K. Kimbler and C. Wohlin, "A Statistical Approach to Feature Interaction", Proceedings Telecommunications Information Networking Architecture (TINA´95), pp. 219-230, Melbourne, Australia, 1995.

A Statistical Approach to Feature Interaction

Kristofer Kimbler and Claes Wohlin
Department of Communication Systems, Lund University
Box 118, 221 00 Lund, Sweden
Phone: +46 46 109008
Facsimile: +46 46 145823
e-mail: chris@tts.lth.se

Keywords: feature interaction, service modelling, statistical usage testing, reliability certification

Abstract: This paper presents the concept of a new, statistical approach to the feature interaction problem. The approach is based on the ideas originating from Statistical Usage Testing, and aims at rationalizing the process of feature interaction detection during service creation by utilizing the statistical properties of system usage. The most probable combinations of service features should receive the most attention, whereas the combinations whose occurrence probabilities are extremely low or zero may not be considered. In this sense, the proposed statistical approach is an added value to other feature interaction detection methods. By taking advantage of the statistical distributions of the user's behaviour, it is also possible to provide for efficient verification of service software and certification of service quality from the feature interaction point of view.

The paper discusses general aspects of the feature interaction problem. The principles of statistical testing and statistical usage modelling of services are outlined. The process of creating the usage specifications of telecom services is presented, and possible applications of the usage specification in feature interaction detection, service software verification, and service quality certification are discussed.

1. Introduction

Every communication network is the subject of continuous evolution. Unfortunately, this process rarely has a conservative nature [15]. The operation of newly introduced services and features alters, often intentionally, the behaviour of the existing ones [9]. This phenomenon is known as Feature Interaction (FI). The range of problems it may cause varies from an unexpected or undesired service behaviour slightly annoying to the users, to large financial losses, or serious network breakdowns.

The FI problem is commonly regarded as one of the major obstacles to the rapid deployment of new services for Intelligent Networks (IN). The inability to assure consistency between the expected and the actual service behaviour might jeopardize this fundamental goal of IN. The problem is not unique to IN, and can be encountered in other technical and non-technical domains. In telecommunications it has been encountered since the first PBXs with a diversity of features were introduced, over 20 years ago.

The FI problem can be addressed in different phases of the service life cycle and on different levels of abstraction. Research in this area focuses on detection, resolution, and avoidance of FIs. Despite a number of significant efforts like [4], no widely approved solutions and no common taxonomy exists in this area, so far (e.g. since *service* and *feature* are often used as synonyms, the term *service interaction* can also be found). In this paper, by service we mean a commercial package, or rather combination, of features that is offered by the service provider. Thus we can talk about intra-service and inter-service feature interactions as well as about interactions between services.

There are tendencies to address the FI problem already in the early stages of service creation, by applying formal modelling of services in SDL, LOTOS, Z [5, 1, 14], or different types of logics [7, 6, 15], combined with automatic or manual verification of their properties. Other efforts concentrate on finding network mechanisms, such as the Negotiating Agent Model [10]. The proponents of the latter approach stress that run-time FI resolution provides flexibility and simplifies service creation, and fits perfectly the idea of rapid service deployment in a multi-provider environment. Also, new concepts of service and network architectures, such as TINA are being investigated [3, 21]. Their aim is to separate services from the underlying switching capabilities by using, for instance, the concept of Open Distributed Processing (ODP). As claimed, this should increase the orthogonality of different features, and thus eliminate, to a great extent, their interactions.

The two latter approaches are the most promising in the long run. However, the need for the detection and resolution of FIs at the early stages of the service life cycle will still exist in the future. Different features, for example, Unlisted Number and Calling Line Presentation, may be introduced to realize contradictory goals, independently of their implementation. Possible combinations of such features and the nature of their interactions have to be analysed during service creation in order to provide appropriate resolution mechanisms or to take appropriate policy decisions. If such policy decisions can lead to restrictions, the service provider should make the user aware of them.

The detractors of FI detection and resolution during service creation contend that this task is a virtually impossible, since the number of feature combinations to be analysed grows exponentially with the number of features in a network [2]. From the combinatorial point of view this is true, but, in reality, many feature combinations need not be considered [12], because they can either never occur or their occurrence probability is extremely low. This is the fundamental observation that leads us to the idea of a statistical approach to feature interaction.

2. A Statistical Approach to Feature Interaction

Growing customer expectations and deregulation of the telecom market in many countries is causing an increased pressure to rapidly deploy new, technologically advanced telecom services. In order to meet this challenge, service development cycles will have to be radically shorter than today, while the product quality will have to remain very high. In particular, service providers and network operators will have to find time- and cost-efficient methods for detecting and resolving feature interactions which will enable them to shorten the time-to-market for new services, and to minimize the negative influence of the feature interactions on the quality (availability and reliability) of the services provided.

