Identification of Call Scenarios with Potential Feature Interactions

Dirk O. Keck

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

Abstract. This paper presents a method to extract scenarios in which feature interactions may occur from the usually much larger set of topologically different scenarios involving two different service features or two instances of the same service feature. Simple conditions for the identifications of such scenarios are given. In three examples involving the service features Call Forwarding, Terminating Call Screening, and Consultation Call the application of the method is demonstrated. The scenarios identified by this method may serve as a starting point for further investigations, like, e.g., the generation of test cases or the application of formal verification techniques.

1 Introduction

1.1 The Intelligent Network (IN)

The architectural concept of the Intelligent Network has been developed in order to facilitate the rapid and cost efficient introduction of new services and service features into a communication network.

In conventional non-IN switching technology, integration of a new service feature into the existing system makes it necessary to update the software of the whole switching system. This comes along with a lot of problems, like the need to test the *whole* new software system before deployment into the network and the necessity to simultaneously update the software of many switches in order to keep the network software in a consistent state. The IN architecture reduces this problem by providing a standardised interface between the basic call processing functionality and the execution of the additional services.

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

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

editing facilities on the global functional plane (GFP) [6] level of abstraction. It should integrate development, verification, and validation of new service features and provide an interface to service management and service processing in the service control point (SCP) to generate executable service implementations.

1.2 The Feature Interaction Problem in IN

The abundance of new services, however, lets the well known phenomenon of the mutual influence of service features onto each other become a real problem, the *Feature Interaction Problem* [1, 3].

Each service feature enhances the basic capabilities of the telephone service ("Plain Old Telephone Service," POTS) by additional functions. When the activation of two or more service features has the effect of affecting the functionality of one of these features, so that its actual behaviour becomes inconsistent with its specified behaviour [4], this is called a *feature interaction.* Confusion of the subscribers, financial loss for the network operator or service provider, or even severe network problems may be among the consequences.

In order to ensure that two features do not show unintended interactions, it is necessary to identify and avoid these as early as possible in their life cycles. There are several approaches to tackle this problem, as it is a major obstacle to the achievement of the objectives of the IN architecture. Cameron and Velthuijsen [2] use a classification into three categories: the term *avoidance* subsumes system architectures and design methodologies that inherently reduce the risk of unwanted interactions. When interactions in existing specifications are identified by means of verification, validation and simulation, or by observing their behaviour in the network, the term *detection* is used. *Resolution* methods are employed to eliminate interactions in service specifications or implementations and to resolve interactions occuring during service operation.

In this contribution, simple criteria for the identification of call scenarios involving pairs of feature instances are presented. These criteria aim at the reduction of the huge number of topologically different scenarios. The results belong to the category of interaction detection. They can be used to support the establishment of validation and testing procedures for newly developed and existing service features.

Section 2 describes the basic structure of an IN call. The terms "participant" and "scenario" used in this contribution are informally defined and explained in section 3. In section 4, conditions for the existence of potential interactions are given and explained by several examples.

2 Structure of an IN Call

2.1 Basic Call State Models

On the distributed functional plane (DFP) [7] of the IN architectural concept, basic call processing is strictly divided into an originating and a terminating half, which are represented by an originating and terminating basic call state model (O- and T-BCSM)

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

respectively. A basic call state model is a communicating finite state machine, consisting of states *(points in call)* and transitions. Most transitions contain *detection points* as a means to report the occurrence of a specific transition, i.e., a specific event during an IN call, to the Service Control Function (SCF). A detection point has to be activated ("armed") before it can be used. It is possible to specify additional criteria for the activation of a detection point, like specific patterns in the dialled number, etc. A detection point can be used to trigger the invocation of a IN service (i.e., as a trigger detection point) or to communicate the occurrence of a specific event to an IN service instance (i.e., as an event detection point).

An IN call usually starts with the originating BCSM assigned to the originating party of the call. When at the originating side the destination information has been entered and routing information is complete, a terminating BCSM associated to the originating BCSM is created. This *association* of an O-BCSM and a T-BCSM usually represents the basic element of an IN call. Each basic call state model belongs to the same party during the whole call relationship, and it can neither change its owner nor its role (originating/terminating) during this call relationship.

