Tesi di Dottorato:

Analysis and Modeling of Dependability and
Performability of Telecommunication Systems

CANDIDATO: Stefano Porcarelli

RELATORI: Prof. Simoncini Luca

Prof. Bondavalli Andrea

Giugno 2003
Abstract

Our life relies on telecommunication systems which are more and more pervasive and ubiquitous; in general, people need to rely upon them. Failures of such devices can get people just disappointed sometimes but other times catastrophic consequences may happen. Thus, dependability and performance evaluation are a valuable mean to help operating companies and systems constructors to reduce the risk of customer’s complaints and loss of business to competitors as well the risk of unnecessary investments. This work provides a modeling framework for evaluating dependability and performance (also performability) measures of some aspects of telecom systems. Our studies concentrate on wireless systems, and, particularly, on the random access procedure to GPRS (General Packet Radio Service) at one side and data maintenance of databases supporting telecom systems at the other side. A contribution of our approach is in the explicit separation between the architectural, and user (service) concerns of a system. The overall models, based on different Petri-like formalisms, consists of i) an architectural model describing the behavior of system hardware and software components, ii) a user-level model and, optionally, iii) a maintenance policy model. The achieved results show the behavior of the target systems in normal conditions, in presence of faults (in term of dependability and Quality of Service (QoS) related measures) and possible solutions to alleviate their effects has been suggested and explored. Concerning to GPRS random access procedure, different aspects has been analyzed such as the behavior of the system during the transient period following its outage–repair, both the user and system QoS perspective in normal and degraded situations as the presence of faults, and queueing and threshold mechanisms to alleviate uplink congestions during the random access procedure. On the other side, the database maintenance studies dealt with a definition of a methodology to identify an appropriate scheduling of database audits. These audits cope with run time data corruptions and they differ for coverage (of faults) capability and costs. Their scheduling (frequency, order, and targeted fields of the database) posed challenging problems in term of performability optimization. Finally, our studies can be useful for an accurate dimensioning of the target systems which takes into account the effects of faults and/or congestion phenomena.
Acknowledgments

The research presented throughout this thesis has been conducted at the Dependability Computing Lab (DCL) of ISTI institute of the Italian National Research Council (CNR) of Pisa and has been partially supported by Motorola Italia of Turin.

Many people in different ways helped me to complete this work. First, I owe a particular thanks to Dr. Felicita Di Giandomenico of DCL for her essential helps and collaborations during the whole period of this doctoral research. Also, I would like to thanks all the other members of the group, Andrea, Daniela V., Daniela T. which sustained me and, explicitly, Silvano, for his repeated and patient assistance.

A great part of my work has been carried out in collaboration with other people which I sincerely want to thank. They are also co-authors of the publications yielded during these years: Francesco Tataranni (who carried out his master thesis at CNR), Amine Chohra (on leave at CNR during his ERCIM fellowship), Ivan Mura and Massimo Barbera of Motorola Italia, and Marco Castaldi of University of L’Aquila.

Finally, many thanks to whoever gave me a moral support and encouragement. Particularly, I want to thank the colleagues who enriched my working life or were helpful in some way: Luisa and Raffaele. Moreover, a person outside the work that I would like to thank for her support is Valeria. Last but certainly not least, I want to thank my family.
1 Dependability Concepts
   1.1 Introduction ................................................. 9
   1.2 Faults and Their Manifestations .............................. 9
   1.3 Basic Concepts and Definitions ............................... 12
   1.4 The Means for Dependability ................................ 15
       Fault Prevention (Avoidance) ............................... 15
       Fault Tolerance ............................................. 15
       Fault Removal .............................................. 16
       Fault Forecasting ......................................... 16
   1.5 Models for Dependability and Performability Evaluation ... 17
   1.6 Combinatorial Models ....................................... 19
       1.6.1 Reliability Block Diagrams ............................ 20
       1.6.2 Fault–Tree ............................................. 20
       1.6.3 Reliability Graphs .................................... 22
   1.7 State–Based Models ......................................... 22
       1.7.1 Queueing Models ...................................... 23
       1.7.2 Markov Chains ......................................... 23
       1.7.3 Automated Generation of Markov Chains ............... 25
1.7.4 Place Transition Petri Nets ........................................... 26
1.7.5 Stochastic Petri Nets .................................................. 27
1.7.6 Generalized Stochastic Petri Nets (GSPNs) ..................... 28
1.7.7 Stochastic Reward Nets (SRNs) ................................. 28
1.7.8 Deterministic and Stochastic Petri Nets (DSPNs) ......... 29
1.7.9 Stochastic Activity Networks (SANs) ............................ 29
    Activation and Reactivation Functions ............................ 31
    Reward and Impulse Measurements ............................... 31
1.8 Model Power-Hierarchy ................................................. 32
1.9 Simulation Techniques .................................................. 32
    1.9.1 Importance Sampling Techniques ............................ 34
1.10 Conclusions ............................................................. 34

2 Analysis of Telecommunication Systems 35
    2.1 Introduction .......................................................... 35
    2.2 The Needs for Dependability ..................................... 36
    2.3 General Packet Radio Service ................................... 37
        2.3.1 A System Point of View .................................. 37
        2.3.2 The Protocol Architecture ................................ 39
        2.3.3 GPRS Air Interface Protocol .............................. 42
        2.3.4 Data Flow .................................................... 44
        2.3.5 The Random Access Procedure ........................... 46
    2.4 Database Maintenance .............................................. 46
    2.5 Conclusions .......................................................... 47

Part 2 49

3 GPRS Analysis without Outages 51
    3.1 Introduction .......................................................... 51
    3.2 Analyzing QoS at Network Level ................................ 52
        3.2.1 Measures of Interest ....................................... 52
        3.2.2 Modeling Assumptions .................................... 52
        3.2.3 The SAN Model .............................................. 53
        3.2.4 Settings for Numerical Evaluation ....................... 56
        3.2.5 Numerical Evaluation ...................................... 56
            A First Scenario ............................................ 57
            A Second Scenario ......................................... 60
    3.3 Analyzing QoS of GPRS from a User’s Perspective .......... 61
        3.3.1 Measures of Interest ....................................... 63
        3.3.2 Modeling Assumptions .................................... 63
        3.3.3 Overall Model .............................................. 64
7.5 Conclusions ............................................... 144

