2016 IEEE Intelligent Vehicles Symposium (IV)
Gothenburg, Sweden, June 19-22, 2016

Model based scenario specification for development and test of automated driving functions

Johannes Bach[1], Stefan Otten[1] and Eric Sax[1]

***Abstract*— Research and evaluation of algorithms and system architectures for automated driving advanced to a stage where transition from prototyping to series development seems practicable in some extent. While particular systems for environmental perception play a key role in advanced driving assistance systems, we still lack feasible methods for specification and validation of complex driving scenarios. This leads to increased effort in testing and inconsistent requirement definition along different development phases. In this paper we propose a methodology for abstract positional and temporal description of driving scenarios. The approach utilizes a movie related and omniscient view composed of sequential acts. Each act combines both states and interactions of distinct participants as well as the rudimental scenery. Selective events trigger changes in conduct leading to transitions between acts. Graphical visualization provides simple presentation of complex scenarios. Rule sets provide consistency checks and support semi-automated generation of test cases. The presented methodology facilitates model based test specification and requirements design constituting a consistent characterization of system environment from early concept and development to validation.**

I. Introduction

Within the last decade many innovations regarding the automotive domain were based on Advanced Driver Assistant Systems (ADAS). As technology rapidly advances towards automated driving this trend further continues, integrating more and more intelligent, interconnected and complex functionality. Thereby challenging established methods, processes and tools for system and software development to cope with rising demands. Especially specification, verification and safeguarding of ADAS require innovative solutions particularly to satisfy regulatory and legal demands. Requirements and test goals for such systems are often determined in text based documents such as Software Requirements Specifications [1] or Test Plans [2]. In addition, exemplary Unified Modeling Language (UML) [3] or Systems Modeling Language (SysML) [4] utilize model based methodologies in form of diagrams and charts for software and system specification.

This leads to a multitude of diverse methods, activities and tools applied by various development and test personnel in order to perform inevitable verification and validation during product development [5]. Ranging from basic unit testing over X-in-the-loop (XiL) simulation to test runs on dedicated compounds, all activities base on the initial system requirement definition and are tracked in lists or catalogs. Recently introduced international standard ISO 26262 for functional safety of road vehicles describes activities and work products regarding the entire development process. Part 3: Concept phase [6] defines the standard's nucleus as "the first objective is to define and describe the item, its dependencies on, and interaction with, the environment and other items." The item definition determines the context to which ISO 26262 is applied and composes the foundation of all other activities. It requires an explicit and uniform depiction throughout product life-cycle, whereas ISO 26262 explicitly states the importance of the environment and interaction with other items. Expanding system boundaries and increased interaction of ADAS complicate this essential task by introducing further dependencies and correlations. While the environment of Anti-lock Breaking System (ABS) is confined to the vehicle, Adaptive Cruise Control (ACC) and Lane Keep Assist (LKA) utilize single dedicated sensors such as RADAR or video based lane detection to scan the vehicles surroundings. Upcoming semi-automated functions, such as stop-and-go assist or predictive ACC [7], often require a combination of sensors for environmental perception. By applying formal descriptions subsequent machine comprehension of the recorded surrounding is enabled.

[1] Johannes Bach, Stefan Otten and Eric Sax are with FZI Research Center for Information Technology, 76131 Karlsruhe, Germany bach@fzi.de, otten@fzi.de, sax@fzi.de

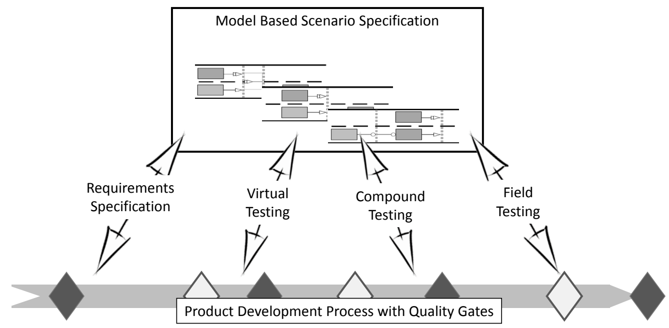

Fig. 1. Usage of abstract driving scenario modeling throughout the automotive development process to support requirements specification and test design.