It is possible that an IN participant simultaneously is assigned to more than one O-BCSM and/or T-BCSM. These BCSMs can be independent from each other, i.e., used alternatively like in a Consultation Call service feature, where the subscriber can select the party he wants to communicate with using the hook flash, or they can be coupled, like the combination of T- and O-BCSM in the forwarding party of the Call Forwarding features [9]. Examples of these BCSM associations are depicted in Figure 1.

2.2 Legs and Connection Points

The concept of *legs* and *connection points* is a means to express the connectivity of a call relationship on the distributed functional plane [7]. In CS-1, two types of legs exist, a *controlling leg,* which is the leg for which an IN service instance is invoked, and one or more *passive legs,* that represent the endpoints of communication with no IN service instances. Legs are uniquely identifiable in a CS-1 connection control switching state model (SSM). Legs can be connected by *connection points.* If all logical connections which can be created during an IN call relationship by the legs and connection points under the influence of a specific IN service instance are combined in a representation similar to the examples in Figure 1, an *IN service* or *service feature scenario* can be established, which can be used as a starting point for further studies on the possibility of feature interactions. This is discussed in detail in the following sections.

There exist service features where the originating and terminating roles cannot easily be assigned to the participants, like multi way calling, because so-called *Type B* capabilities, which are beyond Capability Set 1 are required [5]. In many cases, however, it is possible to circumvent this drawback for the method presented here by assigning O-/T-BCSM associations to all possible communication paths within such a scenario. In general, it remains for further study whether this approach is feasible in all situations or whether it has to be modified.

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

3 Characterisation of IN Services Features

3.1 IN Service Scenarios

In this section, some informal definitions of the IN service and call scenarios are given. These concepts are fundamental to the method presented in this document.

> DEFINITIONS
>
> *Formal Participant:* Each IN service contains a number of roles which are taken by end users in reality. These roles are represented by the formal participants of the IN services in this document.
>
> *Actual Participant:* Each actual participant represents one specific end user in reality. All formal participants of an IN service are assigned uniquely to actual participants. Note that more than one formal participant can be assigned to each actual participant.
>
> *Service (Feature) Scenario:* The set of formal participants of an IN service (feature), their basic call state models, and the associations between these models form a service (feature) scenario.
>
> *Call Scenario:* When actual participants are assigned to the formal participants of one or more service scenarios, a call scenario results.

In this paper, formal participants are denoted by uppercase letters. When service scenarios are instantiated, the formal participants are replaced by actual participants, which are denoted by integer numbers.

Caused by the limited amount of influence a CS-1 service feature has got on call processing (see section 3.2), the variety of possible service scenarios is rather small. The call structures created by CS-1 service features (listed in Q.1211 [5]), can be divided into three classes:

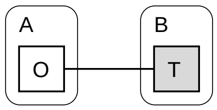
1. Simple two party structure consisting of a O-BCSM associated with a T-BCSM. This is the simplest structure for an IN call, it is depicted in Figure 1 (a). This kind of scenario is created, e.g., for simple Call Screening service features.

2. Linear structure with at least two T-BCSMs and two O-BCSMs, where the T-BCSMs and O-BCSMs in the middle are connected "back to back." This is a typical scenario for all kinds of Call Forwarding features (Figure 1 (b)).

3. Branching structures like multi way calling, call waiting, or consultation call. These structures are characterised by usually having more than two active participants. However, they often require capabilities which are beyond CS-1. An example for this kind of scenario is the Consultation Call scenario in Figure 1 (c).

