A Tool Concept for Feature Interaction Detection in Intelligent Networks

Dirk O. Keck

University of Stuttgart
Institute of Communications Switching and Data Technics
Prof. Dr.-Ing. Paul J. Kühn
Seidenstr. 36, D-70174 Stuttgart (Germany)
Phone: +49-711-121 2484, Fax: +49-711-121 2477
E-mail: keck@ind.uni-stuttgart.de

Abstract. The architecture of the Intelligent Network (IN) has been introduced to provide a platform for the rapid development and deployment of new services and service features to satisfy customer needs. The service designer is supported by a Service Creation Environment (SCE), a software tool that allows to design new services and service features on a high level of abstraction. Unfortunately, even if each single service feature is designed and tested properly and behaves according to its specification, unwanted and adverse behaviour may occur when it is activated in combination with other features or other instances of the same feature. This so-called *feature interaction problem* is a major obstacle for the development and introduction of new services, as it may confuse customers or even compromise network operation.

In this contribution, a concept for an extension of a Service Creation Environment is presented, capable of supporting service developers to detect interactions between IN service features in an early stage of development. Our approach uses a formal description technique (SDL) for the specification of the features, and formal verification techniques combined with the simulation of IN call and service processing for detection of unwanted interactions. The tool is divided into two complementary parts: a simulator that models basic call processing based on the call state models of ITU-T IN Capability Set 1, and an analytical part, able to identify interactions within services and combinations of services based on analysis and verification of service specifications.

1 Introduction

The Intelligent Network (IN) is an architecture that supports the rapid and cost efficient introduction of new services and service features into a communication network. While in conventional non-IN switching technology integration of new service features comes along with an software update for the whole switching system, IN separates call processing from service processing and provides an interface in between, the Intelligent Network Application Protocol (INAP) standardised by the ITU-T (Recommendation Q.1218), and its European counterpart, the ETSI Core INAP.

The IN architectural concept is complemented by a design methodology for services and service features on a high level of abstraction based on so-called *Service Independent Building Blocks (SIBs).* A software tool called *Service Creation Environment (SCE)* supports the design of new services with graphical editing facilities.

This research is carried out in cooperation with Alcatel SEL AG, Stuttgart, and is supported by the Software-Labor of the University of Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart (Germany).

SUBMITTED VERSION printed 04/01/96
to be presented at the 5th Open Wksp. on High Speed Networks
March 20-21, 1996, Paris (France)

2 The Feature Interaction Problem

The large and continuously growing number of coexisting service features in an IN environment lead to the occurrence of mutual interactions between different service features or different instances of the same feature. This phenomenon is called the *Feature Interaction Problem,* a term which has been coined by Bowen et al. [5]. It is currently one of the major obstacles to the rapid introduction of new services and service features. Zave [24] defines a feature interaction as "a case, in which system behaviour as a whole does not satisfy the separate specification of all its features." It has to be emphasized, that no distinction between *wanted* and *unwanted* interactions is made, because "the goal of eliminating feature interactions has nothing to do with discouraging or encouraging any kind of behaviour in telecommunications systems." This definition is used throughout this contribution.

As long as there is no viable solution for the feature interaction problem, the advantages of rapid service creation cannot be fully exploited. The risks of unidentified interactions are too high, with consequences like customer confusion, massive financial losses, or even network breakdown. With our concept we want to show how a feature interaction toolset can be integrated into a Service Creation Environment.

In the next section, an overview on methods for interaction treatment is given. A concept for a software toolset for interaction detection to complement a Service Creation Environment is presented in section 4.

3 Methods for Interaction Treatment

3.1 Classification of Methods

According to Cameron and Velthuijsen [6], there are three different approaches to tackle the feature interaction problem in IN. The *avoidance* of interactions relies on the use of a suitable architecture and design methodology in order to prevent the occurrence of any interaction. The *detection* refers to techniques that are able to identify interactions in a given set of service features, whereas *resolution* supplies the means to resolve these interactions. These methods are further divided with respect to the phase during a service lifecycle where they are applied. *Off-line* detection methods are applied during an early phase in the lifecycle, i.e., prior to the deployment of a service feature into the network. *On-line* methods are applied after the deployment of a service.

The concept presented in this contribution serves as an extension to a Service Creation Environment (SCE) and belongs therefore to the category of *off-line* methods. It aims at supporting the service designer at the *detection* of feature interactions. Their resolution, however, is currently out of the scope of this tool concept and remains to be carried out by the service designer. The different ways to attack the problem of the detection of feature interactions in an IN are described in the next three subsections.