As stated above, design and test of ADAS often resorts to XiL simulation. Consequently there is a wide range of simulative approaches and tools. Reaching from macroscopic traffic simulation with a global perspective [8] to microscopic driving simulation [9], augmented reality applications [10] with egocentric perspectives or hybrid simulation and data-driven approaches [11]. Between different tools specification of test scenarios varies broadly in method and detail. Thus hampering exchange, reutilization and comprehension.

In 1997 UML introduced "a language that produces

978-1-5090-1821-5/16/$31.00 ©2016 IEEE

drawings comparable to the blueprints long used in other technical disciplines" [12]. The main benefit therein consists in graphical abstraction of highly complex software products, providing facile comprehension, and simplifying links in between requirements, designs and test cases by means of consistent language. Today we lack similar improved methodologies for consistent description and specification of driving scenarios in context of ADAS interaction. Including representation of static scenery elements as well as abstract definition of complex behaviors and interactions of and in between different road users. In addition to better understanding, visual abstraction of scenario description supports the tedious process of item definition, requirements specification and test design throughout the automotive development process as depicted in Figure 1. Using a comprehensive and coherent depiction facilitates reusability and interdisciplinary exchange between different roles and responsibilities within enterprises as well as with their customers and partners. Further a standardized approach with a common language fosters efficiency and diminishes misunderstandings in test design. By providing different perspectives the analysis of driving scenarios, test recordings as well as the validation of test specifications is supported.

Abstract modeling of driving scenarios requires characterization of temporal and spatial causalities on a logical level. To create a comprehensive and simple apprehension of complex driving scenarios we propose a methodology of graphical abstraction referred to geometric and chronological as well as behavioral and interactional characteristics. This paper extends previous work on modeling concepts [13] [14] with a logical abstraction layer and presents a domain model specification based on a machine-readable language. In Section II we discuss our fundamental concept, utilized terminology, the abstract modeling approach and a graph based rule set. Section III describes the developed domain model, while Section IV presents our prototype implementation. The contribution concludes with Section V providing a prospect on future enhancements.

II. Driving Scenario Abstraction

Modeling of driving scenarios is discussed in a variety of publications. For example Richard et al. [15] utilize a combination of time lines, geometric diagrams and textual tables in driver task analysis. Their approach is focused on a ego vehicle based perspective with sequential task and event instructions, not including interactions between other road users or different perspectives. Interpretation of surrounding vehicles is based on scenarios by Guo et. al. [16]. Scenario specification for simulation environments often utilizes very practical or even physical characteristics (cf. [17], [9]), useful for direct translation into events and actions, but difficult to comprehend or adapt. Geyer et al. [13] propose an ontology concept for test and use-case catalog generation based on a theater analogy. This approach is further detailed by Ulbrich et al. [14] to foster consistency of terms and definitions. Another ontological approach introduced by Xiong [18] utilizes a knowledge base "for orchestrating scenarios with

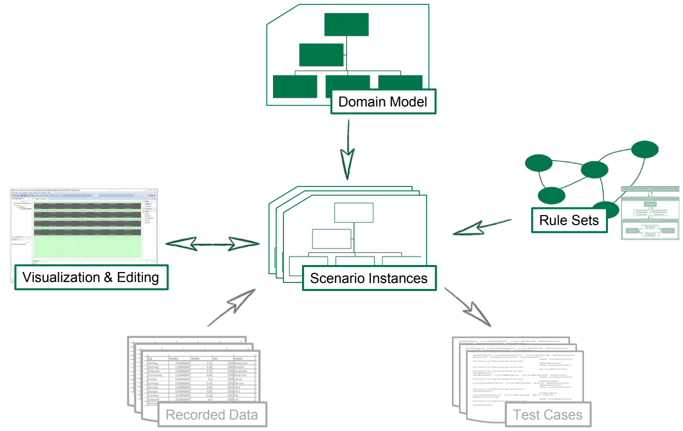

Fig. 2. Fundamental concept of the presented driving scenario design approach based on a domain model and graph based rule sets.

autonomous vehicles intelligently". Unfortunately none of these approaches allows abstraction of the described scenarios to a logical level. Thus hindering seamless transition between different development phases and roles, as well as reuse for varying systems.