This paper presents a new, statistical approach to the feature interaction problem, based on the ideas originating from **Statistical Usage Testing** (SUT). The main objective is to utilize the statistical properties of the user's behaviour in order to rationalize FI detection and verification at different stages of service creation, and to enable certification of the service quality from the feature interaction point of view. Before we present this new approach, an outline of SUT will be given.

2.1 Principles of Statistical Usage Testing

The idea of SUT originates from Cleanroom software engineering [16]. In general, statistical testing is a specific type of functional black-box testing in which test data are selected from the input domain according to probability distributions. The basic idea of SUT is to match the actual usage profile during the testing process in order to detect the system failures that affect the users most frequently, i.e. those that have a significant impact on the system reliability. SUT promotes early system exposure to realistic usage, which brings objectivity and realism to the testing process and helps to achieve significant reliability improvement during the process. A similar statistical testing method called Operational Profile Testing (OPT) has been adopted by AT&T in the field of telecom switching systems [17].

The famous study by Adams, quoted in [16], shows that just 1.5% of the faults existing in the program code cause over 50% of all the reported execution failures, whereas 60% of the faults occur so rarely that they cause only 3% of the failures. These figures show that equal treatment of all software faults, as is done in conventional testing, is not the best way of improving system reliability during the testing process. Using non-statistical sampling we can only assess system reliability for a particular, arbitrarily selected, set of test cases.

It should be stressed, that by system reliability we mean here a measure of how well the system provides the services expected by its users. The system reliability is thus not an absolute value. Instead, it is very much dependent on how the system services are employed by the users in practice. Therefore, the ability to provide a realistic **usage specification** is the key factor in the process of assessing the system reliability. The usage specification consists of the usage model and usage profile. The **usage model** describes the structural aspects of the usage (e.g. possible sequences of service and feature invocations generated by the user), whereas the **usage profile** quantifies this usage in terms of the probability distribution of user's choices that lead to the execution of specific services and features in specific environmental context.

Whittaker in [22] proposes the discrete state Markov chain as a stochastic model of system usage. A usage specification based on a Markov chain is very detailed, i.e. it models every single move of the user and defines its probability. Though the occurrence probabilities of system services, features, and their combinations are not defined explicitly, they can be derived from the usage specification due to the analytical (mathematical) properties of Markov chains. Unfortunately, for large, multi-access systems, such as IN, this approach may lead to a so-called state explosion which results in the usage model becoming so big and complex that it might become unmanageable. The problem can be solved by developing a hierarchy of Markov chains (see Section 3).

2.2 Properties of Statistical Usage Testing

Statistical testing is a relatively new approach which is just beginning to be used on the industrial scale. There are, however, some interesting applications of SUT and OPT. The results of these applications show that statistical testing requires up to 50% more effort during test planning, but the total time spent on test preparation and generation can be reduced to 70% compared with non-statistical testing methods, since an unlimited number of tests cases can be automatically generated from the usage specification. The overall time and cost of system testing may be reduced by 50% [7]. Statistical testing can be as much as 20 times more cost-effective in finding execution failures than conventional approaches, and the number of user-reported faults can be reduced by a factor of 10 [17]. Since the problems caused by FIs are just a subset of all possible network failures, we could expect that the application of SUT in this area will have similar, positive effects.

Despite these encouraging results, sceptics claim that statistical testing is not able to assure a sufficient requirements coverage and thus several critical operations may not be verified during the testing process. The objectivity and realism of test selection in statistical testing is the strongest argument against this thesis. Practical results show that statistical methods provide a good requirements coverage which usually reaches as high as 90% [16], which is not worse than for traditional, coverage-oriented functional testing methods. Moreover, a properly constructed usage profile may emphasize rarely used, but critical operations [17].

The real problem with SUT is that the predicted system reliability relies very much on the defined usage profile of a system. Wrongly defined usage profiles might result in an irrelevant reliability assessment. Creation of a relevant and realistic usage specification for newly provided services and features may present a great challenge, requiring extensive market research, prototyping, etc. This is especially difficult when new types of services are provided, for which no realistic usage data exist. Musa [17] describes a systematic, top-down approach to the creation of usage (operational) profiles for telecom systems.