Conclusions .................................................. 147

Bibliography .................................................. 151

A Further Modeling Details ................................. 159
  A.1 Further Details of the SAN Model of Figure 5.6 .... 159
  A.2 Model Validation in the Database Maintenance Problem .... 161
List of Tables

1.1 Downtime per year for several availability figures . . . . . . . . 14
2.1 GPRS coding schemes . . . . . . . . . . . . . . . . . . . . . . . 42
2.2 GPRS logical channel . . . . . . . . . . . . . . . . . . . . . . . 45
3.1 Relevant parameters and their default values . . . . . . . . . . . 56
3.2 User parameters . . . . . . . . . . . . . . . . . . . . . . . . . . 68
3.3 Detailed model parameters . . . . . . . . . . . . . . . . . . . . 69
4.1 Relevant parameters and their values (Scenario 1) . . . . . . . . 83
4.2 Relevant parameters and their values (Scenario 2) . . . . . . . . 83
4.3 Relevant parameters and their values . . . . . . . . . . . . . . . 88
5.1 Activity distributions for the BTS model . . . . . . . . . . . . . 95
5.2 Activity distribution for the Gb link model . . . . . . . . . . . 96
5.3 Output gate definition for the GbLINK subnet . . . . . . . . . . 97
5.4 Activity distributions for SAN model of SGSN . . . . . . . . . 98
5.5 Gate definitions for the SGSN subnet . . . . . . . . . . . . . . 99
5.6 Settings for numerical evaluation . . . . . . . . . . . . . . . . . 100
5.7 User parameters in Scenario 1 . . . . . . . . . . . . . . . . . . 114
5.8 User parameters in Scenario 2 . . . . . . . . . . . . . . . . . . 114
7.1 Relevant parameters and their default values . . . . . . . . . . . 134
## List of Figures

1.1 The failure classes ........................................ 11
1.2 Error chain .................................................. 11
1.3 Error propagation ............................................ 12
1.4 Power hierarchy among dependability model types ........ 33

2.1 GPRS Architecture ........................................... 38
2.2 A simple example of routing ................................ 39
2.3 GPRS Transmission Plane .................................... 40
2.4 GPRS radio block structure .................................. 42
2.5 GPRS coding procedure ...................................... 42
2.6 Packet transformation flow ................................... 44

3.1 SAN model of the Random Access Procedure of the GPRS system ........................................ 54
3.2 Plots of $P_{\text{block}}$ in scenario 1 ......................... 57
3.3 Plots of $C_{\text{busy}}$ in scenario 1 .......................... 58
3.4 Plots of $P_{\text{block}}$ in scenario 1 with SPDCH = 1 (a), and SPDCH = 3 (b) ......................... 59
3.5 Plots of $C_{\text{busy}}$ in scenario 1 with SPDCH = 1 (a), and SPDCH = 3 (b) ......................... 59
3.6 Plots of $D$ in scenario 1 with PRACH = 1 (a), and PRACH = 2 (b) ................................. 60
3.7 Plots of $P_{\text{block}}$ in scenario 2 with SPDCH = 1 (a), and SPDCH = 2 (b) ......................... 61
3.8 Plots of $C_{\text{busy}}$ in scenario 2 ............................ 62
3.9 Plots of $D$ in scenario 2 with PRACH = 1 (a), and PRACH = 2 (b) ................................. 62
3.10 Composed model ............................................. 64
3.11 Modified random access model ............................ 66
3.12 User model .................................................. 66
3.13 Delay distribution for users belonging to $Group_1$ .... 71
3.14 Delay distribution for users belonging to $Group_2$ .... 71
3.15 Timeout probability ............................................. 72
3.16 Mean service delay ............................................. 73
3.17 Threshold’s effect on the mass probability function of the serv-
     ice time ......................................................... 74
3.18 Failure Probability and Delay at varying of $P_{thr}$ under 90% load 74
3.19 Failure Probability and Delay at varying of $T_{thr}$ under 90% load 75
3.20 Failure Probability and Delay at varying of $P_{thr}$ under 110%
     load .............................................................. 76
3.21 Failure Probability and Delay at varying of $T_{thr}$ under 90% load 76
4.1 Subnet which models the outage effects ............................. 81
4.2 Modified random access model .................................. 81
4.3 Behavior during outages and recovery (Scenario 1, 150 users) . 85
4.4 Behavior during outages and recovery (Scenario 1, 75 users) . 85
4.5 Behavior during outages and recovery (Scenario 2, 150 users) . 86
4.6 Behavior during outages and recovery (Scenario 2, 75 users) . 86
4.7 Time to steady state (recovery time) vs. outage duration .... 87
4.8 Effects of outages with/without queueing mechanism ......... 89
4.9 Effects of the queue on the recovery time at varying of outage
duration ............................................................ 89
5.1 GPRS architecture under analysis ................................ 93
5.2 Redundant links between BSC and SGSN ......................... 94
5.3 SAN model of the BTS .......................................... 95
5.4 SAN model of the redundant Gb link .......................... 96
5.5 SAN Model of SGSN ............................................ 97
5.6 Composed model for the network availability model .......... 99
5.7 Outage distribution .............................................. 101
5.8 Availability at varying of $K_F$ and $K_R$ ...................... 102
5.9 Network availability submodel composed with the random ac-
     cess submodel ................................................ 103
5.10 New details of the random access model of Figure 3.1 ....... 103
5.11 Modified random access model ................................ 104
5.12 Percentage reduction of served users per hour with and with-
     out considering the transient effects ......................... 106
5.13 Availability reduction with and without considering the tran-
     sprint effects .................................................. 106
5.14 Number of served users per hour at varying of the MTTR . 107
5.15 Served users per hour as a function of MTTR for $inter_{request\_time}$
     = 60 sec ...................................................... 108
Introduction

Software and hardware systems, computer architectures and networks have become pervasive in all fields of human activity. Their performance and reliability is now crucial to the proper functioning of all areas of industry, commerce, the arts and entertainment. Thus, dependability and performance evaluation of computer architectures and computer communication networks has become a key issue in information technology and computer science. Many challenges of system performance and dependability are being constantly revisited while new problems and concerns are coming to the forefront of this area.