According to Evans [19] in domain-driven design "the model is distilled knowledge", "the model is the backbone of a language used by all team members" and "the model dictates [...] the heart of the software". These three uses in mind, the presented concept aims on achieving a representation that is comprehensible for humans as well as machine-readable. Besides design and specification of consistent driving scenarios, analysis of conducted field tests and automated generation and mutation of test cases should be supported by the domain model and associated applications. A core challenge designing the model is the abstract representation of temporal and geographical information without sacrificing causal relationship. In a first step, we focus on multi-lane environments, especially highway scenarios.

The fundamental application concept of our model based approach is depicted in Figure 2. Based on the domain model representation for driving scenarios instances can be specified. Rule sets for various aspects preserve validity of scenarios and support refinement. The fundamental concept includes the intention to provide automated parsing of scenario instances from recorded real world test data. Application of physical parametrization allows derivation of test cases for virtual validation methods from a scenario instance. Integration of reasoners and solvers provides opportunities to partially automate the parametrization task. A graphical representation offers a comprehensible format for analysis and refinement by development and test personnel.

A. Concepts and Terminology

The entities and actions relevant to comprehensive understanding of driving scenarios prompt the usage of theatre and movie related terminology, as already suggested by previous work [13] and [14].

The term scenario is utilized in versatile domains and applications as described in [13] and [14]. In the described domains, the main purpose of utilizing the term scenario is abstraction of highly complex context. The authors agree with the definition stating "a scenario spans a certain amount of time" [14]. According to [20], a scenario can be defined as "a postulated sequence or development of events". Referring to the movie analogy, a scenario encompasses all relevant content to carry out a specified storyline.

The principle of the presented approach is an abstraction of all relevant entities within a driving scenario. As suggested in Geyer et al. [13] all geo-spatial static entities are subsumed as the scenery of the driving scenario. This includes, but is not limited to, geometric features such as the road network with its nodes and links, as well as global attributes such as weather and time of the day. A sound approach to scenery modeling is specified by the OpenDRIVE format [21]. It provides a consistent and detailed base for scenery description for this approach. Focusing on a generalized specification of driving scenarios, only abstract features such as roads, junctions and lanes will be used. Spatial information is reduced to logical relations between the entities. The omitted geometric details will be complemented during parametrization.

The term participant is utilized to describe all dynamic elements within a scenario with the capability for interactions with other participants. For example humans or human operated vehicles are categorized as participants. This categorization applies to the mere capability and does not imply movement, behavior or interaction in a scenario. In contrast to previous work there is no distinction between participants regarding roles, such as ego vehicle [13] and actor or observer [14]. This supports the concept of an omniscient perspective towards the scenario.

Geyer et al. [13] introduce the terms scene and situation. These terms are further substantiated by Ulbrich et al. [14]. While both works agree upon the definition of a situation as a "set of criteria that need to be true to conduct the associated action" the understanding of the term scene is at variance. In this context, the definition "a scene describes a snapshot of the environment including the scenery and dynamic elements, as well as all actors and observers self-representations, and the relationships among those entities. Only a scene representation in a simulated world can be all-encompassing (objective scene, ground truth)" [14] is applied. Thus, a situation is described as the subjective subset of relevant parts of a scene for a certain participant, enhanced by its intentions and function-specific aspects.

To specify the behavior, actions and interactions of and in between participants on an abstract level, a common concept is the usage of maneuvers. A list of 17 principal maneuvers is presented by Nagel et al. [22]. Referring to these, the presented approach defines maneuvers as an abstract definition of actions. This allows us to model a chain of subsequent maneuvers for each participant within a scenario. Further, maneuvers can be described as states in a graph representation. With edges describing generally feasible transitions between maneuvers and situations representing transition conditions. A detailed elaboration of this concept is presented in Section II-C.

Utilizing the definition of a scene as a snapshot in time, a further entity for the abstraction of time is necessary. Picking up the idea of movie strips and screenplays, in our context each scenario consists of different acts. The sequential order of acts specifies the complete movie, in our case the scenario. Each act contains an explicit and self-contained play of the actors inside. For our approach this results in participants, each performing a certain maneuver. An act lasts a undefined, but finite time span and encapsulates basic and similar situations. Subsequent acts vary in exactly one participant's maneuver to ensure a consistent storyline. As defined above, transitions between maneuvers conditionally depend on situations. Within a scenario, a specified event actuates an available transition. As a result the transition between acts is also triggered by events.

