same variables as $\mathcal{C}$ and satisfies the guarantee G whenever subjected to the assumptions A , (i.e. $A \cap M \subseteq G$). Currently, we use STL (Signal temporal logic) [10] and PrSTL (Probabilistic Signal temporal logic) [20] for the constraint specification.

System architecture models constitute the basis for the contract design. For example, an automotive vehicle braking system has been introduced in [7] as a case study for the modeling support. In the system, there is an ABS (Anti-lock Braking System) controller and an electromagnetic brake actuator at each wheel of the vehicle. For the braking control, a global brake controller receives the driver braking request and then sends brake force request to each ABS controller. Fig. 2 shows an excerpt of the system architecture model (*SystemModel*).

Fig. 2. The architecture model of an automotive vehicle braking system and the timing specification in EAST-ADL (implemented with the DSM Workbench MetaEdit+)

A timing model augments the system model with information about the executional events and their timing constraints. The declarations are also shown in Fig. 2. An executional event defines the occurrence of data arrival on a port (e.g. $pStimuli_PedalPosIn$), or the triggering of function execution (e.g. E_pI). An event chain binds then together events to establish synchronization relations between the events e.g. to capture a complete end-to-end flow timing requirement from sensor inputs to actuator outputs. For example, the event chains *End-to-End Braking Response_FL* and *End-to-End Braking Response_FR* capture the timing requirement from driver braking request ($pStimuli_PedalPosIn$) to the brake torque actuation on the FL and FR wheels ($pResponse_ForceActuation_FL$, $pResponse_ForceActuation_FR$) respectively. The timing model also captures the timing constraints within the end-to-end flow, such as the preferred delay (A) from the braking request ($pStimuli_PedalPosIn$) to the arrival of torque request on the receiving port ($driverRequestedTorque_gbc_in$).

This EAST-ADL specification of timing properties is converted to *A-G* contracts for the integration of modules implemented by separately developed services or components. Suppose the entire braking system will be based on COTS, the *A-G* contract C_{com_1} shown in Fig. 3 stipulates the end to end communication and timing in a service-oriented way. The contract first declares the related executional events as contract variables: x for $pStimuli_PedalPosIn$, y_1 for $pResponse_ForceActuation_FL$, y_2 for $pResponse_ForceActuation_FR$. All these events are of the type *control* and *latency critical*. Given the assumption x about its *size*, *periodicity* and *priority*, each separately developed solution for the braking system needs not only to guarantee the preferred size and periodicity y_1 and y_2 , but also to satisfy the delays $x \rightarrow F_{[0..#delay_{y_1}]}y_1$ and $x \rightarrow F_{[0..#delay_{y_2}]}y_2$ for the *End-to-End Braking Response_FL* and *End-to-End Braking Response_FR* respectively.

$$C_{com_1}: \left\{ \begin{array}{l} \text{messages: } \begin{cases} \text{in: } x \\ \text{out: } y_1, y_2 \end{cases} \\ \text{message_types: } x, y_1, y_2 \in (\text{Control}, \text{Latency_Critical}) \\ \text{assumptions: } x = (\#size_x, \#periodicity_x, \#priority_x) \\ \text{guarantees: } y_1 = (\#size_{y_1}, \#periodicity_{y_1}), y_2 = (\#size_{y_2}, \#periodicity_{y_2}), \\ \quad G(x \rightarrow F_{[0..#delay_{y_1}]}y_1), G(x \rightarrow F_{[0..#delay_{y_2}]}y_2), \dots \end{array} \right.$$

Fig. 3. The *A-G* contract for the operation of vehicle braking. Here, we use STL (Signal temporal logic) [10] to denote the behavioral constraints, with the operator G for always (globally) and F for eventually (in the future).