2.3 Utilization of Usage Specification

As far as the FI problem is concerned, the usage specification of services can be utilized in three ways:

- The usage specification can be used to calculate occurrence probabilities of different combinations of features. By focusing on the most probable combinations, and "neglecting" those whose probability is under a certain level or zero, we can dramatically reduce the time and cost of FI detection and resolution during service creation. By applying this policy, we will also minimize the negative impact of FIs on the users, and shorten time-to-market for new services.
- The usage specification can be employed to generate test case scenarios resembling the real user's behaviour. Such test cases can then be applied for verifying the (executable) specifications and models of the system as well as for testing of the system itself. By measuring the mean time between observed FIs (analogous to Mean Time Between Failures), we can certify the system reliability from the FI point of view.
- If some of the detected FI cases are intentionally left in the network, for instance, for the sake of flexibility, or if their resolution imposes certain restrictions, the service provider must inform the users about it. It is unacceptable to supply user's manuals with hundreds of pages describing in detail all such cases. By employing the usage specification it is possible to choose the FI and restriction cases that will be encountered most frequently and include only them in the user's manuals.

By implementing the statistical approach to FI in practice, the idea of rapid service deployment could be facilitated, and the negative effects of FIs for the service subscribers and providers minimized. Moreover, we would be able to quantify the actual impact of the FI problems on the service reliability.

3. Usage Specification of Telecom Services

3.1 Usage modelling of services

Statistical testing is based on a usage specification which consists of a usage model and a usage profile. As mentioned in Section 2.1, discrete Markov chains can be applied for service usage modelling, but on a limited scale. As a matter of fact, the concept of statistical testing is so new that no well-established theory of creating usage specifications exists so far. It should be stressed, however, that usage modelling of telecom services, though without statistical aspects, has already been successfully applied to interaction detection [13]. Below, we discuss two service-oriented usage modelling techniques proposed in [20, 24] and [13] which may be useful in the statistical approach to FI.

The first of the usage modelling techniques introduces the **State Hierarchy** (SHY) model which extends the concept of discrete Markov chains. The SHY model has been designed to support the modelling of real-time, multi-access telecom systems. The SHY model specifies the usage in a hierarchical manner with services and their features as the lowest levels of this hierarchy. Such a structure of the usage model facilitates incorporation of new services and features into the usage specification, and reduces the negative effect of the state explosion typical for "flat" Markov chains. Though the SHY model is designed to support automatic test-case generation, it still preserves the nice analytical properties of Markov chains. In particular, it enables calculation of service occurrence probabilities.

The SHY model assumes that there are different types of users. A distribution of user types, i.e. a percentage of users in each type, should reflect reality. This distribution is modelled on the user type level, whereas particular users are represented on the user level. Each user type has a number of services available. These are represented on the service level. The actual behaviour of the users of particular services and features is described on the behaviour and sub-behaviour levels. These levels contain finite state machines which are Markov chains. The SHY model also introduces the notion of time in the usage model.

The SHY concept is still evolving, and there are a number of problems to be solved. In particular, it has been designed with test case generation in mind, which requires explicit specification of the dependencies between different services and users. Such dependencies are described by means of so-called *links* between different finite state machines on the behaviour and sub-behaviour levels. This means that certain interactions between services have to be explicitly specified in the model. This issue must be further investigated if the model is to be used for solving the feature interaction problem.

The second of the mentioned usage modelling techniques does not have the drawback mentioned above. The second method is based on the concept of a **Service Usage Model** (SUM) developed for the purpose of detection of feature interactions in Intelligent Networks [13]. The SUM enables determination of which combinations (sequences) of service and feature invocations are possible in call scenarios. The model shows many similarities to the SHY model discussed above; however, it concentrates purely on service usage modelling corresponding to the behavioural level of the SHY model. In particular, service features are treated as atomic operations, rather than as separate finite-state automata like in the SHY model. It is worth noticing that the SUM-based technique has been successfully applied to the detection of interactions between European-wide IN services in EURESCOM Project P230 [12].

A nice property of the Service Usage Model is that each service is described separately by means of Service Usage Graphs (SUGs). A SUG shows explicitly all possible orders of feature invocations within the service it represents. Possible orders of invocations of different services and features in calls are also shown by means of so-called *common states*. This means that, unlike SHY, no explicit specification of the dynamic relations between different services is necessary. The SUM, as it stands, does not include any statistical profile. Such extension is, though, possible.

The concepts of SHY and SUM are complementary, i.e. both have some disadvantages, but a proper combination of their elements may give a usage specification which has both a manageable and simple structure, and good analytical properties. The proposal for such a usage specification is outlined in the next section. This is, however, a subject for further investigation.