As already stated, a situation contains a subjective subset of a scene. The selector of this subset is implemented by means of a geometric abstract representation of the participants' environmental perception. To enable differentiation between varying levels of perception, we define a concept of near, mid, and far field perception layers. In addition, this layers also express the participants areas of interaction. The precise geometric extent of the perception and interaction layers may vary according to several reasons, such as utilized sensors, road topology or human fitness.

As the sequential behavior of each participant is already defined by a sequence of maneuvers, the exact geometric extend of the perception and interaction layers will affect the temporal aspects only. The causal sequence of the scenario is not influenced. This facilitates enrichment of geometric detail within the parametrization phase. Thereby, spatial abstraction during modeling of the scenario is maintained.

B. Modeling approach - spatial and temporal abstraction

Development of the presented modeling approach was driven by several conceptual drafts based on possible driving scenarios. Figure 3 outlines the driving scenario abstraction concept based on the movie analogy.

The example depicts a very common scenario used for validation of ADAS such as distance warning or ACC. The approach-and-follow scenario consists of two participants moving in the same direction and on the same lane. The following participant initially has a higher velocity. Within the presented modeling approach, this scenario constitutes three subsequent acts. Participant P1 undergoes the maneuvers free-cruise, approach and following. Participant P2 demonstrates the approached vehicle and performs a free-cruise maneuver with constant speed during all three acts.

The maneuver free-cruise describes movement along the participants lane at a desired maximum velocity, which is adapted according to road topology, regulatory elements and driving style. The maneuver approach is characterized by an existent velocity difference, which is reduced as the distance between two participants is closed to a defined time gap. Subsequently the maneuver following delineates adaption to

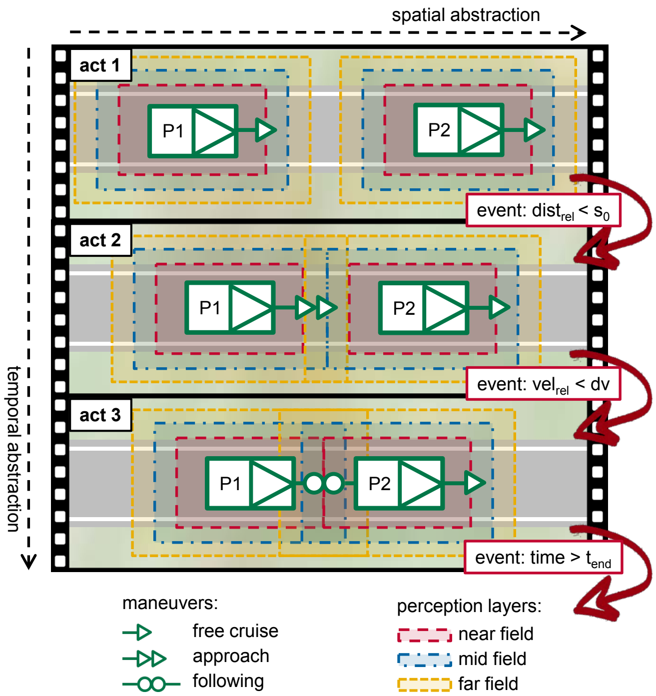

Fig. 3. Conceptual draft of the movie analogy driving scenario abstraction with a single lane approach-and-follow scenario as might be used for validation of ADAS such as distance warning or ACC.

the velocity trajectory of the preceding vehicle. The adaption is best described by a control process with desired time gap as reference input and velocity as the controller output.

The end of act 1 is determined by the maneuver transition of P1 from free-cruise to approach. The transition is triggered by the event $dist_{rel} < s_0$. The entity event is utilized to determine which maneuver transition is selected and consequently triggers transition from act 1 to act 2. Exactly one event is assigned to each act in order to preserve an unambiguous line of events.

The participant's perception and interaction layers are visualized by three square shapes representing the near, mid and far field. Entry of P2 into P1's mid field perception layer validates the condition for the maneuver transition from free-cruise to approach. Besides the spatial relation of participants, maneuver conditions can refer to the type of maneuver a participant within range performs, as well as to dedicated scenery elements. For example accrual of an additional lane establishes the possibility for a lane change maneuver, whereas on removal of the same the condition for free-cruise drops out. In addition, unlocking of a condition does merely enable a transition into another maneuver but not trigger it.