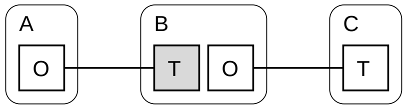
FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

(a) Simple association of O- and T-BCSM



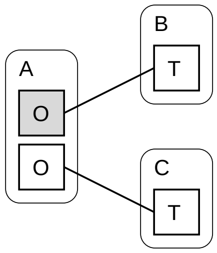
Example: *Terminating Call Screening*
A = Caller
B = Party with active TCS

(b) Service scenario with coupled T- and O-BCSMs



Example: *Call Forwarding*
A = Caller
B = Forwarding party
C = Forwarding destination

(c) Service scenario with coupled T- and O-BCSMs



Example: *Consultation Call*
A = Caller (using Consultation Call)
B = Called party
C = Party called with Consultation Call

Figure 1: Three types of service scenarios with formal participants. The locations of the initial detection points of the example service features are indicated by the shaded BCSMs.

3.2 Influence of CS-1 Service Features on Call Processing

As mentioned previously, an IN service is invoked when in its corresponding BCSM an armed trigger detection point, a so-called *Initial Detection Point* is encountered and additional criteria, like specific pre- or postfixes in the dialled number are met. Service processing takes place in the *Service Control Point (SCP)* and is initiated by an INAP message "InitialDP" [8] sent from the *Service Switching Point (SSP)* to the SCP. During service processing, other detection points, called *event detection points* indicating the occurrence of specific events can be armed by the service logic and thus give the possibility to react and influence call processing.

The possibilities of a CS-1 service to influence basic call processing (and thus to manipulate the topology of a call relationship) are limited, as all services are single ended and have a single point of control. Such services are called *Type A* services in Q.1211 [5].

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

All services that do not fulfil the Type A criteria are placed in category Type B. These services are beyond the scope of Capability Set 1. The following list gives a brief outline of the means a CS-1 service feature has got to influence basic call processing. Establishment and release of connections to specialised resources are not taken into account in this list, as they just interrupt call processing. This does not imply, however, that these connections cannot be responsible for feature interactions, but only, that the progress of call processing is not affected. Basically, there are the following possibilities represented by a *Point of Return (POR)* on the Global Functional Plane:

- **Continue with existing data.** This POR resumes call processing where it has been interrupted by the detection point. It indicates, that the call continues with no changes, neither call topology nor call processing are influenced.
- **Proceed with new data.** This POR is used, when data has been changed (like in Freephone service) or if the call has to be forwarded. It indicates the creation of new legs and influence on call topology. Call processing may also be influenced in some situations.
- **Clear Call.** The call is torn down for all participating parties. Obviously, this indicates an influence on call processing.
- **Initiate Call.** A call is initiated by the SCF, like, e.g., wake up service. This POR indicates the creation of new legs and influence on call topology and call processing.

4 Identification of Potentially Interacting Scenarios

4.1 Conditions for Interactions

In this section, we assume that interactions occur between two instances of different service features or between two instances of the same service feature. The method presented here, however, can easily be adapted to identify potentially interacting scenarios with three or more feature instances, but this will result into an exploding number of different call scenarios to be considered.

When the two service scenarios with formal participants are instantiated with actual parties, some conditions have to be fulfilled to have the possibility for feature interactions in these instances:

> CONDITION 1
>
> *The two service scenario instances must have at least one common actual participant.* It is necessary, that the set of participants of each of the two features must not be disjoint, i.e., at least one actor of one service scenario is at the same time an actor of the second one.

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

This condition is obvious, as there cannot be any interaction if the scenarios of the two features do not "get in touch" with each other. Even if the two instances of the service scenarios have got one common participant, this does not necessarily imply that both features can be activated during a call relationship following this scenario. This observation directly leads to

> CONDITION 2
> *At least one of the both features must be able to trigger the execution of the second one.* There must exist at least one actual participant that (a) is a participant of both features (this is condition 1) and (b) contains the BCSM that triggers one of the features and (c) also contains this BCSM as a member of the other feature.

A less obvious condition is presented next. It can be established due to the fact that BCSMs do exist only in pairs of originating BCSMs associated with terminating ones in IN service scenarios.

> CONDITION 3
> *One pair of associated BCSMs containing an initial trigger detection point of one feature must be common to both feature scenarios.* It is only necessary to study interaction scenarios where the association of the O- and T-BCSM containing the initial trigger detection point of one feature is also contained in the second feature scenario. If this is not the case, i.e., only condition 2 is fulfilled, the feature is triggered, but there exists another scenario that covers this case and fulfils condition 3.