3.2 Creation of Usage Specifications

The process of creating a usage specification requires a systematic approach, for example, a usage specification cannot be produced by means of ad hoc methods. A proposal for systematic creation of a usage specification is shown in Figure 1. The creation process starts with the Use Case Driven Analysis of service requirements, as described in [11] and [18]. Different call scenarios (use cases) are identified for typical system users (actors). Use cases are described as sequences of events that take place in call scenarios. The use cases are then analysed and merged into a number of Service Usage Models, as proposed in [13]. Each SUM is related to one actor (user type). In of some service cases, the Service Usage Graphs (see previous section) may have an identical structures for different actors. It is also possible that SUGs representing the same service have different shapes for different actors, e.g. due to differences in features offered for private and business users.

In parallel to Use Case Driven Analysis, the Statistical Usage Analysis of services is performed. Here a structured approach, like the one proposed in [17] for operational profiles, should be applied. As a result of this analysis, probability distributions of users' behaviour, environmental conditions, and service invocations should be obtained. We will not elaborate on this subject here, as a great deal of inventory work remains to be done. The statistical analysis is only possible if the service providers and network operators systematically collect information about service usage. Note that, unlike POTS and PSTN, no commonly accepted mathematical models of IN or GSM services exist today. This is equally relevant for modelling of system usage as for modelling of the traffic related to these new services and networks.

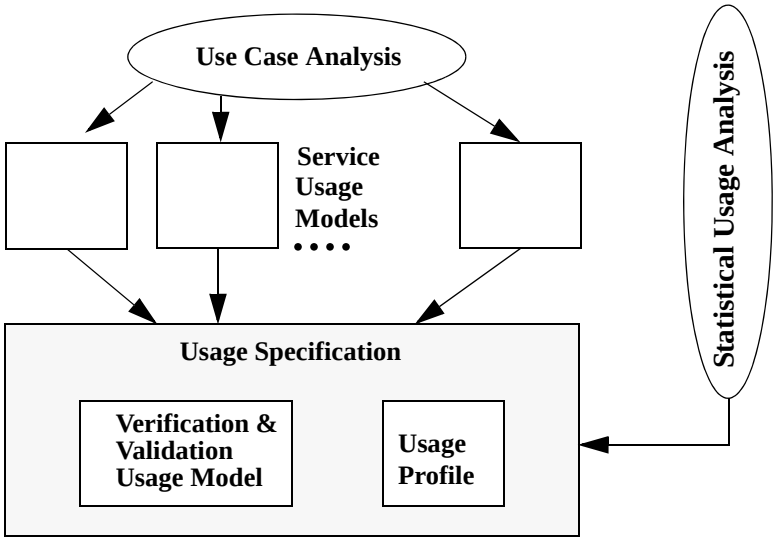

FIGURE 1. Creation of Usage Specification.

The SUMs form the basis for creating a hierarchical **V&V Usage Model**, according to the ideas behind the SHY model. In this sense, the SUMs form the behavioural level of the SHY model. The V&V Usage Model is then complemented with the **Usage Profile** yielded by the Statistical Usage Analysis, and together they form the **Usage Specification**. The Usage Profile captures statistical usage properties of every service used by each of the actors. Note that the same service may have different usage profiles for different actors (users types). For instance, specific features of the Credit Card Calling (CCC) service may be used with different intensity by private and business users. Hence, the SUG describing CCC may appear in the SUMs related to these two user types, but with different usage profiles.

4. A Statistical Approach to Feature Interaction Detection

The detection of FIs during service creation requires specification of the properties of the services, their features, and the network as such. As mentioned in the introduction, different formal description techniques can be applied here. If the properties of one feature are violated by the properties of another, or if they are not satisfied by a composition of their formal models (if such are provided), an interaction case is spotted [6, 7]. The verification of feature properties is still done manually for most of the models. This is a lack of balance between expressive and analytical power typical for the formal models in computer science. Many interesting attempts are being made to apply techniques such as exhaustive simulation or model checking to enable automatic verification of service and feature properties, and thus automatic detection of FIs [5]. None of the documented attempts of this kind have, however, proved to be useful for a large number of services and features.

Since most existing methods of formal service modelling take the user's point of view [6, 7], we believe that the usage specification outlined in the previous section can be a step towards the creation of more detailed, formal models of services in SDL, LOTOS, or any other formal language or modelling technique. In this way, consistency and traceability between the usage model and the formal model of services could be assured. Of course, the ultimate goal for the proposed statistical approach is to develop a formal service model that could also cover statistical properties of service usage. In Figure 2, a possible role of the usage specification and the formal service model in FI detection are shown.