4 Embedded Dynamic Self-Assessment Service and Models

4.1 The Embedded Software Services

Dynamic self-assessment refers to the capability of a system to autonomously perceive its own operational situations (i.e. situation-aware) and thereby estimate the satisfactions of related requirements. It constitutes the basis for self-management, referring to the capability of a system to autonomously choose or alter its own config-

urations, behaviors, or external interactions for the reasons of optimization, robustness and safety [11]. As a support for self-assessment and self-management, two additional software services, shown in Fig. 4, have been proposed in our approach. These software services allow self-managed error detection and fault treatment, while complementing the verification and validation effort at development time with the post-deployment data and analysis. These two services are: **1. Monitoring and Assessment Service (MAS)** – the embedded software service for self-assessment by monitoring and defining the operational conditions of component and system; **2. System Adaptation Service (SAS)** – the embedded software service for self-management by planning and coordinating necessary system adaptations.

Fig. 4. The architecture of software services for self-management system services and components. (Note, ①②③④ are for the methodological steps as depicted in Fig 1.)

Contracts are used for the decision-making in the MAS. For example, for a critical component, this embedded service monitors the actual external conditions (A) and tracks the internal error states (X_{Err}); it then triggers appropriate error handling behaviors ($ErrHandling$) by the adaptation service SAS when necessary. This is given by the expression: $A \Rightarrow G \wedge ((Pr(\neg A) \geq l') \vee (Pr(X_{Err}) \geq l'')) \Rightarrow ErrHandling$, in Fig. 4, where two probability assertions with the operator Pr specify the corresponding probabilistic fault assessments in regard to two acceptance levels (l' and l''). The decision-making of the SAS requires built-in knowledge about the configuration variability and the related rules for deciding and planning changes (e.g., mode switches). For system dependability, the ability of inferring the causing factors of anomalies or other undesired conditions plays a key role for the success.

4.2 Contracts and Models for Operation Assessment

The embedded service MAS tracks the component operational behaviors based on some observed evidences in terms of sensor measurements and system service feed-

backs. Such behaviors evolve dynamically over time, while exhibiting stochasticity. The stochasticity could be due to varying workloads, emerging failures, as well as the actions of quality of service adaptation, error handling and fault treatment. Accordingly, the reasoning of such behaviors is supported by Markov Model (MM), which is a special type of DBN (Dynamic Bayesian Network) with a chain structured states in time series [12]. We assume here that the target system or component satisfies Markov Property (i.e. its future state is only affected by its present state).

In particular, we use the Operation Contract (*OC*) and Safety Contract (*SC*), (shown in Fig. 1) for capturing the operation concerns of module. The key content is related to the constraints of nominal and error states as well as the conditional probabilities of the transitions among these states. For example, assume that one component *BrakePedalSensor* operates with a simple normal mode and a simple error mode, shown in Fig. 5. We denote the operational states with X given by two state variables: $X = \{Norm, Err\}$, where *Norm* – Normal Mode, and *Err* – Timing Error. The overall operational behavior is characterized by the Markov Model as a set of chain structured states in discrete time series: $(X_1, X_2, \dots, X_{k-1}, X_k, X_{k+1}, \dots)$. At a time instance k , the conditional probabilities of state transition outcomes are given by the values of $p(X_k | X_{k-1})$. For instance, the conditional probability of state transition from *Norm* at X_1 to *Err* at X_2 equals to 0.01 , which corresponds to the component reliability requirement; while the conditional probability of state transition from *Err* at X_2 to *Norm* at X_3 equals to 0.03 , which corresponds to the requirement on component error recovery. With such constraints, each particular execution exhibits a sequence of state conditions in time series, such as $(X_1 = Norm, X_2 = Norm, X_3 = Err, X_4 = Norm, \dots)$. For a component in general, the probability figures can be estimated from related experience or laboratory testing in regard to particular operational conditions (e.g. failure rates in different temperature levels and repair rates by related component services). Fig. 6. shows the corresponding specification with two probabilistic temporal assertions as the fault contract extension: $(\tau) = 0.99, \Pr(\text{Next}(\neg\tau) = \tau) = 0.03$.