An important peculiarity of the chosen representation is the depiction of a satisfied condition. Its depiction is placed in the act in which the according maneuver is performed, as it depicts a necessity for the corresponding act, but not an impediment for the preceding act. This mode of modeling conserves temporal and spatial abstraction and enables a fuzzy changeover between acts. The exact time of transition is based on the related event and determined consecutively to modeling during parametrization of the abstract scenario instance.

In contrast to Geyer et al.'s [13] description "Depending on the driver's behavior, the driven route can deviate from the planned route" our approach determines exactly one course of events and does not allow deviations from the specified causal chain. This is an important premise for the intended usage of the modeling approach for requirement and test specification. According to the international standard for requirements engineering [1], requirements need to be "unambiguous and verifiable". Part four of the international standard for software testing [23] defines scenario design as a specification-based test design technique. Derivation of test conditions is specified such, that scenario testing "shall include identification of the 'main' scenario [...] and 'alternative' scenarios [...]". Therefore, deviations from the specified course of events by the System under Test (SuT) within the limits of specified requirements, imply the specification of additional alternative scenarios.

For basic graphical representation, different types of symbols representing maneuvers that are attached to the participant. Heading of symbols specifies the maneuver's effective direction. Within this conceptual draft, three maneuvers are used and visualized. The first one is a usual arrow, which represents free-cruise in a specific direction. The second one is a double arrowhead, indicating a higher speed than a referenced participant or layer. Depending on the participant's lane positions at hand, this depicts the maneuver approach for identical lanes or overtake for different. The third one is a circle. This describes a link between participants or layers to specify follow maneuvers between two or more entities. Each event is depicted within the act it terminates. Basic information about the type and reference of the event is supplied. The event of the last act also terminates the scenario.

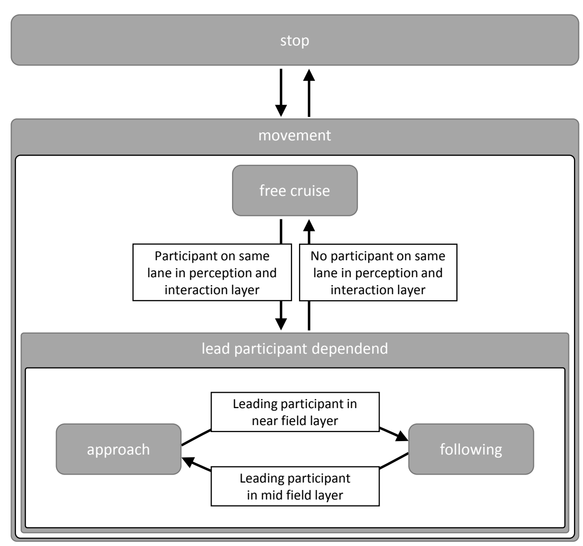

Fig. 4. Graph including the three maneuvers free-cruise, apporach and follow utilized in the approach-and-follow scenario above supplemented with the primitive maneuver stop.

There is no special graphical representation intended for the last event.

C. *Rules and consistency*

Consistency of specified scenario instances is ensured by graph based rule models. This rule models associate with the entities of the domain model. Figure 4 depicts an exemplary rule model utilized to describe possible transitions between maneuvers and their according conditions. Though there is a distinct similarity to statecharts, the rule model graph differs from common state charts. In contrast to state charts, the additional information attached to the transitions represents a restricting condition and not the triggering event. The triggering event is defined within the scenario instance elaboration. Therefore, our rule model represents a subset of possible maneuver transitions within the entire maneuver space.

In the depicted rule model for maneuver transitions, the primitive maneuver stop is supplemented. Stop does not only represent the participant's state of no movement, but includes the explicit intention not to move. While a situation of no movement could also occur during the maneuver following, for example in a stop-and-go scenario, the maneuver stop would be used to express, for example, a pull over.

In the rule model shown in Figure 4, the maneuvers introduced in Section II-B are summed up in the maneuver group movement. Such grouping is sensible for maneuvers holding identical transitions to or from another maneuver. This applies for transitions between the maneuver group movement and the primitive maneuver stop. Another grouping is defined inside the movement group. Depending on the presence of a participant on the same lane and inside the entity's perception layer, the maneuver free-cruise or the maneuver group lead-participant-dependent is available. Inside the latter, transitions between approach and following depend on the level of the perception layer including the leading participant.