Historically, the distinction between performance and dependability has been very useful in the development of probabilistic modeling techniques suited to each purpose. However, individual evaluations of system performance and dependability may fail to provide an adequate assessment of system’s ability to perform. In fact, system performance can be degradable in the sense that consequences of faults can reduce the quality of the delivered service even though that service, according to its specification, remains satisfactory. Such degradation may result directly from fault–caused errors, may be due to additional computational demands associated with error processing, or may be due to additional consequences of subsequent fault–related actions such as reconfiguration and repair. If performance is degradable due to both performance events (causing performance state change) and dependability events (causing changes in the structure of the system), more refined assessment of a system’s ability to serve can be obtained via measures of performability.

In telecommunication systems the most impacting dependability requirements, both on user satisfaction and operators revenue, is the availability of the service. In fact, the system can tolerate short outage periods, but must be available as long as possible. From the previous considerations descends that analysis of a telecommunication system from the pure performance viewpoint tends to be optimistic since it ignores the failure/repair behavior in the system. On the other hand, pure availability analysis tends to be too con-
servative since performance considerations are not taken into account. Also, in a well-designed system, the failure of a component node will cause partial outage of the system, specifically, the decrease of system capacity available to the users, which further affects the performance and Quality of Service (QoS) to the users. Therefore, in real systems, availability, and performance are important QoS indices which should be studied in a composite manner. The combined evaluation of the indices described above is useful when the system under study can operate in a gracefully degradable manner in the presence of component failures [1].

Research Framework

The main goal of this work is to investigate and to propose a methodology for assessing dependability and performability of telecommunication systems. As said before, availability (and most generally dependability) is an important aspect of telecommunication systems and it is tightly linked to the concept of Quality of Service (QoS). The availability and reliability requirements constitute an important factor in the competition among service providers, network operators and equipment manufacturers.

Wireless systems are particularly vulnerable to failures and malfunctions for several reasons. Firstly, wireless architectures are naturally distributed over a wide geographic area. Secondly, equipment-pricing pressure and fast technology progresses prohibit massive fault-tolerance similar to that in the slowly evolving wireline business. Thirdly, the responsibility for ensuring dependable functioning of the system is generally distributed among several independent collaborating entities (wireless equipment owners and leased lines providers). Fourthly, the access medium of wireless is much more vulnerable than the access medium of wireline (wire to the home).

In order to alleviate these problems, it is necessary to introduce a quantitative approach for dependable system design in all phase of the design process. This approach is based on modeling and analysis of the system architecture and includes the following steps:

1. Identify potential architectural deficiencies in relation with customer requirements.

2. Build a statistical model for the architecture. The model is driven by outages due to the following components: hardware, software, distributed links, and system update.

3. Evaluate availability, and, if appropriate, its tradeoffs with performance and requirements.
INTRODUCTION

This work provides a modeling framework for evaluating dependability and performance (also performability) measures of some aspects of telecommunication systems. Throughout this dissertation, a modeling approach has been pursued based on different Petri-like formalisms. A contribution of our approach is in the explicit separation between the architectural, and user (service) concerns of a system. Thus, the overall models are composed of i) an architectural model describing the behavior of system hardware and software components, ii) a user-level model and, optionally, iii) a maintenance policy model. This modeling approach expresses the “separation of concerns” property. It has been very recently better formalized also in [2].

With this in mind, this research has been focused on wireless communication systems, mainly on GPRS (General Packet Radio Service) at one side and maintenance of databases supporting such systems at the other side.

GPRS has been developed to enhance the Global System Mobile Communication (GSM) system with the introduction of services based on a packet switching transmission technique. These services provide a more efficient use of the radio resources, by accommodating data sources that are bursty in nature, at lower costs for subscribers. Typical examples of applications producing bursty traffic are Internet applications, e.g. World Wide Web, FTP, Telnet and e-mails. Work on GPRS started in 1994, and a standardization of the GPRS specification has been recently frozen by ETSI (European Telecommunications Standard Institute). Both GSM and GPRS rely on databases for user mobility management and billing purpose. The Visitor Location Register (VLR) is a database storing actual user related information of the users currently served in a given area and the Home Location Register (HLR) holds further user information, like the actual location and the subscription data of the users.

Thus, two important specific points of a packet wireless communication system (as GPRS) concerning dependability and performability issues are addressed in this work:

1. A combined analysis and evaluation of dependability and performance of particular aspects of the GPRS network, more closely to the random access procedure to the network.

2. The definition and scheduling of fault-tolerance, and maintenance strategies and for database supporting wireless systems.

**Topic 1.** With respect to the first point, the aims has been to highlight possible “bottleneck” of the system and suggest a methodology to save the scarce radio resources. The random access procedure to the network by
mobile users (uplink connection) has been identified as critical because of collision of users in getting bandwidth resource to start a connection. A modeling approach has been afforded to analyze the congestion phenomenon that arise during the contention among a number of users trying to access the network. Starting from a quite classical idea of QoS analysis different scenario’s have been analyzed considering congestion phenomenon both during properly working and outage periods for the system. The peculiarity our analysis is the dependability point of view this study is framed in.

The study has been developed toward two main directions:

- From a system QoS to real perception of the QoS by users.
- Analyze the effects of availability critical events (like outages) on the QoS indicators.

Analysis of the GPRS expected behavior have been performed, essentially focusing on measures like throughput, delay for the end-to-end frame transmission, average number of attempts necessary to win contention on the random access channel (e.g., [3], [4], [5], [6]). Particular care has been reserved at the user modeling. In fact, the performance as perceived by users has been evaluated. Different user classes have been defined to evaluate the impact at user level of the uplink connection performance. The resulting modeling approach allows bridging the gap between the classical network perspective that is commonly taken when studying the availability of telecommunications systems, and a user and application centric analysis of the dependability of services that can be provided through the packet data service of GPRS.

As it is for most telecommunication systems, and particularly for its “always connected” vision, GPRS classifies as an availability-critical system and a significant effort is thus being devoted by systems manufacturers to improve availability. Availability is defined as the property of “readiness of usage” [7], measured as the delivery of correct service with respect to alternation of correct–incorrect service. However, in the case of communication systems, like GPRS, whose services are continuously required by users, the mere estimation of availability in terms of intervals of times the system is operational with respect to those in which the system is halted is not a satisfactory measure to know. In fact, it can be easily observed that, during the period of stoppage (outage), users requesting services accumulate, waiting to have their requests accepted as soon as the system is up again. At system restart, the GPRS must face an overload; the high number of requests leads to a higher probability of collisions to get access to system resources, with a negative impact on the offered quality of service (QoS). The system requires
some time to get over the congestion before getting back to the “normal” behavior, in which the nominal QoS is provided to users. Finally, this work contributes to the analysis of GPRS by providing a modeling approach suitable for investigating on the effects of outage periods on the service provision, with special attention on the user perception of the QoS.