Given the contracts for operation and safety constraints, the service MAS (Section 4.1) monitors the system operation and thereby estimates the component status for situation awareness, performance and dependability management. The estimation is supported by Hidden Markov Model (HMM) [12], which consists essentially of two pieces of Markov Model: one for the conditional probability of operational states $p(X_k | X_{k-1})$, and another one for the conditional probability of external observation given a system state $p(Y_k | X_k)$. Here, Y_k refers to the variables of observation given by the target system architecture and component interfaces, while X_k and X_{k-1} refer to the operational state at time k and $k-1$ respectively. As the conditional probability $p(Y_k | X_k)$ plays a key role in the status estimation, it is preferred that such information is available in the contracts (i.e. *OC* and *SC*) as operation observability or diagnostics constraints. Otherwise, additional efforts for the elicitation of such constraint through analysis, testing and machine-learning would be necessary. The definition of state variable X of a module depends on the needs of situation awareness and quality management. The state variable could include non-observable analytical states, for which the HMM models support the inference. For example, due to some technical or managerial constraints, the embedded service MAS for a distributed system would only be

deployed on the network switches. The indicators Y_k then provide information about the timing and resource utilization at the switches at time k . By such info, the service MAS would estimates the end-to-end communication deadlines and related timing failures for some messages based on the knowledge of $p(Y_k|X_k)$ and $p(X_k|X_{k-1})$. One experimental result is shown in Fig. 7. See e.g. [12] for the further details about the algorithms for hidden state inference.

Fig. 5. The component *BrakePedalSensor* and its operational behavior in terms of Markov Model given by a chain-structured discrete-time states: $(X_1, X_2, X_3, \dots)$. The table for conditional probabilities $p(X_k|X_{k-1})$ defines the probabilities of state transitions. The Markov model representation is extended with the indicators $Y_1, Y_2, Y_3, \dots$ to form a Hidden Markov Model.

$$C_{\text{comp_1}}: \begin{cases} \text{messages: } \begin{cases} \text{in: } v_{in} \\ \text{out: } s_{out} \end{cases} \\ \text{message_types: } v_{in}, s_{out} \in (\text{Control}, \text{Latency_Critical}) \\ \text{assumptions: } v_{in} = (\#size_{v_{in}}, \#periodicity_{v_{in}}, \#priority_{v_{in}}) \\ \text{guarantees: } s_{out} = (\#size_{s_{out}}, \#periodicity_{s_{out}}), \quad \tau = G(v_{in} \rightarrow F_{[0..\#delay_{s_{out}}]} s_{out}) \\ \mathbf{Pr}(\tau) = 0.99, \mathbf{Pr}(\text{Next}(\neg\tau) = \tau) = 0.03 \end{cases}$$

Fig. 6. The Operation and Fault Contract for a brake pedal sensor with additional probabilistic assertions. Here, V_{in} and S_{out} denote the input and output events respectively. We use *PrSTL* (Probabilistic Signal Temporal Logic) [20] to denote the probabilistic behavioral constraints, with the operators **Pr** for the probability assertion, **Next** for next state, and $\neg$ for negation.

Fig. 7. Estimated probability of state *Norm* of $p(X_k|X_{k-1})$ according to a specific sequence of emissions Y_k .

4.3 Contracts and Models for System Performance and Safety Awareness

One key aspect of performance and safety awareness is related to the estimation of component operational conditions and then their system-wide impacts. Such a task is supported in our approach by the service MAS. By observing the system operation and inferring local conditions of components, the risks of violating a component contract is assessed according to the actual probability of being in particular error states. Moreover, by composing the estimated local operational conditions, the service also

estimate the corresponding system-level operational conditions and thereby assess the risks of violating system-level performance and safety constraints.

Given a distributed E/E system with switched Ethernet for the communication of messages in mixed criticalities, the system performance levels can be analyzed and derived using Stochastic Network Calculus [13, 14]. The corresponding timing constraints in regard to the arrival patterns of messages, resource utilizations and end-to-end performances can be captured by contracts. In operation, the MAS observes the arrival processes of messages to a network switch and thereby estimates the overall performance. The service detects certain combinations of performance levels of messages that may cause exceptionally high interference to critical messages.