The approach provides the opportunity to complement rule sets similar to the maneuver transitions in a later stage.

III. SPECIFIED DOMAIN MODEL

The presented modeling concept was developed and realized utilizing the Eclipse Modeling Framework (EMF) [24]. Based on the corresponding Ecore meta model, the EMF includes tools for code generation and model based application development. Especially its ability to create Eclipse Applications based on the specified meta model and the variety of existing plug-ins for implementation motivated this choice.

The suggested domain model representation realizing our conceptual approach is shown in Figure 5. The meta model classes are briefly introduced in the following. *Scenario*, the root class of the meta model, represents the entirety. Referring to Geyer et al. [13] and our presented modeling concept, the class *Scenario* comprises the classes *Scenery*, *Act* and *Participant*. There is exactly one scenery for each instance of a scenario, containing all geo-spatially static objects and globally valid entities. The class *Weather* exemplary

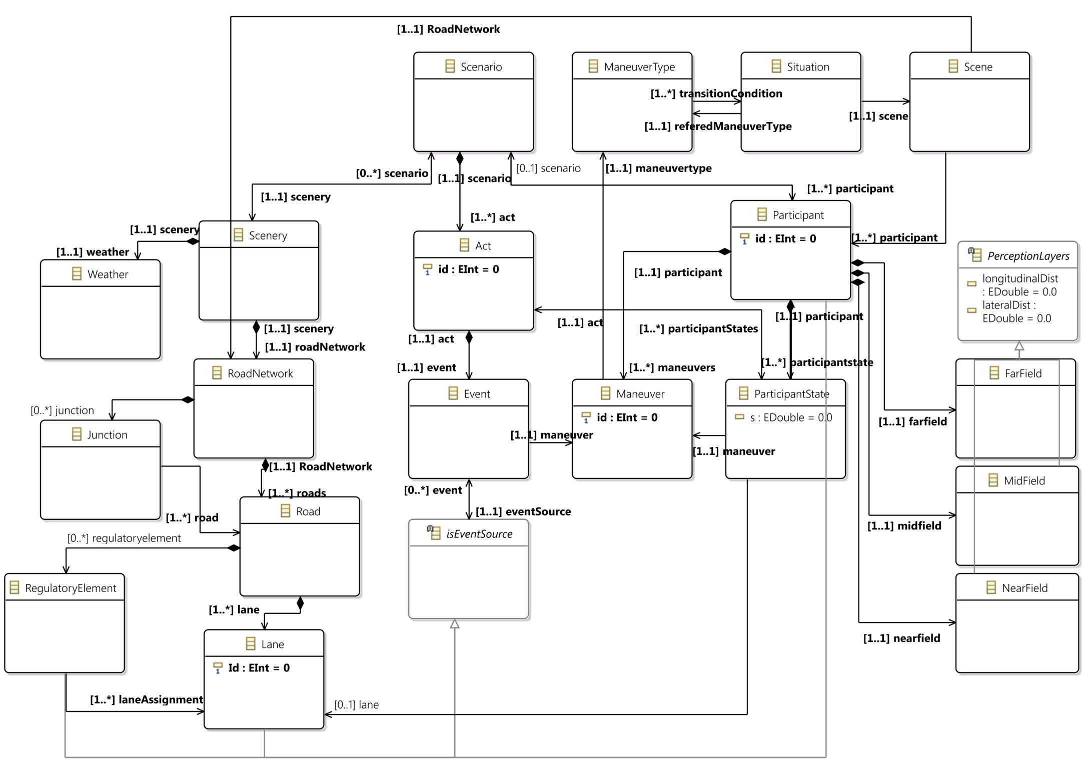

Fig. 5. Domain model implementation within Eclipse Modeling Framework (EMF)

represents such a globally valid entity.

The class *RoadNetwork*, which is contained in the scenery, represents all geo-spatially static objects related to road traffic. The road network is based on the OpenDRIVE format [21], tailored to relevant entities. In principle, the class *RoadNetwork* contains one or more objects of class *Road* and an arbitrary number of junctions. Whereas in the OpenDRIVE format the road accumulates detailed geometries, the details were omitted in the abstract scenario representation. They are appended during the previously introduced parametrization step. In our representation, the class *Road* depicts a logical entity with undefined topology. Therefore, it is reduced to contain one or more lanes and an arbitrary number of regulatory elements. The class *RegulatoryElement* exemplary represents traffic signs and traffic lights. The class *Junction* describes virtual interconnections between road segments. The actual linkage is realized by referenced road entities.