3.2 Formal Specification and Verification

The most ambitious way to detect feature interactions is the application of formal specification techniques in combination with formal verification methods. Formal description techniques (FDTs) can be used to establish concise and unambiguous specifications of IN service features and their underlying infrastructure. Furthermore, they are a necessary precondition for the application of formal verification techniques for feature interaction detection.

There exists already a significant number of approaches to feature interaction detection using several formal description and verification techniques, like the ITU-T Specification and

SUBMITTED VERSION printed 04/01/96
to be presented at the 5th Open Wksp. on High Speed Networks
March 20-21, 1996, Paris (France)

Description Language (SDL) [3,7,9,16,20], the Language Of Temporal Ordering Specification (LOTOS) [4,10,11,12,22,23], Petri Nets [8], Temporal Logic [1,2], State Transition Rules (STR) [14,15,17,18,21], and Labelled Transition Systems (LTS) [13].

The use of formal methods for feature interaction detection comes along with some difficulties, that are caused either by the principal restrictions of formal verification or by the specific properties of the specified system. Examples for these problems are:

- The well-known *state explosion problem.* Specification of interacting service features tend to be relatively complex. This has the consequence that the formal verification by exhaustive state space exploration may be impossible because of memory requirements, if the granularity chosen for the specification of the problem is too fine.
- The *complexity and heterogeneity of a real-world telecommunications platform* that may involve IN, ISDN, and GSM services and features cannot be mapped onto a formal specification which is still suitable for the application of formal verification techniques. Simplifications are necessary, reducing the chance of spotting *all* interactions.

These problems lead to the insight that formal verification cannot be the only technique to rely on for interaction detection.

3.3 Simulation

It is obvious that feature interactions can be observed in a simulated system as well as in the real world, provided that the model has been established carefully. However, it may be difficult to find an interaction when there is no hint in which situation it will occur. Therefore, interactive simulation of service processing in an IN environment is not suitable for identifying interactions when there is no idea in which situation they appear, but it is an excellent aid to intuitively understand the cause of an interaction by directly observing it and its consequences. Thus, simulation is a good complement to formal verification, as the results of the latter sometimes are difficult to interpret.

3.4 Testing

If simulation is accompanied by a systematic testing methodology and if the generation of test cases is supported by the formal specification technique used, interactions can be identified in the simulated system as well as test procedures for the integration test in the real IN system can be established in a systematic way.

4 A Tool Concept for Interaction Detection

As none of the methods listed above is able to cover the whole field of off-line feature interaction detection alone, the tool concept presented here uses a combination of them. Its structure, as shown in Fig. 1, is sketched in the following paragraphs.

Interface to the SCE. In the Service Creation Environment, a service designer constructs service features based on Service Independent Building Blocks (SIBs), which can be connected to a directed graph structure representing the service logic. For interaction detection, however, this information is not sufficient. Additional descriptions of the internal mechanisms of each SIB and the data structures involved are required. The final description of an IN service feature mainly consists of three parts:

- *Specification of the SIBs involved.* SDL is used for this description, as it is suitable for the description of communications services and it is a standardised language (ITU-T Recommendation Z.100) understood by almost all telecommunications engineers.

SUBMITTED VERSION printed 04/01/96
to be presented at the 5th Open Wksp. on High Speed Networks
March 20-21, 1996, Paris (France)