**Topic 2.** With regards to the second research area the focus is on the databases that support telecommunication systems. In fact, databases will, already and in the next future, have an important role in telecommunication systems. The databases contain persistent and temporary information needed in operations and management of telecommunication networks and services. Such databases exhibit stringent reliability, availability and performance requirements. To match these requirements and ensure the consistency of database subsystems involved in communication systems (e.g., cellular systems), appropriate scheduled maintenance policies are necessary. Audit operations, consisting in periodic checks and recovery actions, are typically employed in databases to cope with run time faults which may affect the dependability and quality of service of the overall system. This work aims at investigating on appropriate tuning of audit operations, so as to find optimal balances between contrasting requirements, namely satisfactory database availability and low overhead due to audits. For this purpose, a methodology to analyze the behavior of the database under scheduled maintenance is suggested. Analytical models, essentially based on Deterministic and Stochastic Petri Nets (DSPN), are defined and analyzed, in terms of dependability indicators.

Moreover, the short–persistence of most of the data stored in the database and the highly dynamic evolution of the environmental conditions, which characterize such target systems, pose relevant issues in devising efficient maintenance policies. Aiming at deriving optimal maintenance strategies, a learning approach is presented to dynamically adapt the maintenance policy at varying database and environmental parameter values leading to select, in each time period, the optimal maintenance policy. Some efforts has been done to define a dependability manager able to schedule the appropriate maintenance operations in front of dynamic change in the environment.

Case studies are presented and evaluated in terms of dependability indicators to show the effectiveness of the proposed approaches.
Thesis Organization

This dissertation is structured in two parts. The first part, which encompasses Chapter 1 and 2, describes the research framework while the remainder is the original contribution of this work.

In more detail, Chapter 1 introduces the main concepts and terminology of dependability. An overview of the methodologies used in dependability and performability analysis is given. Chapter 2 introduces the target systems, focusing on the relevant aspects of the GPRS and of database maintenance in wireless systems. Some motivations of the studies undertaken follow and firstly the aspects of the GPRS system that are relevant for the sake of the modeling and analysis conducted in the subsequent sections are introduced. We focus on the description of the Radio Access Network, and we treat in specific detail the slotted ALOHA random access procedure that plays a major role in the period immediately after an outage. Secondly, the framework study related to maintenance of databases in telecommunications is introduced and motivated.

Chapter 3 describes the model of the GPRS random access procedure and of a detailed user model; a performance analysis follows. Chapter 4 analyzes the GPRS with the objective of understanding its behavior under critical conditions, as determined by periods of outages, which significantly impact on the resulting dependability. Chapter 5 contains the models and the results of their evaluation campaign, aiming at quantitatively estimating the potential effects of outages on GPRS service dependability and at performing a sensitivity analysis to the most relevant parameters. The analysis has been carried out combining both a network and user perspective. The overall network structure of the GPRS has also been considered, which allows evaluating the availability of the whole system accounting of outages phenomena.

With regard to database maintenance topic, Chapter 6 introduce the overall framework of the audit modeling approach. We focus on pointer failures, which may yield to service unavailability, and a methodology and the measures to compare audit policies have been proposed. Actually, audit policies can differ for the audit scheduling, depending on the audit type, cost and effectiveness. Finally, Chapter 7 shows the results achieved and investigates a way to implement an intelligent agent capable to automatically select an optimal database maintenance strategy.

Conclusions close this dissertation.
Part 1
Chapter 1

Dependability Concepts

1.1 Introduction

Our life relies on computing systems which are more and more pervasive and ubiquitous; in general, people need to rely upon the services provided by them. Depending on the context, the focus will be on different properties of these services, e.g. the average real-time response and the reliance can be placed on the service accomplishment. Failures of such systems can get people just disappointed sometimes but other times catastrophic consequences may happen. The notion of dependability provides a very convenient means of subsuming these various concerns within a single conceptual framework.

In this Chapter we summarize the fundamental concepts of dependability. A structured view [7] of dependability follows, according to a) the threats, i.e. faults, errors, and failure, b) the attributes, and c) the means for dependability, that are fault prevention, fault tolerance, fault removal and fault forecasting. After, the focus moves on fault forecasting and some foundations of techniques for dependability and performability evaluation are detailed. Particularly, a list of means and their properties to perform fault forecasting follows. Particular emphasis will be deserved to the kind of Petri Nets that have been used for the statistical evaluation throughout this dissertation.

1.2 Faults and Their Manifestations

Systems fail for many reasons, including hardware failure, incorrect design of hardware or software, improper operation or maintenance, and unstable environments. The probability of error is distributed over this entire spectrum with non single cause dominating [8].
The following terms are used:

- **Fault**: Is an adjudged or hypothesized cause of an error. Faults can be classified on the basis of their duration and causes. Based on their duration, faults can be classified on: permanent, intermittent or transient. Permanent fault describes a fault that is continuous and stable; in hardware, permanent fault describes an irreversible physical change (the word hard is used interchangeably with permanent). Intermittent fault is a fault that is only occasionally present due to unstable hardware or varying hardware or software state. Transient fault describes a fault resulting from temporary environmental conditions (the word soft is used interchangeably with transient). Based on their origin, faults can be also classified as physical fault or human fault [8]. Physical faults stem from physical phenomena internal to the system, such as threshold changes, shorts, opens, etc., or external changes, such as environmental, electromagnetic, vibration, etc. Human faults may be either design faults, which are committed during system design, modification, or establishment of operating procedures, or they may be interaction faults, which are violations of operating or maintenance procedures. Based on their status, faults can also be classified in active, when it produces an error, otherwise it is dormant.

- **Error**: Is the manifestation of a fault within a program or data structure; errors can occur some distance from the fault sites. An error is a part of the system state that may cause a subsequent failure: a failure occurs when an error reaches the service interface and alters the service.

