

Case study on the use of SDL for Specifying an IETF micro mobility protocol

Telemaco Melia, Amardeo Sarma, NEC Europe Ltd., Germany

[meliasarma@netlab.nec.de](mailto:{meliasarma}@netlab.nec.de)

Rui Aguiar, IT Aveiro, Portugal

rui1aa@det.ua.pt

Dieter Hogrefe, Institute for Informatics, University of Goettingen, Germany

hogrefe@informatik.uni-goettingen.de

Abstract—Recent protocols are become increasingly complex, and lead to further level of complexity when used in combination, often resulting in ambiguous behavior. This paper, starting from a case study, presents an improved approach to validate protocols in a tight loop, leading to faster and higher quality standards. The formal specifications can be used as supplementary material to resolve behavior that is ambiguous from only reading the standards, mostly specified in ASCII text. Further, it addresses how to validate the behavior of several protocols running together without the need for several independent implementations. The approach allows early validation using a large number of environmental triggers, including external unexpected behavior. Finally, useful guidelines are provided to allow easy development of such mobility environments by means of SDL tools.

Index Terms—

I. INTRODUCTION

Modern all IP based communication systems are constantly increasing the levels of complexity and becoming comparable to telecommunication protocols. The composition of different protocols, that are proven to work properly in a stand alone manner, introduces new flaws leading to extreme integration difficulties and protocols that are not integrated as intended by the inventors. The specifications do not always clarify how such protocols should behave during operations that deviate from the normal (e.g. link failure, idle states), increasing the potential of new bugs that often need to be solved in an ad hoc manner. Typically, within the IETF world, before a protocol is consolidated towards an RFC (Request for Comment), several implementations from different partners (typically the authors of the standard) have to be tested and proved to interoperate. Interoperability tests are usually the main event of the standard document lifetime where errors and wrong protocol specifications are detected and eventually corrected through exhaustive debugging sessions. Although this approach is valid to run basic implementations and to assess the general quality of a specification document, it does not completely solve the problem of precisely specifying how implementers should tackle ambiguous statements. Typically, the partners

contributing to the standard play the key role of taking decisions and resolving ambiguous conditions.

The lifecycle of a protocol is therefore quite long, and several iterations are needed before the specifications can be successfully checked against running implementations. Thus, to overcome these and other shortcomings, some approaches that support protocol validation have been widely studied and proposed.

Formal Methods (FMs) are techniques that provide rigorous specifications for software development, achieving provable correctness and reliability in the various steps of system design and implementation. These have been extensively applied for PSTN, 2G (2nd Generation) and 3G (3rd Generation) protocols, but applications in the IETF area have been few so far. Among the different proposed methods, we have chosen the SDL approach that can better fulfill our requirements of dynamic model verification and modeling of IP protocols. With its strong formal basis, SDL can specify and validate the protocol behavior. We address modeling and analysis of a specific handover control mechanism in an IP based world. The approach chosen to validate the specifications is the commercial SDL Telelogic [1] tool already presented in other works as referred to in section **Fehler! Verweisquelle konnte nicht gefunden werden.** SDL provides the capability to specify the protocol behavior as a state machine and to trigger a transaction reacting to a stimulus either generated by the system itself (e.g. the reception of a signal, timer expiration) or sent by the external environment (e.g. sent by the tester during the validation process). We demonstrate the benefit of translating specifications from text format into a formal specification and how this formal specification can be finally validated to meet input requirements. We shorten the typical lifecycle of an implementation based approach such as within the IETF and we propose an innovative approach to prove protocol behavior correctness. Finally, by providing both text and formal specifications, implementers have no possibility to introduce wrong statements while implementing and testing prototypes.

The remainder of the paper proposes a case study of an IP micro mobility scheme recently proposed at the IETF

standardization [2]. Therefore, we assume the reader to be familiar with the terminology and network architecture discussed in the next sections.

The rest of the work is organized as follows. **Fehler! Verweisquelle konnte nicht gefunden werden.** introduces the related work. **Fehler! Verweisquelle konnte nicht gefunden werden.** discusses the model and **Fehler! Verweisquelle konnte nicht gefunden werden.** evaluates the chosen approach. Finally **Fehler! Verweisquelle konnte nicht gefunden werden.** concludes the paper.

II. RELATED WORK

There have been quite a few examples on how to approach formal specifications of IP protocols and how to exploit SDL features to for these purposes.