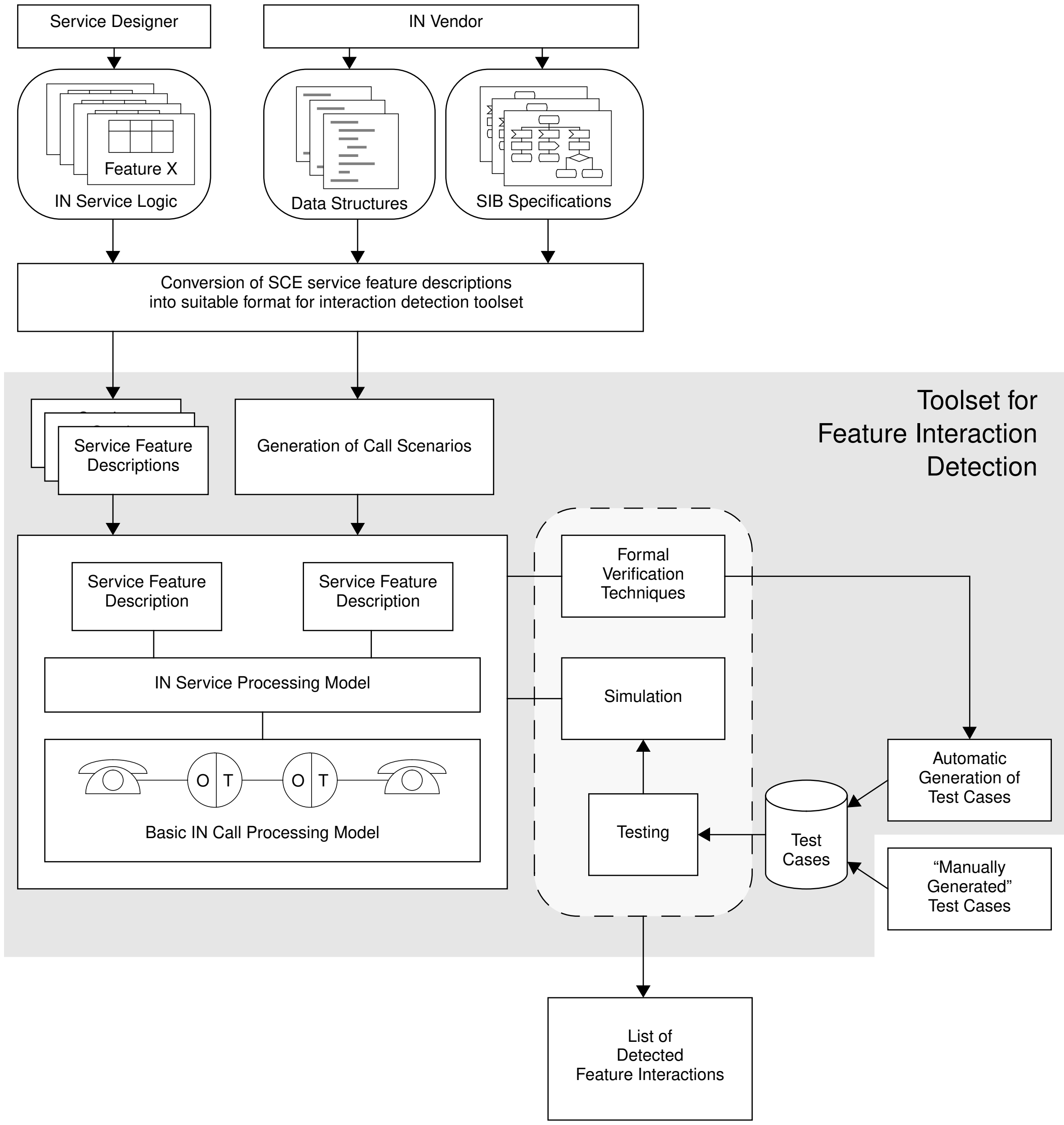

Figure 1: Structure of the toolset for feature interaction detection.

- *Description of data structures used by the service feature.* A standardised notation, preferably ASN.1 (Abstract Syntax Notation 1) or C syntax is used for this purpose.
- *Tabular data representing the interconnections of the SIBs* (directed graph structure). This are mainly the edges between the logical ends of SIBs and the logical starts of their subsequent SIBs.

The service designer usually only has access to the tabular data listed in the third item, the interconnection structure of the SIBs, the design of which is supported by the SCE. The internal SIB functionalities and data structures mentioned in the first two items require deeper insight into the internal architecture of the system. They are provided by the IN vendor or an associated company which has the knowledge to carry out these tasks.

This data provides the information base for the interaction detection toolset. The *crucial* part for the applicability of detection methods is the specification of the internal SIB functionalities.

On the one hand, it has to be as detailed as necessary for a thorough description of the service logic, on the other hand as coarse as possible for the application of formal verification methods for interaction detection. As it is not the primary purpose of the description parts listed above to serve as an input to feature interaction detection, it is necessary to convert them into a suitable format before the toolset is applied.

Model of IN call processing. An effective model for call processing is an essential part for the toolset, as service logic alone is not sufficient to describe the operation of a service feature. It is important that the accuracy of the model can be adapted to the used method, i.e., that it is possible to use a simpler model with less options when verification techniques are applied (in order to keep the state space small), but a complex model with all possible options when an interaction is examined using the simulation utility.

Generation of call scenarios. There are two typical situations where a feature interaction detection tool will be applied:

- *A new service feature has to be integrated into an existing system.* In this situation, a list of feature pairs to be checked must be generated. For each one of the pairs in this list, a complete collection of call scenarios involving several subscribers and an instance of each of the features of this pair has to be built. A method for the establishment of such scenarios is described in [19]. Afterwards, the verification, simulation or testing tools have to be applied on these call scenarios.