It has to be noted, that the three conditions are not independent from each other. Condition 3 implies condition 2, which in turn implies condition 1.

The application of these three conditions is illustrated using the example of two instances of the call forwarding service feature. In Figure 2, three call scenarios with this service feature are displayed. The parties that belong to each of the service feature instances are enclosed by dashed polygons. These are drawn to highlight the IN service scenarios for the Call Forwarding feature. Each of the call scenarios consists of two instances of the service scenario, where the formal participants have been replaced by actual participants. The location of the initial trigger detection points are denoted by the shaded basic call state model instances. The associations of originating and terminating BCSMs on which the feature instances can be triggered are drawn as bold arrows. In this figure, part (a) shows a call scenario which fulfils condition 1, but not conditions 2 and 3. Although party 1 is common to both scenarios, the features cannot interact, because party

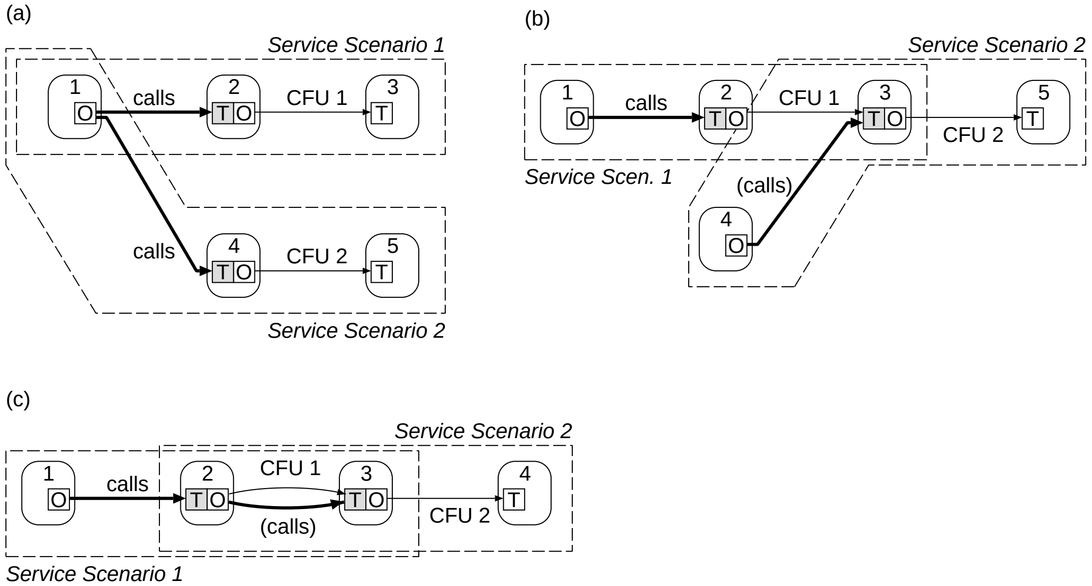


Figure 2: Example of two instances of “call forwarding” service features. These call scenarios illustrate the application of the three interaction conditions

1 either can call party 2 and thus trigger the feature “CFU 1” or it calls party 4 and triggers “CFU 2.”

The call scenario in part (b) of this figure fulfils condition 2, but not condition 3. Here, both features can only be triggered when party 1 calls party 2, the first one by the trigger of party 2, the second one on the forwarded part of this call by the trigger of party 3. In this case, however, party 4 is not involved in the call relationship. Part (c) displays a call scenario that fulfils condition 3. This implies conditions 1 and 2. When party 1 calls party 2, the call is forwarded to 3, and on this first forwarding section, the second call forwarding feature is triggered.