In [4], the authors propose to use the SDL capabilities to address a message routing problem in IP networks. The problem addressed in the paper deals with the fact that link state failure detection in the Resource Reservation Setup Protocol (RSVP) relies on timer based procedures. The standard specifications however do not properly specify how recovery should be performed. The authors therefore present a formal analysis to solve the identified issues. By means of the SDL specifications, an emulated network is realized. Each process (router) in the system gets a unique identifier, the so-called Process Identifier (PID) used to route the signals to the correct destination, emulating the IP behavior. Since the SDL tool does not have a dynamic routing layer, the authors build a routing layer based on the distance vector routing philosophy to allow the study of the RSVP protocol with link failures. A formal analysis of RSVP state machines is therefore presented.

Yang, Yang and Xiaokang [5] design the Open Shortest Path First (OSPF) protocol for a wireless private network and demonstrate correct implementation and flexibility of the approach, both related to adaptation with respect to changing topologies as well as generally related to code reusability. SDL has also been used by O. Monkewich [7] to compare flooding algorithms in the framework of OSPF standardization at IETF. Noudem and Viho [6] describe the specification and validation of the MIPv6 protocol, and go on to generate test suites using TTCN-3, which serves as a basis for conformance and interoperability testing.

Though protocol behavior is mainly characterized by the state machines definition, timers and error recovery procedures might affect normal operations and introduce unexpected failures. In [8], an interesting approach is proposed to better test communication systems. One of the main problems of protocols based on timers' expiration for error recovery is how to dynamically compute timers' value to adapt to network congestion and unpredictable link failures. The author proposes the implementation of an artificial intelligence layer able to learn from the environment and to make predictions on timers' duration. Although this issue does not directly affect the work proposed, potential problems

related to layer two handovers could be here considered and used to compute accurate predictions on how layer two conditions affect the overall handover procedure.

Taking into account the above mentioned and other similar examples, we propose a novel approach on how to introduce mobility at SDL level and enable the emulated environment to route signals accordingly to mobility patterns. Hence, we introduce a new entity to collect all the PIDs and to redistribute a specific PID upon request. This specific feature is proposed to overcome the problem of how to simulate the dynamic behavior (due to mobility) of the considered environment. A bootstrapping phase is required to connect the nodes after selecting a connection point to the network. Therefore, routing is performed via the specified SDL connectors. To the best of our knowledge, formal models have so far always focused on a static model in which the network topology is fixed. To allow several tests, the scenario has to be fine tuned at each run. The advantage of the proposed method is to reflect the dynamic behavior typical of wireless mobile devices. This is important to simulate the handover over the wireless channel.

III. THE MODEL

In the following, we address the design of the state machines for an IP micro mobility protocol presented during the 61st IETF meeting [2]. The draft introduces a novel method to manage micro mobility in an IPv6 based environment and the combination of two decision points for handover initiation. The protocol specifies the way to combine Mobile Terminal Initiated Handovers and Network Initiated Handovers. The composition of two different transport protocols is also described and here validated via a unique design of reliable state machines. For a full and exhaustive protocol explanation, we refer to the existing IETF draft.

The key problems to be solved are as follow. First, the concurrency of different decision points, potentially operating at the same time, is solved by introducing detailed state machines that are able to recover from race conditions. Second, the probability to reject a handover decision has to be taken into account. Third, the loss of connection while the terminal performs handover requires setting a timer. A relation exists between the layer-two handoff latency and the timer deleting the current handover process on expiry. The fourth issue is related to packet loss over the wireless link (the packet loss over the wired link is negligible). To do so, timers are required to manage the state machines in case of packets being lost on the wireless link. This is important to make the state machines resilient.

The SDL Telelogic tool was used to address the four points mentioned above. The tool is not intended to evaluate performance (see [9] for combining performance and functional evaluation). It is rather used for validation, and has several functionalities to create statistical processes, emulating what affects protocols operations in a real-life environment. For the particular given task, we make full use of the tool's

ability to generate random numbers as well as events based on a preferred distribution function. This feature is useful, for instance when signals between two processes have to be dropped to simulate a datagram loss over a link transmission, or if a decision is made according to a desired probability function (e.g. Poisson process). The mentioned features, applied to a communication protocol, can be very useful in the design of reliable state machines, thereby reducing “try and error” implementation efforts.