- *The cause of an interaction that has been reported by a user has to be understood.* In this case, the set of features involved in the problem is known. Therefore, the list of scenarios that has to be checked is much smaller, in the most pleasant case there is only one such scenario (if the users' reports are very precise).

Formal verification. For each of the call scenarios, a model containing the underlying call processing, the service processing, and the two service features has to be established. In order to allow the successful application of formal verification techniques, they have to be kept as simple as possible. There is also the possibility to reduce the models further and to verify different aspects of the service sequentially. As there are not many SDL verification tools (mainly SDT and Geode) and as these tools tend to be extremely complex, the suitability of other specification methods has to be examined.

Simulation. Simulation of service features for IN is based on a model of the IN distributed functional plane (ITU-T Recommendation Q.1214) for call and service processing. The model components (mainly CCF, SSF, SCF) in the IN model are mapped on processes on a multiprocessing workstation, and the communication between the components is mapped to interprocess communication. The concurrent execution of service features triggered by the same detection point is not allowed, as this would introduce ambiguity and contradictions.

Testing. The inclusion of a testing methodology into this toolset has got two goals for feature interaction detection:

- If the complexity of the interworking of two service features is too high for the application of formal verification, this work has to be done partly by testing. As there is a large number of different call scenarios to be tested, the system usually cannot be tested exhaustively. The major drawback of testing is that there is always the possibility not to reach relevant (interacting) areas, although this problem can be reduced by a careful selection of test cases.

- Test cases established during the off-line detection phase can be re-used for the integration test of a new feature into the existing system. This is an important argument, as it cannot be

SUBMITTED VERSION printed 04/01/96
to be presented at the 5th Open Wksp. on High Speed Networks
March 20-21, 1996, Paris (France)

expected that new service features will be put into operation by network and service providers without having tested them in a real IN environment.

Furthermore, the availability of a simulator allows the validation of test cases for the integration test created “manually” or by use of other tools.

Evaluation of results. The results obtained from the application of the toolset described here is a list of detected feature interactions and the scenarios in which they occur. The set of test sequences are regarded as a by-product in this context and are not considered here any further. It is out of the scope of this toolset, at least at the present time, to support the service designers to resolve any interactions. It has to be recalled, that no distinction between “wanted” and “unwanted” interactions is made. Whether the combination of two interacting features will be prevented, or changes in the specification or design of the features are made, or the interaction is accepted, as it is “logical” in a sense that the service user expects this behaviour, has to be decided by the service designer.

5 Conclusions

In this contribution, a tool concept for off-line interaction detection for IN service features is presented. It serves as an extension of a Service Creation Environment. Its main part using formal verification techniques for the identification of interactions is accompanied by a simulator which is able to simulate IN call and service processing on the IN distributed functional plane. Additionally, testing is integrated in the toolset, because the formal specifications of service features can be used to derive test cases and the capabilities of the simulator can be enhanced by a systematic testing methodology. The test cases can be re-used for an integration test taking place when a new feature is deployed into the IN.

Even if the goals of the tool can be fulfilled, some important limitations remain: the list of call scenarios to be checked probably will become quite large, so that an “explosion” problem must be expected when the number of features grows. There is also the large field of interaction resolution that is presently not covered at all by this toolset. However, it is out of question that, in spite of the limitations mentioned above, such a toolset is a valuable support for IN service designers.

References

[1] J. Blom, B. Jonsson, L. Kempe, “Using Temporal Logic for Modular Specification of Telephone Services,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 197-216.

[2] J. Blom, R. Bol, L. Kempe, “Automatic Detection of Feature Interactions in Temporal Logic,” in *Feature Interactions in Telecommunications Systems, III,* K. E. Cheng, T. Ohta (Eds.), IOS Press, Amsterdam, 1995, pp. 1-19.

[3] W. Bouma, W. Levelt, A. Melisse, K. Middelburg, L. Verhaard, “Formalisation of Properties for Feature Interaction Detection: Experience in a Real-Life Situation,” in *Towards a Pan-European Telecommunication Service Infrastructure - IS&N ’94,* H.-J. Kugler, A. Mullery, N. Niebert (Eds.), Lecture Notes in Computer Science 851, Springer-Verlag, 1994, pp. 393-405.

[4] R. Boumezbeur, L. Logrippo, “Specifying Telephone Systems in LOTOS,” *IEEE Communications Magazine,* Vol. 31, No. 8, August 1993, pp. 38-45.

[5] T. F. Bowen, F. S. Dworack, C. H. Chow, N. Griffeth, G. E. Herman, Y.-J. Lin, “The Feature Interaction Problem in Telecommunications Systems,” *Proceedings of the 7th International Conference on Software Engineering for Telecommunications Switching Systems,* Bournemouth, U.K., July 1989, pp. 59-62.