The class *Participant* constitutes the entities able to perform actions and interactions. Each participant contains the three introduced perception layers *FarField*, *MidField* and *NearField*, as well as the aggregation of all maneuvers it is going to perform during the scenario. To place these maneuvers in context of specific acts, the class *ParticipantState* is used. This concept facilitates a representation of participants in each act including its distinct maneuver. By means of *ParticipantState*, each participant is assigned with a lane and a relative position thereon during the according act. Hence, the particular state of the dynamic entities is specified. As we aim for a holistic representation of the scenario, each participant is obtained across all acts.

The abstract class *PerceptionLayers* contains geometric attributes to provide differentiation between the layers and to enable a spatial representation on a logical level. Succession of acts represent the course of actions and interactions performed during the scenario. The class *Act* references objects of the class *ParticipantState*. Temporal progression between acts is expressed by events. The class *Event* holds a reference to the triggering instance and the succeeding maneuver. The abstract class *isEventSource* represents the possibility to trigger an event. In the presented version of the domain model this capacity is inherited by the classes *Participant*, *RegulatoryElement* and *Lane*.

The intention of a participant within an act is characterized by its maneuver. The class *Maneuver* holds a reference to the class *ManeuverType* to determine and classify similar intended actions. Instances of *ManeuverType* can be referred to rule graphs introduced in Section II-C. A reference to the class Situation represents the transition condition, which leads to succeeding maneuvers.

The situation, which is used in describing the maneuver rule graph, links the behavior of generic participants to a corresponding generic *Scene*. The scene provides a reduced and generalized view of a road network occupied by participants.

IV. Prototype implementation

The presented meta model was utilized in a prototype implementation of the described graphical scenario editor introduced in Section II. Applying the EMF code generation facility to the meta model, the basic Eclipse application was generated. The user interface of the application integrates the Graphiti plug-in for graphical representation. As shown in Figure 6, the movie strip approach is realized using Graphiti diagram. The functional implementation for designing the scenario links instances of the meta model with visual elements rendered by Graphiti. Besides a complete and comprehensive graphical representation of scenario instances

Fig. 6. Prototype implementation of the movie strip inspired scenario modeling editor based on Eclipse Graphiti plug-in visualizing an instance of the Approach-and-Follow maneuver.

for manual analysis and refinement, a significant purpose of the implemented prototype is the validation of the defined meta model.

The current implementation supports graphical specification and editing of single road scenarios with an arbitrary number of lanes and participants. The given screenshot picks up the exemplary Approach-and-Follow scenario introduced in Section II-B. The participants' positions represent the logical relation within the act. Spacing within the scenario instance does not resolve to physical distance on a geo-spatial accurate road model.

The model based description of driving scenarios represents the centerpiece for requirements specification and automation of test case derivation. Enriched and refined rule sets enable verification of logical and, in a further step, physical specification. The presented example scenario could be initialized with a seed parametrization in order to autogenerate mutated test cases for ACC validation. Common variations could include different absolute and relative velocities of the participants, as well as selection of different time-gaps in the ACC system.

V. Conclusion and Future Work

In this approach, a concept for specification of driving scenarios based on a domain model was introduced. The essential core of our concept is the abstraction of temporal and spatial information to enable comprehensible modeling of the scenario. The approach utilizes a movie-related view, consisting of different acts and maneuvers. This supports seamless development and validation of automated driving functions, as well as specification of requirements and automated derivation of test cases. Initial verification of the concept was achieved by an Eclipse prototype implementation of the scenario design approach.

Beside further validation and refinement of the presented domain model based on more complex scenarios, the approach should be extended to support multi road sceneries. Implementation of the outlined parametrization phase will support derivation of simulator-ready test cases and facilitate estimation of test coverage. Furthermore, parsing of recorded in field data may support classification and evaluation of real world tests according to specified requirements. Interdisciplinary exchange of scenarios in between different roles and teams as well as merging conceived and derived scenarios within a comprehensive database will foster integrity of requirements and test activities.