4.2 Establishment of a List of Potentially Interacting Scenarios

As a starting point, let there be two service scenarios S_I and S_{II} with n_I and n_{II} formal participants respectively. Of course, it is possible that S_I and S_{II} denote the same service scenario. First, actual participants have to be assigned to the formal participants of the scenarios. There are $n = n_I + n_{II}$ formal participants in these scenarios, which leads to n^n different possibilities to assign n actual participants. However, only topologically different assignments are relevant, i.e., assignments which can be created by a simple re-numbering of the actual participants are ignored. The number of topologically independent scenarios can be calculated using the following recursive formula:

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

n	$f(n)$
1	1
2	2
3	5
4	15
5	52
6	203
7	877
8	4,140

n	$f(n)$
9	21,147
10	115,975
11	678,570
12	4,213,597
13	27,644,437
14	190,899,322
15	1,382,958,545
16	10,480,142,147

Table 1: Numbers of topologically different scenarios with n participants (Equation 1).

$$f(n) = \begin{cases} \sum_{i=0}^{n-1} \binom{n-1}{i} f(n-1-i) & \text{for} \quad n \geq 1 \\ 1 & \text{for} \quad n = 0 \end{cases} \tag{1}$$

Table 1 gives an impression of the numbers obtained by this formula.

Using the conditions 1 to 3 listed previously, the number of relevant scenarios out of the list of topologically different ones can be reduced significantly. This is illustrated by the examples in section 4.3.

Sometimes it is possible to apply additional constraints to the obtained list of scenarios in order to reduce the number of significant ones, like, e.g.

- prohibiting activation of two features at the same actual participant,
- requiring the disjointness of two or more formal participants, i.e., it is not possible to assign the same actual participant to them,
- exploiting symmetry when two instances of the same feature are under investigation.

4.3 Examples

4.3.1 Two Instances of the Call Forwarding Service Feature

In this example, call scenarios consisting of two instances of a call forwarding service scenario (as shown in Figure 1 (a)) are established. Totally 6 formal participants are involved, 3 for each instance of the call forwarding feature. This results in a number of 203 topologically different scenarios (see Table 1). The application of condition 3 and the fact that only one of two symmetric scenarios have to be used, leads to a reduction to 24 scenarios. When, as an additional constraint, call forwarding directly to the forwarding party itself is prohibited, only 13 scenarios remain. It would not make sense to allow activation of two instances of call forwarding at the same actual participant, as this could

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

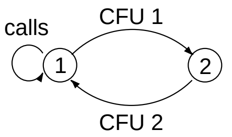
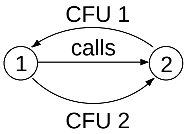
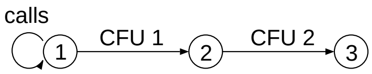
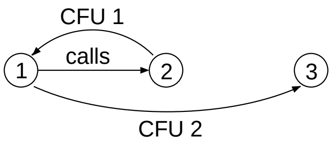
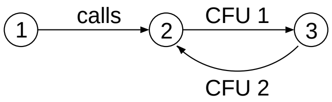
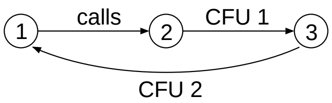
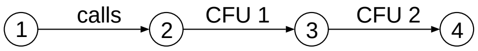
	Party 1 calls itself. Party 1 has activated CFU to party 2. Party 2 has activated CFU to party 1.
	Party 1 calls party 2. Party 2 has activated CFU to party 1. Party 1 has activated CFU to party 2.
	Party 1 calls itself. Party 1 has activated CFU to party 2. Party 2 has activated CFU to party 3.
	Party 1 calls party 2. Party 2 has activated CFU to party 1. Party 1 has activated CFU to party 3.
	Party 1 calls party 2. Party 2 has activated CFU to party 3. Party 3 has activated CFU to party 2.
	Party 1 calls party 2. Party 2 has activated CFU to party 3. Party 3 has activated CFU to party 1.
	Party 1 calls party 2. Party 2 has activated CFU to party 3. Party 3 has activated CFU to party 4.

Table 2: Call Forwarding Unconditional (CFU) with itself

result into a direct contradiction. Thus, exclusion of these scenarios removes another 6 possibilities. The remaining 7 relevant scenarios are shown in Table 2. In Figure 2, examples for the application of the three conditions are shown.