[6] E. J. Cameron, H. Velthuijsen, “Feature Interactions in Telecommunications Systems,” *IEEE Communications Magazine,* Vol. 31, No. 8, August 1993, pp. 18-23.

SUBMITTED VERSION printed 04/01/96
to be presented at the 5th Open Wksp. on High Speed Networks
March 20-21, 1996, Paris (France)

[7] K. E. Cheng, “Towards a Formal Model for Incremental Service Specification and Interaction Management Support,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 152-166.

[8] J. H. Choi, H. S. Kim, W. J. Lee, Y. R. Kwon, “A Petri-Nets Based Approach for Detecting Feature Interactions in Telecommunications Services,” *Proceedings of the 12th International Conference on Computer Communications (ICCC),* Seoul (Korea), August 1995, pp. 596-601.

[9] P. Combes, S. Pickin, “Formalisation of a User View of Network and Services for Feature Interaction Detection,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 120-135.

[10] O. C. Dahl, E. Najm, “Specification & Detection of IN Service Interference Using LOTOS,” in *Formal Description Techniques, VI (C-22),* R. L. Tenney, P. D. Amer, M. Ü. Uyar (Eds.), Elsevier Science B. V. (North-Holland), Amsterdam, 1994, pp. 53-69.

[11] M. Faci, *Detecting Feature Interactions in Telecommunications Systems Designs,* PhD Thesis, Department of Computer Science, University of Ottawa, Ontario (Canada), 1995.

[12] M. Faci, L. Logrippo, “Specifying Features and Analysing Their Interactions in a LOTOS Environment,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 136-151.

[13] A. Gammelgaard, J. E. Kristensen, “Interaction detection, a logical approach,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 178-196.

[14] Y. Harada, Y. Hirakawa, T. Takenaka, N. Terashima, “A Conflict Detection Support Method for Telecommunication Service Descriptions,” *IEICE Transactions on Communications,* Vol. E75-B, No. 10, October 1992, pp. 986-997.

[15] Y. Harada, Y. Hirakawa, T. Takenaka, “A Design Support Method for Telecommunication Service Interaction,” *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM ’91),* Phoenix (Arizona), December 1991, pp. 1661-1666.

[16] M. Hori, M. Ito, Y. Fukunaga, T. Nagaishi, “A Verification Method for Service Script of the Intelligent Network,” *IEEE Global Telecommunications Conference (GLOBECOM ’92),* Orlando (Fla.), December 1992, pp. 380-384.

[17] Y. Inoue, K. Takami, T. Ohta, “Automatic Detection of Service Interactions in Telecommunications Service Specifications,” *IEEE International Conference on Communications,* May 1994, pp. 1835-1841.

[18] Y. Inoue, K. Takami, T. Ohta, “Method for Supporting Detection and Elimination of Feature Interaction in a Telecommunication System,” *International Workshop on Feature Interactions in Telecommunications Software Systems,* St. Petersburg (Fla.), December 1992, pp. 61-81.

[19] D. O. Keck, “Identification of Call Scenarios with Potential Feature Interactions,” to be presented at the *International Workshop on Advanced Intelligent Networks (AIN ’96),* Passau (Germany), March 1996.

[20] B. Kelly, M. Crowther, J. King, “Feature Interaction Detection Using SDL Models,” *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM ’94),* San Francisco (Ca.), November 1994, pp. 1857-1861.

[21] T. Ohta, Y. Harada, “Classification, Detection and Resolution of Service Interactions in Telecommunication Services,” in *Feature Interactions in Telecommunications Systems,* L. G. Bouma, H. Velthuijsen (Eds.), IOS Press, Amsterdam, 1994, pp. 60-72.

[22] B. Stepien, L. Logrippo, “Feature Interaction Detection using Backward Reasoning with LOTOS,” in *Protocol Specification, Testing and Verification XIV,* S. T. Vuong, S. T. Chanson (Eds.), Chapman & Hall, London 1994, pp. 71-96.

[23] B. Stepien, L. Logrippo, “Representing and Verifying Intentions in Telephony Features Using Abstract Data Types,” in *Feature Interactions in Telecommunications Systems, III,* K. E. Cheng, T. Ohta (Eds.), IOS Press, Amsterdam, 1995, pp. 141-155.

[24] P. Zave, “Feature Interactions and Formal Specifications in Telecommunications,” *IEEE Computer,* Vol. 26, No. 8, August 1993, pp. 14-18.