Fehler! Verweisquelle konnte nicht gefunden werden. describes the model architecture and the messages exchanged. We can identify the four main processes. The system can be conceived of as a collection of several access networks interconnected via a core network providing the core services. However, since we are interested in the study of a micro mobility handover management protocol, we focus on the access network design. The access network is therefore modeled as a block composed of several processes. The choice reflects the fact that, in a mobile operator environment, the number of access networks is typically static, which is not true for the number of access routers and mobile devices sending and receiving IP data traffic. SDL specifications do not allow a block (as opposed to processes) to be instantiated several times, whereas access routers and especially mobile devices need this property. As in **Fehler! Verweisquelle konnte nicht gefunden werden.**, we consider one Network Controller (PDP), two access routers and one mobile node. This setup is sufficient to assess the validity of the state machines.

The mobile node is implemented in a less complex manner, and only one state machine has to be maintained and updated following normal protocol operations. The access router is a more complex entity, as it has to implement the logic to handle up to several hundred thousand users. We suggest having fast mechanisms to access the memory such as hash tables (eventually implemented in hardware). A hash key obtained by the unique mobile identifier (e.g. the mobile node home address) is used to get the mobile’s information and to store the “SOFT STATE”. Soft states to model the process have been chosen for performance reasons, and avoid the need to maintain states for several parallel entities.

Connectors are used to exchange messages between the entities. The connector “niho” allows the communication between the external environment (the tester) and the PDP. The messages exchanged are used to start, stop and trigger a manual network initiated handover. The same applies to the connector “mitho” connecting the environment to the mobile terminal. These features are useful to create manual triggers and test the state machines in critical cases. The connectors “mn”, “ar”, “nc” are used to bootstrap the setup phase and to exchange the PID necessary for the dynamic behavior of the model itself. Finally, the “wired” and “wireless” connectors are used to route messages between the terminals, access routers and the network controller. This represents the core of the system, where these messages reflect the draft specifications. For convenience, we keep the same

terminology. However, the content of the packets cannot be considered as ICMPv6 or COPS alike. The overhead of creating such a structure compared to the benefit is not desirable. The specific parameters required for taking decisions are implemented following SDL data structures, since we believe this does not impact the presented studies.

In **Fehler! Verweisquelle konnte nicht gefunden werden.**, an overview of the SDL states is proposed. The states are used to handle the messages and to change the SOFT state of each single entity as in **Fehler! Verweisquelle konnte nicht gefunden werden.** displays the states implemented in the state machines. Please note that they are different from the soft states used to handle the mobile node, the access router and the network controller.

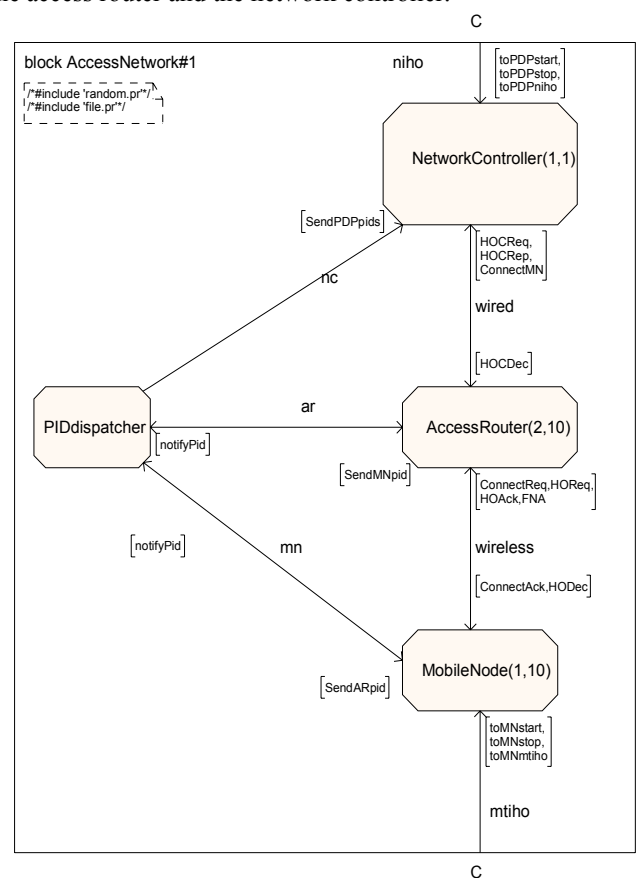


Figure 1 Access Network Architecture

The mobile node is initially waiting for the access routers’ PIDs. Once the PIDs are received, it sends a connection request to one of the selected access routers and waits for acknowledgement of the connection. This handshake can be compared to any authentication mechanism implemented to get network connectivity. After the connection acknowledgement, the mobile node changes its status to “Connected”, and is now ready to randomly initiate handovers and to answer to any request coming from the network. The last state is the one marked as “L2_Handover”. This state is entered when the terminal performs layer two handover. To

emulate the change of layer two connectivity (e.g. change channel in a WLAN handover), an initialized timer controls the attachment to the new access router. During this time the terminal is not reachable.