For the safety awareness, fault models capturing the overall system failure logic in terms of FTA and FMEA [15,16] play a key role. They allow effective safety reasoning by reusing the results of regular safety analysis in regard to the system-wide consequences of local errors. Current EAST-ADL technology allows a multi-viewed approach to safety engineering [17], where error models capturing the plausible anomalies in terms of local faults and failures and their permeations across the system are synthesized automatically from nominal architecture models. The analysis engine for FTA and FMEA is given by the HiP-HOPS tool [18, 19]. For example, pre-calculated minimum cut-sets can be used to quickly determine the system-wide effects of particular component errors. An alternative way is to learn and create a Dynamic Bayesian Network (DBN) for capturing the system-wide effects of local errors. In regards to functional safety, the estimated probability of system failure represents the uncertainty that the target system has in regard to the related functionality. Such a functionality, when subjected to safety requirements, has associated safety integrity levels (i.e. ASIL), representing essentially the degree of (in)tolerance in regard to the failures. For example, given some safety requirements: 1. *SafetyGoal#1: ASIL A, the ... control function shall not cause a clash ... when the vehicle has speed in 15 km/h;* 2. *SafetyGoal#2: ASIL D, the ... control function shall not cause a clash ... when the vehicle has speed in 45 km/h.* The requirement *SafetyGoal#1* with ASIL A has better tolerance to uncertainties by system failures. For a system, the estimated failure probabilities by the embedded service MAS provides a dynamic justification of the overall system uncertainty in regard to its control actions. Such an estimated awareness in regard to safety makes it possible to avoid control actions with unacceptable high risk.

5 Related Technologies

In recent years, generic and domain specific modeling frameworks have been developed for model-based development (MBD) of cyber-physical systems in many industrial domains, such as SysML [21], AADL [22] and EAST-ADL [6, 7]. All these technologies focus on the support for system description and information management only during the development time. For dealing with the partially unknown or inaccurate system properties, the specification of systems with abstract goals, probabilistic properties and contracts together with the provision of related intelligent services and platform support will be necessary. For example, a mission goal description

based approach to the identification of required capabilities for the constituent system, operations, connections, emergent behavior, among other elements of system-of-systems has been proposed in [23]. A contractual description of constituent systems interfaces, to address the imprecision and uncertainty inherent in the description of constituent systems is given in [24]. For quality assurance and certification, such contractual support needs to be defined and managed seamlessly along with the lifecycles of system development, componentization and maintenance. A great many techniques have been developed and explored for advanced verification and validation of complex systems, both statically at design-time and dynamically at run-time. One particularly promising type of model checking is the so-called on-line model checking, as explored in [25]. In this approach, the resulting combined state space is continuously monitored against critical safety properties, or used to compute safe trajectories. Another approach that combines model-checking with dynamic model validation and creation based on machine-learning is learning-based testing (LBT) [26]. New methodological approaches to operational risk assessment include statistical analysis of near-miss and incident data using related Bayesian theories to estimate operational risk value and the dynamic probabilities of accidents sequences having different severity levels [27]. In regard to all these above mentioned approaches, our work aims to provide a framework for facilitating the industrial adoptions.

6 Conclusion

This paper presents a novel approach that aims to enable advanced operation monitoring and self-assessment and thereby automated performance and safety awareness. The ultimate goal is to support the adoption of generic COTS solutions in cyber-physical systems without violating the overall system requirements. To these ends, the approach emphasizes an integration of the domain-specific modeling framework EAST-ADL, the A-G contract theory, and Hidden Markov Model (HMM). With EAST-ADL for the overall system modeling, the provision of appropriate contract formalism allows dynamic self-assessment and adaptations with embedded software services. The approach also addresses the stochasticity of system and component operation. With HMM, the software services estimates the operational status, including non-observable analytical states, for the situation awareness, performance and dependability management. Currently, we are looking into how to effectively integrate the modeling technologies.

References

1. SAE International, SAE Information Report: (J3016) Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems.
2. European Commission: Intelligent transport systems. [Online]. Available < https://ec.europa.eu/transport/themes/its_en >
3. PwC Semiconductor Report: Spotlight on Automotive. PwC. September 2013.
4. ISO, ISO 26262 Road vehicles – Functional safety.