- **Failure**: Occurs when the delivered service deviates from the specified service; failures are caused by errors. The user may choose to identify several severity levels of failures, [9] such as catastrophic, major, and minor, depending on their impacts to the system service. The definitions of these severity levels vary from system to system. An outage is a special case of failure that is defined as a loss or degradation of service to a customer for a period of time (called outage duration). In general, outages can be caused by hardware or software failures, human errors, and environmental variables (e.g. lightning, power failures, and fire). A failure resulting in the loss of functionality of the entire system is called system outage. A system may not, and generally does not, always fail in the same way. The ways a system can fail are its failure modes. These can be characterized according to three viewpoints
There exists a causality relationship between faults, errors and failures that define the error propagation chain, as expressed by the arrows of Figure 1.2. They should be interpreted generically: by propagation, several errors can be generated before a failure occurs.

The creation and manifestation mechanisms of faults, errors, and failures are illustrated by Figure 1.3, and summarized as follows:

1. An active fault is either a) an internal fault that was previously dormant and that has been activated by the computation process or environmental conditions, or b) an external fault. Fault activation is the application of an input (the activation pattern) to a component that causes a dormant fault to become active. Most internal faults cycle between their dormant and active states.

2. The computation process causes error propagation within a given component (i.e., internal propagation): an error is successively transformed into other errors. Error propagation from one component (A) to another component (B) that receives service from A (i.e., external propagation) occurs when, through internal propagation, an error reaches the service interface of component A. At this time, service delivered by A to B becomes incorrect, and the ensuing failure of A appears as an external fault to B and propagates the error into B. The presence of an
error within a system can arise from the: a) activation of an internal fault, previously dormant, b) occurrence of a physical operational fault, either internal or external, or c) propagation of an error from another system (interacting with the given system) via the service interface, that is an input error.

3. A failure occurs when an error is propagated to the service interface and unacceptably alters the service delivered by the system. A failure of a component causes a permanent or transient fault in the system that contains the component. Failure of a system causes a permanent or transient external fault for the other system(s) that interact with the given system.

1.3 Basic Concepts and Definitions

Dependability is a very broad term for describing the quality of service provided by a system that integrates such attribute as:

- **Reliability**: The measure of continuous delivery of correct service,

- **Availability**: The measure of the delivery of correct service with respect to the alternation between correct and incorrect service,

- **Safety**: Absence of catastrophic consequences on the user(s) and the environment,
- **Maintainability**: Ability to undergo repairs and modifications, e.g. the measure of the time to restoration of correct service,

- **Integrity**: Absence of improper system state alterations,

- **Confidentiality**: Absence of unauthorized disclosure of information.

Depending on the application intended for the system, different emphasis may be put on different attributes. The description of the required goals of the dependability attributes in terms of the acceptable frequency and severity of the failure modes, and of the corresponding acceptable outage durations (when relevant), for a stated set of faults, in a stated environment, is the **dependability requirement** of the system. Several other dependability attributes have been defined that are either combinations or specializations of the six basic attributes listed above. **Security** is the concurrent existence of a) availability for authorized users only, b) confidentiality, and c) integrity with “improper” taken as meaning “unauthorized”. Characterizing a system reaction to faults, is of special interest, via, e.g., **robustness**, i.e. dependability with respect to erroneous inputs. The term **performance** generally refers to how effectively (e.g. throughput, delay) or efficiently (e.g. resource utilization) a system delivers a specified service, presuming it is delivered correctly. **Performability** can be considered as the unification of the concepts of performance and dependability, i.e. a system ability to perform in the presence of fault-caused errors and failures. Hence, in contrast with performance, effects of faults are considered and, unlike dependability, failure due to faults is not the only concern. Instead (if addressed), a partial failure is regarded as leading the system to one of many levels of service quality that might be accomplished by the object system over a designated period of time.

Other concepts similar to dependability exist, as **survivability** and **trustworthiness**. Survivability is the property of a system, subsystem, equipment, process, or procedure that provides a defined degree of assurance that the named entity will continue to function during and after a natural or man-made disturbance [10]. Trustworthiness is the degree to which a system or component avoids compromising, corrupting, or delaying sensitive information.

Dependability computing finds applications on long-life application (space flight and satellites), critical-computation applications (aircraft flight control systems, military systems, many types of industrial controllers), maintenance postponement applications (when maintenance operations are extremely costly), and high available applications (banking, communication systems).
### Table 1.1: Downtime per year for several availability figures

<table>
<thead>
<tr>
<th>Availability</th>
<th>Downtime/year</th>
</tr>
</thead>
<tbody>
<tr>
<td>90%</td>
<td>&gt; 1 months</td>
</tr>
<tr>
<td>99%</td>
<td>≈ 4 days</td>
</tr>
<tr>
<td>99.9%</td>
<td>≈ 9 hours</td>
</tr>
<tr>
<td>99.99%</td>
<td>≈ 1 hour</td>
</tr>
<tr>
<td>99.999%</td>
<td>≈ 5 minutes</td>
</tr>
</tbody>
</table>

Given a specification in terms of system dependability attributes (non-functional requirements), it is important to validate the system to ensure the compliance with them.

Several of the metrics (attributes) of dependability listed above can be expressed as probabilistic functions [11], [12] because of the **random nature of failure events**. **Reliability** can be equated with the probability that the system does not fail during the period of mission of the system (e.g., a flight). For continuous mission system (e.g., a web server), reliability can also be expressed by the **mean time to failure** (MTTF) or by the **mean time between failures** (MTBF). Finally, reliability can be expressed as failure rate probability, for example, $10^{-9}$ failures per hours, a typical figure for safety-critical systems. Given reliability MTBF and maintainability MTTR (mean time to repair) of a system, availability can be expressed as MTBF/(MTBF + MTTR). For the sake of simplicity, table 1.1 shows the corresponding downtime per year for several availability figures.

The framework of **performability** modeling and evaluation has been firstly defined by Meyer in the late 1970s [13]. In this framework, the performance of a system $S$ over a utilization period $T$ is modeled as a random variable $Y$ taking value in a set $A$ of accomplishment levels. Each element of $A$ represents a possible outcome which can be obtained by $S$. This notion of “performance” is very general, since it refers to any possible aspect of the total system behavior with respect the system’s ability to perform is to be evaluated. Given this model, the performability of $S$ is defined as the probability that $S$ performs at a level in $B$ where $B$ is any measurable subset of $A$, that is $\text{Perf}(B) = Pr\{Y \in B\}$. Performability is evaluated by solving a stochastic process $X = \{X(t), t \in I\}$, referred to as the base model, where $X(t)$ represents the state of the system $S$ at time $t$, and the index set $I$ must include the utilization period $T$, (i.e. $T \subseteq I$). To solve $X$ means to obtain from $X$ values of $\text{Perf}(B)$ for some sets of $B$ of interest of the user.
1.4 The Means for Dependability

The development of a dependable computing system calls for the combined utilization of a set of methods that can be classed into:

- **Fault Prevention (Avoidance)**: How to prevent fault occurrence or introduction.
- **Fault Tolerance**: How to provide a service capable of fulfilling the system function in spite of faults.
- **Fault Removal**: How to reduce the presence (number, seriousness) of faults.
- **Fault Forecasting**: How to estimate the present number, the future incidence and consequences of faults.