Acknowledgment

This document is partially based on the LogDaSim project, which has been funded by the German Federal Ministry of Education and Research (BMBF) under the grant 01IS14024C. The responsibility for the content rests with the authors.

References

[1] "Systems and software engineering - life cycle processes - requirements engineering," *ISO/IEC/IEEE 29148:2011(E)*, pp. 1–94, Dec 2011.

[2] "Software and systems engineering - software testing - part 2: Test processes," *ISO/IEC/IEEE 29119-2:2013(E)*, pp. 1–68, Sept 2013.

[3] *OMG Unified Modeling Language (OMG UML)*, Object Management Group, 2013.

[4] *OMG Systems Modeling Language (OMG SysML)*, Object Management Group, 2012.

[5] E. Sax, *Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie*. München: Hanser, Carl, 2008.

[6] "Road vehicles - functional safety - part3: Concept phase," *ISO 26262-3:2011(E)*, pp. 1–25, Nov 2011.

[7] H.-G. Wahl, K.-L. Bauer, F. Gauterin, and M. Holzäpfel, "A real-time capable enhanced dynamic programming approach for predictive optimal cruise control in hybrid electric vehicles," in *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, Oct 2013, pp. 1662–1667.

[8] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumo simulation of urban mobility," *International Journal on Advances in Systems and Measurements*, vol. 5, no. 3 & 4, pp. 128–137, 2012.

[9] K. Neumann-Cosel, "Virtual test drive - simulation umfeldbasierter fahrzeugfunktionen," Ph.D. dissertation, Technische Universitt Mnchen, Fakultt fr Informatik, 2013. [Online]. Available: http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20140206-1126934-0-2

[10] M. Zofka, R. Kohlhaas, T. Schamm, and J. Zo?llner, "Semivirtual simulations for the evaluation of vision-based adas," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, June 2014, pp. 121–126.

[11] J. Bach, K.-L. Bauer, M. Holzäpfel, M. Hillenbrand, and E. Sax, "Control based driving assistant functions test using recorded in field data," in *7. Tagung Fahrerassistenzsysteme*, 2015. [Online]. Available: https://mediatum.ub.tum.de/node?id=1285215

[12] I. Jacobson, G. Booch, and J. Rumbaugh, *The Unified Software Development Process*. Boston, MA: Addison-Wesley, 1999.

[13] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weißgerber, K. Bengler, R. Bruder, F. Flemisch, and H. Winner, "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183–189, 2013. [Online]. Available: http://tubiblio.ulb.tu-darmstadt.de/62301/

[14] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, Sept 2015, pp. 982–988.

[15] C. Richard, J. Campbell, and J. Brown, "Task analysis of intersection driving scenarios: Information processing bottlenecks," Battelle Human Factors Transportation Center, Seattle, Tech. Rep., 2006.

[16] C. Guo, K. Kidono, and Y. Kojima, "Understanding surrounding vehicles in urban traffic scenarios based on a low-cost lane graph," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, June 2015, pp. 511–518.

[17] *IPG Documentation - CarMaker - User's Guide Version 4.5.2*, 4th ed., IPG Automotive GmbH, Karlsruhe, 2014.

[18] Z. Xiong, "Creating a computing environment in a driving simulator to orchestrate scenarios with autonomous vehicles," Ph.D. dissertation, The University of Leeds, Institute for Transport Studies & School of Computing, 2013.

[19] E. Evans, *Domain-Driven Design. Tackling Complexity in the Heart of Software.* Addison-Wesley, Aug. 2003.

[20] "Oxford dictionaries," http://www.oxforddictionaries.com.

[21] M. D. et al., *Format Specification, Rev. 1.4 - DRAFT*, e ed., VIRES Simulationstechnologie GmbH, Bad Aibling, 2015.

[22] H.-H. Nagel, W. Enkelmann, and G. Struck, "Fhg-co-driver: From map-guided automatic driving by machine vision to a cooperative driver support," *Mathematical and Computer Modelling*, vol. 22, no. 4-7, pp. 185–212, Aug-Oct. 1995.

[23] "Software and systems engineering - software testing - part 4: Test techniques," *ISO/IEC/IEEE 29119-4:2015*, pp. 1–149, Dec 2015.

[24] "Eclipse modeling framework (emf)," https://eclipse.org/modeling/emf.