5. D. Chen, et.al.: A Knowledge-in-the-Loop Approach to Integrated Safety&Security for Cooperative System-of-Systems. IEEE 7th Intl. Conf. on Intelligent Computing and Information Systems, ICICIS'15, Cairo, Egypt. December 12-14 , 2015.
6. EAST-ADL. EAST-ADL Domain Model Specification, Version M.2.1.12. 2014.
7. R. Kolagari, et.al.: Model-Based Analysis and Engineering of Automotive Architectures with EAST-ADL: Revisited. Intl. Journal of Conceptual Structures and Smart Applications (IJCSSA). Vol 3, Iss 2, Jul - Dec 2015. IGI Global Publishing, Hershey, USA.
8. A. Benveniste, et.al.: Multiple Viewpoint Contract-Based Specification and Design. Formal Methods for Components and Objects: 6th Intl Symposium, FMCO 2007.
9. A. Benveniste, et.al.: Contracts for system design. Report RR-8147, Inria, Nov 2012.
10. O. Maler, et.al.: Monitoring temporal properties of continuous signals. Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, Joint Intl. Conf. on FORMATS/FTRTFT, 2004.
11. R. Anthony, et.al.: Context-Aware Adaptation in DySCAS. Electronic Communications of the EASST, Vol 19: Context-Aware Adaptation Mechanism for Pervasive and Ubiquitous Services (CAMPUS). European Association of Software Science and Technology (EASST), 2009. ISSN 1863-2122.
12. Z. Ghahramani: An introduction to hidden Markov models and Bayesian networks. Hidden Markov models: applications in computer vision, World Scientific Publishing Co., Inc., River Edge, NJ, 2001.
13. Y. Liu, et.al.: A calculus for stochastic QoS analysis. Performance Evaluation, vol. 64, no. 6, pp. 547 – 572, 2007.
14. Y. Jiang and Y. Liu: Stochastic Network Calculus. Springer Publishing Company, 2008.
15. W.E. Vesely: Fault Tree Handbook, US Nuclear Regulatory Committee Report NUREG-0492, US NRC, Washington, DC. 1981.
16. P. Palady: Failure Modes and Effects Analysis. PT Publications, West Palm Beach. 1995 ISBN: 0-94545-617-4.
17. D. Chen, et. al.: Integrated Safety and Architecture Modeling for Automotive Embedded Systems. e&i elektrotechnik und informationstechnik, Vol 128, No.6, Automotive Embedded Systems. Springer Wien, 2011. ISSN0932-383X. DOI 10.1007/s00502-011-0007-7.
18. D. Chen, et. al.: Systems Modeling with EAST-ADL for Fault Tree Analysis through HiP-HOPS. 4th IFAC Workshop on Dependable Control of Discrete Systems. York, U.K. Sept 4th- 6th, 2013.
19. Y. Papadopoulos, J.A. McDermid. Hierarchically performed hazard origin and propagation studies. LNCS, vol.1698, pp.139-152. Springer. 1999.
20. D. Sadigh, A. Kapoor: Safe Control under Uncertainty with Probabilistic Signal Temporal Logic. Robotics: Science and Systems (RSS), June 2016.
21. SysML. OMG Systems Modeling Language (OMG SysML™), OMG.
22. P. H. Feiler, D. P. Gluch: Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language. Addison-Wesley Professional. SEI Series in Software Engineering series. 2012. ISBN-10: 0-321-88894-4
23. E. Silva, et.al.: A Mission-Oriented Approach for Designing System-of-Systems. Proc. of the 10th System-of-Systems Engineering Conference (SoSE), May 2015, pp. 346-351.
24. J. Bryans, et.al.: SysML Contracts for Systems of Systems. June 2014, IEEE Systems of Systems Engineering Conference 2014.
25. M. Althoff, et. al.: Online Verification of Automated Road Vehicles Using Reachability Analysis. IEEE Trans. on Robotics 30(4), 2014.

26. K. Meinke, et. al.: Incremental Learning-Based testing for Reactive Systems. Proc. Int. Conf. on Tests and Proofs TAP 2011, LNCS 6706, Springer Verlag, 2011. IEEE Transactions on Robotics 30(4): 903-918. 2014.
27. A. Meel: Plant-specific dynamic failure assessment using Bayesian theory. Chemical Engineering Science 61, pp. 7036–7056. 2006.