**Fault Prevention (Avoidance)**

Used techniques are design review, component screening, testing, quality of control methods, formal methods and software engineering methods in general.

**Fault Tolerance**

Indeed, even with the most careful fault avoidance, faults will eventually occur and result in a system failure.

Fault tolerance is carried out by *error processing*, aimed to removing errors from the computational state, if possible before failure occurrence, and *fault treatment*, aimed at preventing faults from being activated again. In turn, there are two techniques to carry out error processing: error recovery and error compensation. Error recovery means that an error free state is substitute for the erroneous state. The erroneous state must be urgently identified prior to be transformed: this is the purpose of error detection. Error compensation means that the erroneous state contains enough redundancy to deliver an error-free service from the erroneous internal state. Both error recovery and compensation need error detection capability. When error compensation is applied systematically, even in the absence of fault (for example majority vote), it is called fault masking.

Fault treatment (handling) consists of four steps:

- Fault diagnosis, determining the causes of errors in both location and nature,
CHAPTER 1

- Fault isolation (passivation), preventing the faults from being activated again by removing the components identified as being faulty from further executions,

- System reconfiguration, which either switches in spare components or reassigns tasks among non-failed components, whenever the system is no longer capable of delivering the same service. This reconfiguration should enable the delivery of an acceptable service, even if the system capability degrade in a process called graceful degradation [8].

- System reinitialization, which checks, updates and records the new configuration and update system tables and records.

A fault tolerant system can be achieved, for example, by recurring to redundant components: the robustness of the system against faults is increased and the negative effects of faults can be limited [9]. Redundant techniques are useful against independent faults while design diversity is useful against design faults. The use of redundancy can provide the information needed to negate the effects of faults. Redundancy can have a very important impact on a system in the area of performance, size, weight, power consumptions and others. There exists several dimension of redundancy: time redundancy which is provided by software (extra executions of the same calculation which may be accomplished by different methods), components redundancy which is provided by hardware or software (use of extra memory, bus lines, functional modules to supply extra information), and information redundancy (mapping data in new representation containing redundant information to allow fault detection and fault masking).

Fault Removal

Fault removal is obtained by means of a set of techniques used after that the system has been built. They are verification (checking whether the system adheres to properties, termed the verification conditions), diagnosis (diagnosis the fault which prevented the verification conditions from being fulfilled), and correction.

Fault Forecasting

However, whichever fault tolerant technique is used, it is possible for a system to experience a failure scenario that overcomes its capabilities. Hence, the dependability evaluation represents a suitable means to verify the adequacy of a system design with respect to the requirements given in its specification. In this sense, fault forecasting is a way to achieve system assessment
and can be performed using several approaches like testing, fault injection and analytical models, simulations, often combined together. In the following the major focus has been given on means for dependability modeling.

1.5 Models for Dependability and Performability Evaluation

Modeling (both analytical and simulative) approach for dependability evaluation has proven to be useful and versatile in all the phases of the system life cycle. During design phase, models give an early validation of the concepts and architectural choices, allow comparing different solutions to highlight problems within the design and to select the most suitable one. During the operational life of the systems, models allow to detect dependability bottlenecks and to suggest solutions to be adopted for future releases.

One of the main problems that has to be taken into account when we want to model a system is the management of complexity. Being able to describe critical complex systems accounting at the same time for all the relevant aspects is not trivial at all. To master complexity a modeling methodology is needed so that only the relevant aspects can be detailed still allowing numerical results to be effectively computable. A solution in this sense is the introduction of simplifying hypotheses. This is a critical activity because the introduced approximation could limit or invalidate our faithfulness on results. The resulting error should always be estimated, either through sensitivity analysis or by comparing the results returned by the model containing the simplifications and the results of a more complex model or different source assessments.

How has been said before, modeling can be a difficult task for complex system. To manage complexity, another good practice for an effective modeling is to identify different functional parts of the system and model them separately (e.g. through the “separation of concerns” property [2]). In such a way, it is possible to describe the system by several sub-models and in a modular or hierarchical units. A slight distinction can be done between modular or hierarchical decomposition. A hierarchical one is also modular, but is not true the vice versa. With a modular approach different functional parts of the system are described separately. The hierarchical approach can been performed modeling the system at different abstraction level, e.g. [14]. A third modeling solution is an hybrid one where functional concerns are separated in hierarchical way, such as in a protocol stack [15]. To decompose a complex system in sub-models requires that the interactions among the sub-models
are clearly identified and formalized. Then, if, for example, a complex system is modeled by an architectural and an environmental sub-models, this allows reuse of the architectural models, when the same architecture is to be used in different environments and vice versa. In this way, changes can be performed more easily than when the various models are not separated. Another approach to manage complexity is the behavioral decomposition. Indeed, in case of graceful degradable computer systems, two types of events can be observed: performance events, causing performance state changes, and dependability events, causing changes in the structure of the system. In fact, sometime the solution of a model by simulation can be complex. That happens because performance events usually take place far more often than dependability events [1]. Hence it seems reasonable to assume that the performance of the system, between two successive dependability state changes, is in steady state most of the time. Thus, the percentage of time for which the system is not in steady–state between successive structure state changes is assumed to be negligible. This means that the overall model can be decomposed in a structural model to describe the changes in the system structure, and a family of performance models (in general, one for each possible configuration that the system structure can assume) which describe the performance behavior of the system. Such performance models can be solved for their steady–state (which is usually simpler to deal with, in comparison to transient state), and then the information so obtained must be conveyed in some way to the structural model. The resulting combined model can be solved to obtain steady-state or transient performability measures [16].