4.3.2 Terminating Call Screening and Call Forwarding

Terminating Call Screening (TCS) has got two formal participants, the *caller* and the *callee*, represented by an association of an O- and a T-BCSM. The feature is triggered at the terminating side. Its service scenario can be found in Figure 1 (a). Call Forwarding Unconditional (CFU) has got three actors, as depicted in Figure 1 (b). The number of 52 topologically different configurations of the 5 participants of these two features can be reduced to 9 potentially interacting scenarios using condition 3.

Like in the previous example, additional conditions can be used to reduce the size of this list further. If it is not allowed to forward calls to yourself and if it is ensured that

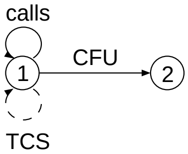
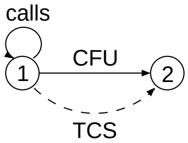
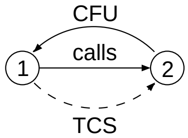
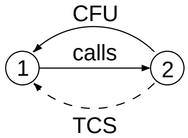
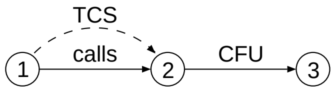
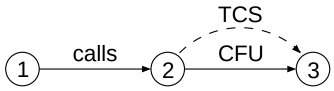
	Party 1 calls itself. Party 1 has activated TCS and it has activated CFU to party 2.
	Party 1 calls itself. It has activated CFU to party 2. Party 2 has activated TCS.
	Party 1 calls party 2. Party 2 has activated TCS Party 2 also has activated CFU to party 1.
	Party 1 calls party 2. Party 2 has activated CFU to party 1. Party 1 has activated TCS.
	Party 1 calls party 2. Party 2 has activated TCS Party 2 also has activated CFU to party 3
	Party 1 calls party 2. Party 2 has activated CFU to party 3. Party 3 has activated TCS.

Table 3: Terminating Call Screening (TCS) and Call Forwarding Unconditional (CFU)

the forwarding destination entered at the time the service is customised by its user is not the user's own number, only 6 potentially interacting scenarios remain (Table 3). In these 6 cases, however, the different possible settings for the screening list are not regarded. It remains for further study, how this can be achieved and integrated in this method. In Table 3, the dashed TCS arrows denote the section in the call scenarios on which Terminating Call Screening is activated (triggered), *not* the screened party.

4.3.3 Call Forwarding and Consultation Call

In this example, the conditions established in section 4.1 are applied to a call forwarding service feature scenario in combination with a consultation call[1] scenario (see Figure 1

[1] The Consultation Call service feature allows a subscriber to place a call on hold, in order to initiate a new call for consultation [5].

(a) *The call forwarding feature can be triggered by the consultation call feature and vice versa*

CF Caller	CF Forwarding Party	CF Forwarding Destination	CC Caller	CC Called Party	CC Party Called with Cons. Call
1	1	2	1	1	1
					2
					3
				2	1
1	2	1	1	2	1
					2
					3
1	2	3	1	2	1
					2
					3
					4

(b) *Consultation call is triggered on the second (forwarding) branch of call forwarding*

CF Caller	CF Forwarding Party	CF Forwarding Destination	CC Caller	CC Called Party	CC Party Called with Cons. Call
1	1	2	1	2	2
					3
1	2	1	2	1	1
					2
					3
1	2	3	2	3	1
					2
					3
					4

(c) *Call forwarding is triggered by one of the consultation call branches*

CF Caller	CF Forwarding Party	CF Forwarding Destination	CC Caller	CC Called Party	CC Party Called with Cons. Call
1	1	2	1	3	1
1	2	1	1	1	2
				3	2
1	2	3	1	1	2
				3	2
				4	2

Table 4: Call Forwarding Unconditional (CFU) and Consultation Call (CC)

(b) and (c)). Using condition 3, from the 203 topologically different scenarios created by the 6 formal participant of these features, 39 scenarios can be extracted. If, like in the previous examples, call forwarding to the forwarding party itself is prohibited, 26 relevant scenarios remain. These scenarios are listed in Table 4, divided into three disjoint groups:

- In the first group (Table 4 (a)), scenarios are listed, in which the features can be mutually triggered, i.e., the call forwarding feature can be triggered on a branch of the consultation call feature as well as the consultation call feature can be triggered on a section of the call forwarding scenario.