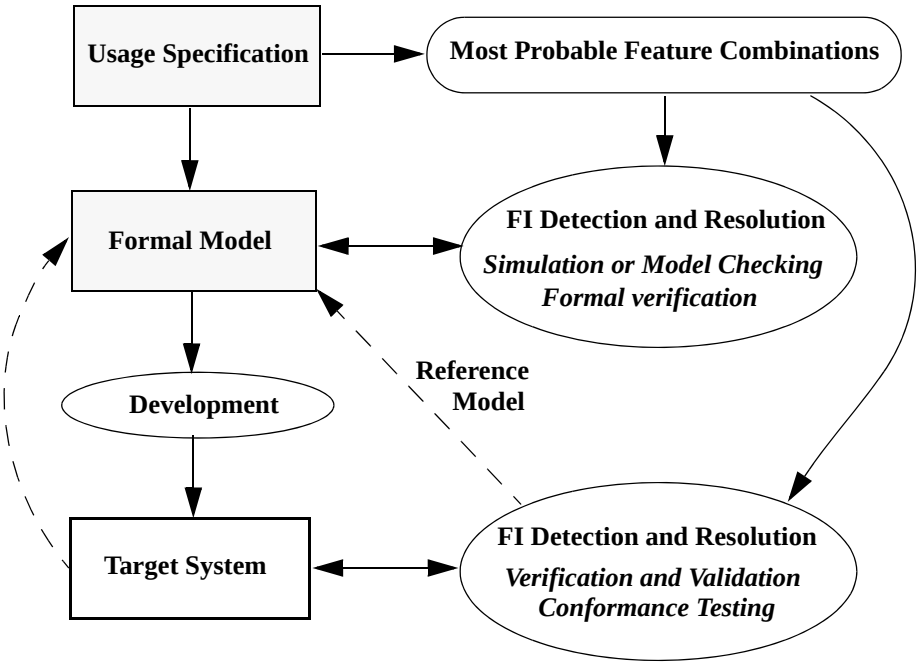

FIGURE 2. Role of Usage Specification in FI detection and resolution.

The proposed statistical approach gives an added value to formal FI detection techniques. By applying the usage specifications to select the most probable feature combinations and call contexts in which their properties have to be verified, the number of combinations to be checked either manually or by some validator will be significantly reduced.

In consequence, the time necessary for FI detection will become fairly reasonable, so that it can be performed every time a new service or feature is added to the network. Moreover, by resolving first the most probable FIs, the overall impact of the FI problem on the users will be minimized already at the early stages of service creation and in a relatively short time. Features with very low occurrence probabilities, but with critical impact on the network performance or billing system, may receive a special treatment, as suggested in [17].

Even if we manage to detect and resolve all the most probable interaction cases existing in the formal model, we still have to verify the consistency between the real service software and the model, and to detect all FIs that have been introduced during the design and implementation of this software. Unfortunately, exhaustive simulation or model checking are not applicable to the target systems and probably never will be. The only means we can use for this purpose is testing. Testing; but not the exhaustive system testing which is commonly used for detecting service and feature interactions today.

Since coverage of all feature combinations under all possible environmental contexts is virtually impossible, we can again apply the Usage Specifications to select the most probable feature combinations, environmental conditions, and call scenarios, as shown in Figure 2. By constraining our verification to such cases, we would be able to carry out the system and conformance testing in a reasonable time. The formal models of the services could be used as a reference model for automatic verification of the test results, which should speed up the testing process even more.

5. Feature Interaction Certification Process

One of the most important ideas of our method is the FI-oriented certification of system reliability during service creation. The FI certification process, which is described subsequently, could become an integral part of the total system reliability certification.

The certification of system reliability is based on all failure occurrences and time between these failures, but, as the objective here is to certify FIs, only failures related to them should be considered. The time between such failures can be put into the same type of mathematical models of reliability as normal failure data (see Section 5.2). Hence, it will be possible to certify the level of network and service reliability with regard to feature interactions.

5.1 Certification process

Randomized generation of test cases, representative of the actual system usage, from the Usage Specifications, is crucial for FI certification. Note that, with this approach there is no need for any extra, FI-oriented testing. Instead, it is necessary to be able to distinguish FI-related failures or problems from the others. The formal system model, serving as a reference model of system behaviour, can be used for this purpose, see Figure 3.