As already said dependability and performability evaluation is a way to determine if a system respects its specification and requirements with satisfactory accuracy. A lot of techniques has been devised for their quantitative evaluation. All of them can be classified in two big categories: measurement–based and modeling–based techniques. Let us consider only the second ones, on which the remainder of this dissertation is based on.

In turn, modeling–based methodologies can be distinguished in analytical and simulative. In a model all the system components are represented by variables and parameters and the interactions among them are described by some relations among these quantities. Simulative models differ from the analytical ones for the reproduction of the dynamic behavior of the system over the time. Their evaluation requires the run of a dedicate program (simulator) able to represent the temporal evolution of the system and to give the estimate of the measures of interest. One of the main advantages of simulation over analytical modeling is the flexibility and generality of the solvable models; on the other side, if a high level of detail for the representation of the system is required, the cost of development and execution of the evaluations
raises. Simulation for dependability application will be discussed again in Section 1.9.

In the following Sections some analytical modeling methodologies that have been developed for the purpose of software dependability and performability evaluation, are presented. In general, any particular method, if used exclusively, will suffice; what will generally be required is some appropriate combination of techniques for model specification, construction, and solution. Among the analytical modeling methodologies, a first distinction can be made between those employ combinatorial models (see Section 1.6) like Fault-Trees, Series-Parallel Diagrams, and those based on state-space oriented representations (see Section 1.7), such as Queueing models, Markov chains, Petri Net models. Even if the combinatorial models offer great simplicity and intuitiveness for the construction and solution, they are not adequate to deal either with systems that exhibit complex dependencies among components, or with repairable systems. This is the reason why their use did not receive the same interest as the more powerful state space based approaches. Many of the existing modeling techniques are supported by automated tools like SURF2 [17], SHARPE [18], [19], SPNP [20], GreatSPN [21], UltraSAN [22], TimeNET [23], Möbius [24], and DEEM [25] for the assisted construction and solution of dependability models.

1.6 Combinatorial Models

Combinatorial modeling is a failure-to-exhaustion approach, in which the system is divided in non-overlapping modules. Each module is assigned either a probability of working, $P_i$, or a probability as a function of time, $R_i(t)$.

The following assumptions are made:

1. Module failures are independent.
2. Once a module has failed, it is assumed always to yield incorrect results.
3. The system is considered failed if it does not satisfy the minimal set of functioning modules.
4. Once a system enters a failed state, subsequent failures cannot return the system to a functional state.

Between the combinatorial models [26] there are Reliability Block Diagrams, Fault Trees, and Reliability Graphs.
1.6.1 Reliability Block Diagrams

A series-parallel reliability block diagram represents the logical structure of a system with regard to how the reliability of its components affects the system reliability. In a block diagram model, components are combined into blocks in series, in parallel or in $k$-out-of-$n$ configurations. A $k$-out-of-$n$ [26] structure is a superset of the series parallel structures and requires $k$ of the $n$ total components to be functionals for an operational system. Therefore, parallel and series structures are represented with $k$-out-of-$n$ structures that are 1-out-of-$n$ and $n$-out-of-$n$, respectively. The equations for the distribution function of these structures are:

$$F(t) = \begin{cases} 1 - \prod_{i=1}^{N}(1 - F_i(t)) & \text{for a series structure,} \\ \prod_{i=1}^{N} F_i(t) & \text{for a parallel structure.} \end{cases} \quad (1.1)$$

The distribution function for the $k$-th order statistic of $n$ independent, identically distributed random variables is:

$$F_{k|n}(t) = \sum_{i=k}^{n} \binom{n}{i} F(t)^i (1 - F(t))^{n-1}.$$ \quad (1.2)

All of these constructs can be used together in a single block diagram. Components of the same type that appear more than once in the system are assumed to be copies with independent, identical distribution functions. Each component may have a failure probability, a failure rate or a failure distribution function attached to it. The assumption of independence and series-parallel structure allows very fast computation of reliability and availability measures. However, many system models in practice do not follow series-parallel structure.

1.6.2 Fault-Tree

Fault-trees are a deductive modeling and analysis technique based on the study of the events that may impair the dependability of a system [27]. A Fault-tree considers the combination of events that may lead to an undesirable situation of the system. In the study of reliability, this undesirable event is system failure event. In assessing the safety of the system, the undesirable event is the potentially hazardous or unsafe condition. Fault-trees are fruitfully employed during the first design phases of a system, in that they are able to represent at a higher abstraction level the various failure scenarios that can affect the dependability figures of the system.

The fault-tree is a pictorial representation of the combination of events that can cause the occurrence of an event. It represents all the sequences of
individual component failures that cause the system to stop functioning, in
a tree–like structure. The construction of a fault-tree model follows a top-
down approach. The starting point is the definition of a single, well–defined
undesirable event, which is the root of the tree. Starting from the root of
the Fault-tree, each event is decomposed as a logical combination of a set of
simpler events, until a set of elementary events is found that are not further
decomposed. A fault-tree is formed by a series of layers of events connected
through logic gates that represent the two Boolean operators AND and OR.
An event at level “i” is thus reduced to a combination of lower-level events.
The process of reduction stops when we reach basic events that we do not
wish to reduce further. An event can be classified as elementary (basic) either
because its occurrence probability is known, or because it is not possible to
carry on the decomposition. Distinct basic events can be component failures,
human errors, external conditions, etc. Usually, the basic events are assumed
mutually independent with a given failure probability, a failure rate or a
failure distribution function. The occurrence of each event is denoted by a
logic 1 at that node; otherwise the logic value of a node is 0. Each gate
has one or more inputs and a single output. The input to a gate is either a
basic event or the output of another gate. Each outgoing event of a port is
obtained by combining the incoming events according to the logic expressed
by the gate itself. The output of an AND gate is a logic 1 if and only if all
of its inputs are logic 1. The output of an OR gate is a logic 1 if and only if
one or more of its inputs are at logic 1. The output of a $k$–out–of–$n$ gate is a
logic 1 if $k$ or more of the inputs are at logic 1. If two gates share an input,
the fault tree is said to have repeated events.