The access router initially sends its PID to the PID dispatcher, and after getting the mobile nodes' PIDs immediately changes to the state "waiting for connection". Two possible stimuli may trigger the transition to the ready state. Either a mobile node connects, or a timer "waiting for connection" expires. In the "ready" state, the access router can handle all the messages coming from the mobile devices or the network controller.

Finally, the network controller gets the entire access router PIDs and enters the "ready" state recording the connection tuple {access router, mobile node}. It can now perform admission control in case of mobile terminal initiated handover and resource optimization in case of network initiated handover.

Fehler! Verweisquelle konnte nicht gefunden werden., chosen as an example, depicts the mobile node behavior and related SOFT states. From the STEADY state, the terminal can either move to MTIHO (mobile terminal initiated handover) without decision (admission control), or to NIHO_ACK (always performed). In the former case, upon the reply from the Network Controller, the state changes to MTIHO_ACK (before performing the layer two handover). The FNA message is sent on the new link to announce the mobile node to the new access router. The terminal subsequently moves back to the STEADY state. If the handover is not granted or the terminal aborts the handover, the terminal shifts back to the original state.

```
newtype MN_STATE
literals STEADY, NIHO_ACK, MTIHO_NO_DEC, MTIHO_ACK
endnewtype;
```

```
newtype AR_STATE
literals STEADY, MN_NOT_CONNECTED, NIHO_NO_ACK, NIHO_ACK,
MTIHO_NO_DEC, MTIHO_ACK, MTIHO_NO_ACK, TIMER_EXPIRED
endnewtype;
```

```
newtype PDP_STATE
literals STEADY, NIHO_NO_ACK, NIHO_ACK,
MTIHO_ACK, MTIHO_NO_ACK, NO_REPORT, TIMER_EXPIRED
endnewtype;
```

Figure 2 SOFT STATES

The access router has the most complex logic to be implemented. Following the MTIHO branch, the entry can get the MTIHO_NO_DEC value indicating that the handover has not been yet granted, the MTIHO_NO_ACK value indicating that the terminal is granted to move, and MTIHO_ACK value indicating that the terminal is effectively performing handover. If the terminal rejects the handover, the value is back to the STEADY state. Following the NIHO branch, we have the two states corresponding to whether or not the handover has been acknowledged. In both cases, when the handover is completed, the state is back to STEADY. In case

the access router is a target access router, to which a node intends to connect, the state is set to MN_NOT_CONNECTED. A received FNA means that the terminal is already connected, and the handover can be successfully concluded.

Finally, the network controller implements two branches, respectively for network initiated handover and mobile terminal initiated handover. The states are updated according to the protocol operation. In both access router and network controller, a timer is initialized as soon as the handover is initiated. This timer handles the condition in which the handover fails for whatever reason (e.g. layer two handover fails, packet loss).

IV. EVALUATION OF THE APPROACH

A stand-alone emulator was compiled from the C code generated from SDL specifications using the Tau tool. The emulator has been enhanced with the capability to output events to log files to help in the analysis. Hence, to verify the correct behavior of the state machines above specified (e.g. deadlock free), a set of parameters have been fixed. The input values used to trigger a transaction (e.g. network initiated handover, mobile terminal initiated handover, rejection of a handover) are random values uniformly chosen between a min and a max value. One trigger is implemented in the mobile terminal and one in the network controller. The terminal can abort the MTIHO handover with a given probability rate, and in this scenario we assume it always accepts the NIHO from the network. Several min/max values as well as different probability handover abortion rates have been tested. Three relevant results are highlighted, specifically the rate of network initiated handovers, the rate of successful mobile terminal initiated handovers and the rate of rejected mobile terminal initiated handovers. As an example, **Fehler! Verweisquelle konnte nicht gefunden werden.** summarizes the results in which the following values have been chosen: rate of acceptance for MTIHO fixed at 0.8, rate of rejected MTIHO fixed at 0.2, random min/max value for MTIHO trigger fixed at {1,10} units of time, random min/max value for NIHO fixed at {1,100} units of time. For simplicity, we assume that only mobile terminal initiated handovers can be aborted from the terminal side, whilst network initiated handovers are always supported by the terminal. The graph proves, out of more than one million units of time, how the system remains stable and maintains on the average the same number of performed handovers observed at fixed intervals.