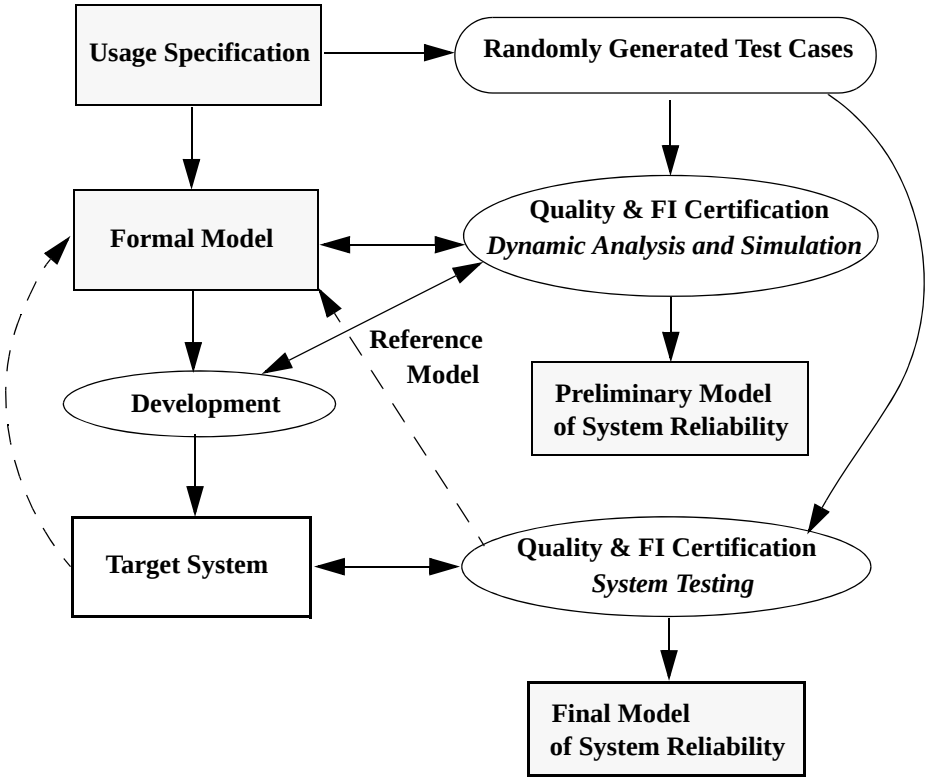

FIGURE 3. Feature Interaction Certification Process.

As mentioned above, the objective of the proposed method is also to allow for this type of certification throughout the whole service creation process. Certification at the later stages by means of system testing is most obvious and is extensively covered in the literature, but some work has also been done to enable reliability certification prior to system testing. In [23], the possibility of certifying the reliability during dynamic analysis of system design models in SDL is discussed, and in [25] some results concerning simulation of SDL specifications are presented. These early-stage methods may also be applied to FI. In this way, the scale of the FI problem and its impact on the service users and subscribers can be estimated very early by the service providers and network operators.

The process is hence based on a usage specification and either a formal model of the system or the target system itself, see Figure 3. Test cases are generated randomly according to the expected Usage Specification and these can be applied for FI detection by means of dynamic analysis [23] or simulation [25] early in the software life cycle. The results from the analysis or simulation must be compared with the requirements specification or the formal models of the system created during development, to determine whether the system is behaving as expected or not. This early certification gives the first estimate of the reliability of the system (or at least its software). The actual mathematical certification procedure is briefly described in the next section.

As the development proceeds and the software becomes available on the target system, then the system testing can be initiated. Test cases are once again generated according to the usage specification. The reliability can now be estimated and the future system reliability can be predicted. One part of the reliability estimation is the FI reliability, i.e., failures or unexpected behaviour caused by interactions can be measured. Again, the correct behaviour can be determined through a comparison with the formal models or the system requirements.

5.2 Certification procedure

In the book by Musa *et al*. [18] a model for reliability demonstration testing is described. The model is a form of hypothesis certification, which determines if a specific MTBF (Mean Time Between Failures) requirement is met with a given degree of confidence or not. A hypothesis is proposed and the testing is aimed at providing a basis for acceptance or rejection of the hypothesis. The procedure is based on the use of a correct operational profile during testing and faults not being corrected.

The primary objective here, however, is to certify that the FI requirement is fulfilled at the end of the certification, hence resolution of the FI problems during the certification process ought to be allowed. A relaxation of the assumption in the model of no changes is discussed below. The hypothesis certification model is based on an adaptation of a sampling technique used for acceptance or rejection of products in general.

The hypothesis is that the time between FIs is greater than a predetermined requirement. The hypothesis is rejected if the objective is not met with the required confidence and accepted if it is. If the hypothesis is neither accepted nor rejected, the certification must continue until the required confidence in the decision is achieved.