- The second group (Table 4 (b)) displays scenarios where the consultation call feature can be triggered by the call forwarding feature on its forwarding section. It must be noted that on the first sight, this kind of scenario do not seem to make much sense. A closer look, however, leads to the insight, that there may evolve interactions also in

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

these cases, caused by the wrong interpretation of a signal from the call forwarding participants or by detection points reached during the progress of call forwarding.

- The scenarios in the third group (Table 4 (c)) allow triggering of the call forwarding by on of the consultation call branches, but not in the opposite direction.

4.4 Limitations of this Method

The method presented here makes it possible to identify a set of scenarios for two IN service features, which is significantly smaller than the number of possible topologically different scenarios. However, the principal problem of an extremely growing number of potential interactions if the number of services increase, still remains unsolved. Whether it can be alleviated by the creation of "service classes" containing services that follow the same principal pattern like, e.g., the class of number translating services (with the pattern "Dial – Initial detection point – Database lookup for new number – Continue with new data"), remains for further study.

5 Conclusions

The method presented in this contribution offers a simple means to extract significant interaction scenarios out of the large number of possible topologically different service scenarios that can be composed out of the participants of two feature instances. It is possible to reduce the number of relevant scenarios considerably. The service scenarios obtained by this method may serve as a starting point for the establishment of a set of test cases as well as a base for further investigations using, e.g., formal verification techniques.

Acknowledgements: The research presented here is supported by the Software Labor of the University of Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart, Germany. It is carried out in cooperation with Alcatel SEL AG, D-70435 Stuttgart. I wish to thank Dr. Stephan Rupp, Rodolfo López Aladros and Gerd Siegmund for interesting discussions. Further, I am indebted to Prof. Paul J. Kühn for his continuing encouragement and to Reiner Sailer and Michael Schopp for their valuable comments.

Abbreviations and Acronyms

BCSM	Basic Call State Model
CC	Consultation Call
CF	Call Forwarding
CFU	Call Forwarding Unconditional
CS-1	Capability Set 1
IN	Intelligent Network

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)

INAP	Intelligent Network Application Protocol
ITU-T	International Telecommunication Union
O-BCSM	Originating Basic Call State Model
POR	Point of Return
POTS	Plain Old Telephone Service
SCE	Service Creation Environment
SCF	Service Control Function
SCP	Service Control Point
SIB	Service Independent Building Block
SSM	Switching State Model
SSP	Service Switching Point
T-BCSM	Terminating Basic Call State Model
TCS	Terminating Call Screening

References

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

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

[3] N. D. Griffeth, Y.-J. Lin, "Extending Telecommunications Systems: The Feature-Interaction Problem," *IEEE Computer,* Vol. 26, No. 8, August 1993, pp. 14-18.

[4] E. J. Cameron, N. Griffeth, Y.-J. Lin, M. E. Nilson, W. K. Schnure, H. Velthuijsen, "A Feature-Interaction Benchmark for IN and Beyond," *IEEE Communications Magazine,* Vol. 31, No. 3, March 1993, pp. 64-69.

[5] ITU-T Recommendation Q.1211 "Introduction to Intelligent Network Capability Set 1," Geneva, March 1993.

[6] ITU-T Recommendation Q.1213 "Global Functional Plane for Intelligent Network CS-1," Geneva, March 1993.

[7] ITU-T Recommendation Q.1214 "Distributed Functional Plane for Intelligent Network CS-1," Geneva, March 1993.

[8] ITU-T Recommendation Q.1218 "Interface Recommendation for Intelligent Network CS-1," Geneva, March 1993.

[9] ITU-T Recommendation Q.1219 "Intelligent Network User's Guide for Capability Set 1," Geneva, April 1994.

FINAL VERSION printed September 16, 1996
presented at the International AIN Workshop
March 25-26, 1996, Passau (Germany)