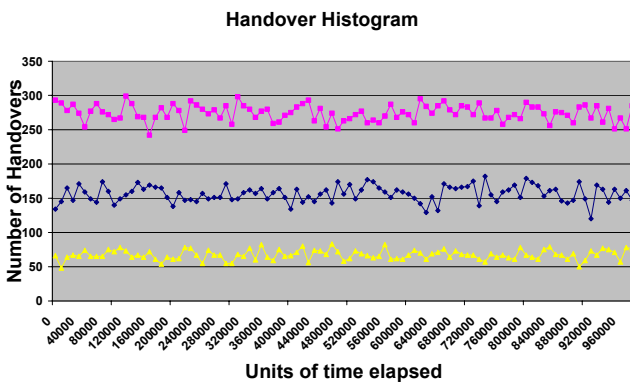


Figure 3: in yellow rejected mobile terminal initiated handovers; in blue network initiated handovers completed and in violet mobile terminal initiated handovers completed.

Following the introduced approach more complex scenarios can be implemented and analyzed. Fine tuning of the input parameters can be adjusted to meet desired requirements. We believe such emulators can be useful to determine network behavior with little implementation effort and shortening typical lengthy development loops.

Finally, the proposed methodology can be easily extended to general mobility frameworks requiring handovers analysis as well as network behavior design depending on different user mobility patterns, network policies for admission control and resource management.

V. CONCLUSIONS

In this paper, we have presented an approach to specify and then validate protocols within a wide range of scenarios in a formal manner. The advantage of the approach lies in a very fast adaptation of protocol specification and their complex interactions and to provide triggers based on different assumptions. This allows us to validate the stability of the protocol for a large period of time under very different conditions. In particular, we have shown how to apply the approach for a combination of network initiated and mobile terminal initiated handovers. We see this as very promising to be extended beyond the handover scenarios to be applied to mobility in general. Ongoing work in the Daidalos [3] project will look into this aspect.

ACKNOWLEDGEMENT

The presented work originated in the framework of the IST Daidalos project [3]

REFERENCES

[1] SDL Telelogic Tau, www.telelogic.com
 [2] IETF, <http://www.ietf.org/internet-drafts/draft-meliambobopts-niho-fmip-00.txt>
 [3] The *Daidalos* project, www.ist-daidalos.org

[4] Werner, C., Fu, X., Hogrefe, D.: *Modeling Route Change in Soft-State Signaling Protocols using SDL: A Case of RSVP*. Proceedings of the 12th International SDL Forum, Lecture Notes in Computer Science 3520. p174 – 186. Springer-Verlag. 2005
 [5] Yang, Y., Yang, L., Xiaokang, L.: *SDL Design of OSPF Protocol for the Wireless Private Network*, Proceedings of the 12th International SDL Forum, Lecture Notes in Computer Science 3520. p149 – 161. Springer-Verlag. 2005
 [6] Noudem, F. N., Viho, C.: *Modeling, Verifying and Testing Mobility Protocol from SDL Language*, Proceedings of the 12th International SDL Forum, Lecture Notes in Computer Science 3520. p198 – 209. Springer-Verlag. 2005
 [7] O. Monkewich, I. Sales, and R. Probert (2001), *OSPF Efficient LSA Refreshment Function in SDL*. In: Tenth SDL Forum (SDL'01), Copenhagen, Denmark, June 2001.
 [8] K. Naik, *Designing Reliable Test Architectures for Communication Protocols*, The Computer Journal 1997 40(7), pages 441-456.
 [9] Kuhn, T., Gerald, A., Gotzhein, R. and Rothländer, F.: *ns+SDL – The Network Simulator for SDL Systems*, Proceedings of the 12th International SDL Forum, Lecture Notes in Computer Science 3520. p103 – 116. Springer-Verlag. 2005
 [10]