The hypothesis certification is performed by plotting the FI failure data in a control chart. Figure 4 shows failure number (r) against normalized failure time (t_{norm}). The failure time is normalized by dividing the failure time by the required time between FIs.

The certification continues while the measured points fall in the *Continue* region. The certification is terminated when the measured points fall into the *Reject* or *Accept* region, and the software is then rejected or accepted accordingly. In Figure 4, a rejection case is shown.

The control chart is constructed by drawing the acceptance and rejection lines. They are based on the accepted risks taken for acceptance of a bad product and rejection of a good product. The calculations are described by Musa *et al*. [18].

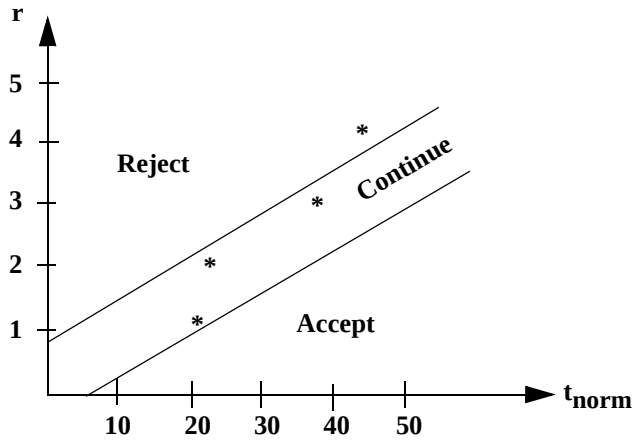

FIGURE 4. Control chart for hypothesis certification of the FI reliability.

When solutions to interaction problems are introduced, the control chart should be reset and the process started from the beginning. It is not practical to reset the control chart after every identified unwanted interaction, so the chart is reset after a number of cases has occurred. The reason for resetting the chart is mainly that after solving a problem, the software can be viewed as a new product.

It can be concluded that the hypothesis certification model is easy to understand and use. The model provides support for decisions of acceptance or rejection of software products at specified levels of confidence. The hypothesis certification model may also be complemented with one or more software reliability growth models to allow for prediction of reliability. These types of models may also be used to predict FI reliability in particular.

6. Conclusions

The idea of a statistical approach to the feature interaction problem presented in this paper is new and still untried. However, the most important elements of this approach have already been practically verified with positive results. To summarize, we can say that the presented approach promotes the following three ideas:

- Statistical usage modelling of telecom services and features,
- Statistical approach to the rationalization of feature interaction detection and service verification,
- Certification of network and service reliability from the feature interaction point of view.

The first idea still requires a great deal of research and case studies before a usage specification that satisfies all the needs of the statistical approach to feature interaction can be achieved. Other formalisms, such as Petri Nets or Harel's State Charts, should be investigated as potential service models. An ultimate model should capture procedural, non-procedural, and statistical usage properties of the services and their features, and it should also demonstrate a proper balance between the expressive and analytical power, i.e. automatic calculation of service occurrence probabilities, automatic generation of test cases, and automatic detection of feature interactions should be possible.

The second idea, as we see it, could become one of the most radical solutions for the feature interaction problem. The statistical approach is complementary to other methods aimed at feature interaction detection during service creation. By utilizing statistical properties of the service usage, we are able to dramatically reduce the number of service and feature combinations to be verified and feature interaction cases to be solved. The biggest obstacle here is the lack of proper statistical data concerning the usage

of new, advanced services. Therefore, to implement this approach in practice, the process of systematic collection of such data has to be started by service providers and network operators as soon as possible. Our job, as researchers, will be to develop a relevant model capturing this statistical data, and to provide methods for creation and analysis of this model.

The third idea addresses the problem of certifying the network reliability from the feature interaction point of view. In fact, we should go beyond this and discuss the reliability of the services and features in separation from the underlying network. Feature interaction certification is then just part of this concept. So far, no requirements on service software quality have been formulated or standardized. In the future, third-party service provider market, the need for such standards as well as for efficient procedures for certifying the reliability of provided services, will be apparent. We believe that our proposal of feature interaction-oriented certification will not only help to handle the feature interaction problem during service creation and to quantify the impact of feature interactions on the users, but will also contribute to the concept of the standardisation in the area of telecom service quality.

The final conclusion is that the proposed statistical approach can be a significant step towards minimizing the time-to-market and maximizing the quality of provided telecom services. If consistently implemented in the service creation process, it can give the service providers a competitive edge in the growing market of advanced telecom services.

References

[1] R. Boumezbeur, L. Logrippo, *Specifying Telephone Systems in LOTOS and the Feature Interaction Problem*. In International Workshop on Feature Interactions in Telecommunication Software Systems, December 1992.

[2] T. F. Bowen, C. Chow, F. S. Dworak, N. Griffeth, G. E. Herman, Y. Lin., *The Feature Interaction Problem in Telecommunication Systems*. In Proceedings of the Seventh International Conference on Software Engineering for Telecommunication Switching Systems, July 1989.

[3] M. Chapman, P. Farley, R. Minerva, A. Oshisanwo, ROSA - *A Service Architecture for TINA.* In Proceedings of TINA'93, 1993.

[4] E. J. Cameron, N. Griffeth, Y. Lin, M. E. Nilson, W. K. Shnure, H.Velthuijsen. *A Feature Interaction Benchmark for IN and Beyond*. In IEEE Communication Magazine, March 1993.

[5] P. Combes, M. Michel, B. Bernard. *Formal Verification of Telecommunication Service Interactions using SDL Methods and Tools.* In SDL'93: Using Objects, Elsevier Publishers, 1993.

[6] P. Combes, S. Pickin, *Formalisation of a User View of Network and Services for Feature Interaction Detection*. In Feature Interactions in Telecommunications Systems, IOS Press, 1994.

[7] M. Dyer, The Cleanroom Approach to Quality Software Development. John Wiley & Sons Inc., 1992.

[8] A. Gammelgaard, J. E. Kristensen. *Interaction Detection, a Logical Approach.* In Feature Interactions in Telecommunications Systems, IOS Press, 1994.

[9] N. Griffeth, Y. Lin, *Extending Telecommunications Systems: The Feature Interactions Problem*. In IEEE Computer, August 1993.

[10] N. Griffeth, H. Velthuijsen, *The Negotiating Agent Model for Rapid Feature Development*. In Proceedings of the Eight International Conference on Software Engineering for Telecommunication Systems and Services, March 1992.

[11] I. Jacobson et al., *Object-Oriented Software Engineering, A Use Case Driven Approach*, Addison-Wesley, 1992.

[12] K. Kimbler, E. Kuisch and J. Muller, *Feature Interactions among Pan-European Services.* In Feature Interactions in Telecommunications Systems, IOS Press, 1994.

[13] K. Kimbler, and D. Söbirk, *Use Case Driven Analysis of Feature Interactions.* In Feature Interactions in Telecommunications Systems, IOS Press, 1994.

[14] A. Lee, *Formal Specification - a Key to Service Interaction Analysis*. In Proceedings of the Eight International Conference on Software Engineering for Telecommunication Systems and Services, March 1992.

[15] C. A. Middelburg, *A Simple Language for Expressing Properties of Telecommunication Services and Features.* In FORTE'94, October 1994.

[16] H. D. Mills, M. Dyer, R. C. Linger, *Cleanroom Software Engineering*. IEEE Software, pp. 19-24, September 1987.

[17] J. D. Musa, *Operational Profiles in Software Reliability Engineering*, IEEE Software, pp. 14-32, March 1993.

[18] J. D. Musa, A. Iannino, and K. Okumoto, *Software Reliability: Measurement, Prediction, Application.* McGraw-Hill, New York, 1987, pp. 201–203.

[19] B. Regnell, K. Kimbler, A. Wesslen, *Improving the Use Case Driven Approach to Requirements Engineering. Proceedings of 2nd IEEE International Symposium on Requirements Engineering*, York, UK, March 1995.

[20] P. Runeson, C. Wohlin, *Usage Modelling: The Basis for Statistical Quality Control*. Proceedings 10th Annual Software Reliability Symposium, pp. 77-84, Denver, Colorado, USA, 1992.

[21] J.B. Stefani, Y. Lepetit, The SERENITE Long Term IN *Architecture.* In Proceedings of TINA'93, 1993.

[22] J. Whittaker and A. Poore, Markov Analysis of Software Specification. ACM Transactions on Software Engineering and Methodology, Vol. 2, Jan. 1993.

[23] C. Wohlin, P. Runeson, *A Method Proposal for Early Software Reliability Estimations*, Proceedings 3rd International Symposium on Software Reliability Engineering, pp. 156-163, Raleigh, North Carolina, USA, 1992.

[24] C. Wohlin and P. Runeson, *Certification of Software Components*. IEEE Transactions on Software Engineering, Vol. 20, No. 6, pp. 494-499, 1994.

[25] C. Wohlin, *Evaluation of Software Quality Attributes during Software Design*. Informatica, Vol. 18, No. 1, pp. 55-70, 1994.