The analysis of a fault-tree model for the evaluation of the probability of
occurrence of the root event is based on the computation of the set of cuts.
A cut is defined as a set of elementary events that, according to the logic
expressed by the fault-tree, leads to the occurrence of the root event. The
numerical solution of the fault-tree is performed by computing the probability
of occurrence for each of the cuts, and by combining those probabilities to
estimate the probability of the root event. Among the possible cuts of a fault-
tree, the minimal cuts provide additional information on the dependability
characteristics of the modelled system. A cut is said to be minimal if it
does not contain any other cut. With the analysis of the minimal cuts, it is
possible to identify the simplest combinations of elementary events that bring
the system to the failure. In particular, it is possible to discover whether the
system has single-points of failure, that is minimal cuts formed by a single
element, which may represent dependability bottlenecks.

When there are not repeated events in a fault–tree, the cumulative distri-
bution $F(t)$ of the root event can be determined with the following equations:

$$
F(t) = \begin{cases} 
\prod_{i=1}^{N} F_i(t) & \text{and gate}, \\
1 - \prod_{i=1}^{N} (1 - F_i)(t) & \text{or gate}, \\
\sum_{i=k}^{n} \binom{n}{i} F(t)^i (1 - F(t))^{n-1} & \text{k-out-of-n gate, i.d.} 
\end{cases}
$$

(1.3)

where $F_i(t)$ denotes the time to failure cumulative density function of component $i$.

When there is a repeated component, these equations cannot be used, because the failure distributions are no longer independent. For these cases, it is first necessary to obtain cut-sets and then use the sum of disjoint product algorithm.

### 1.6.3 Reliability Graphs

Reliability graph is a directed graph where the edges represent the components of the system being modeled and are assigned a given failure probability, failure rates or failure distribution functions, the same as reliability block diagrams. Sometimes the edges do not represent system components, but they represent the structure of the system and an “infinite” distribution (they never fail) is assigned to them. The graph contains one node with no incoming edges called the source, with no incoming edges and a node called the sink (also called destination or terminal nodes) with no outgoing edges. A system represented by a reliability graph fails when there is no path from the source to the sink.

A reliability graph is equivalent to a non-series-parallel reliability block diagram. In the reliability graph, the components are the arcs, while in the block diagram the components are the boxes.

### 1.7 State-Based Models

A powerful tool for modeling systems composed of several processes is the Markov model. Among Markov-model types used for dependability modeling there are:

- Queueing Models [28],
- Continuous-Time Markov Chains (CTMC) and Discrete-Time Markov Chains (DTMC) [12],
- Generalized Stochastic Petri Nets (GSPN) [29],
Markov Reward Models (MRM) [30],
Stochastic Reward Nets (SRN) [30].

Their main characteristics will be briefly described in the following Sub-
sections.

1.7.1 Queueing Models

While combinatorial (non state-space) models serve as effective mod-
els for evaluating how well an individual entity (e.g., a task or a program)
performs, they are unable to account for the effects of resource and data
contention on performability. For this reason, measures that reflect conse-
quences of design faults, such as effective throughput cannot be accommo-
dated. Queueing models are much better suited to the representation of
operational environments, internal scheduling disciplines, program execution
modes (sequential or parallel), and resource/data contention. Moreover, such
models can be extended (e.g. queues with breakdowns [31]) so to represent
systems with faults. Typical applications of this methodology are, for exam-
ple, the queue of jobs waiting for main-memory allocation (the job schedul-
ing queue), the queue of jobs that have been allocated main memory but are
forced to wait for CPU service (the process scheduling queue), the queue of
jobs waiting for some form of I/O service (the I/O scheduling queue), etc.
In these examples “customers” are logical entities, such as transactions and
control functions, that wait for services provided by “physical servers” such
as main memory, a CPU or an I/O device.

1.7.2 Markov Chains

If the measure considered admits a representation of system state such
that: i) its evolution in time is independent from the past, depends but only
on the present value and ii) the number of different states is relatively small,
then it is possible to represent the system directly by a Markov chain - a
Markov process with a discrete state space [12]. Markov chains are a state-
based formalism, very close to that of the automata. The set of possible
states of a Markov chain is called the state space, and is denoted by $S$.
If the state space $S$ is a discrete or countable set, we are properly talking
of a Markov chain, and its cardinality is denoted by $C$. If the state space
is a continuous set, then the term Markov process is more appropriate. A
state change of a Markov chain is called a state transition. More formally,
a Markov chain is a stochastic process \{$X(t), t \geq 0$\} with a discrete state
space such that for any $n > 0$ and any sequence of increasing time instants $t_1, t_2, ..., t_n, t_{n+1}$, the following equation holds:

$$
Prob\{X(t_{n+1}) = j|X(t_n) = i_n, X(t_{n-1}) = i_{n-1}, ..., X(t_1) = i_1\} = Prob\{X(t_{n+1}) = j|X(t_n) = i_n\}, \; \forall j, i_n, i_{n-1}, ..., i_1 \in S
$$

In other words, the above definition (also known as memoryless or Markov property) states that the probability of any particular future behavior of the process, when its current state is known exactly, is not altered by additional knowledge concerning its past behavior. If the exact characterization of the present state of the process is independent from the current time, then the Markov chain is said to be time-homogeneous, otherwise it is said to be a non-homogeneous Markov chain. The parameter $t$ that indexes the Markov chain can be either discrete or continuous. In the first case we have a discrete-time Markov chain $\{X_n, n \geq 0\}$ where state transitions only occur at discrete points in time, often called steps, whereas in the latter case we have a continuous-time Markov chain $\{X(t), t \geq 0\}$ and state transitions may occur at any point in time. Because of the memoryless property, each transition from state $i$ to state $j$ of a homogeneous continuous-time Markov chain occurs in an exponentially distributed time, and the rate of the transition is exactly the inverse of the expected time to the transition, that is the rate of the corresponding exponential distribution. Therefore, in a homogeneous Markov chain, all the transitions among states occur in a negative exponentially distributed time. This implies that time needed to perform whichever activity of a system must be modelled with an exponential transition in the Markov chain model of that system. This is the most severe constraint that limits the applicability of Markov chains. Whenever an activity having non-exponential duration is modelled with an exponential transition, an approximation is introduced in the model and the evaluation results can be significantly different from the exact ones. Several techniques exist to limit this error, such as the phase expansion [32], that uses a sequence of exponential stages to approximate a non-exponential random variable.

Modeling the dependability of a system by using a Markov chain requires first a projection of the set of possible states of the system to define the state space $S$. Then, the matrix (transition rate or probability) ruling the transitions of the Markov chain model from the initial state must be defined. The solution of the Markov chain focuses on the computation of the state occupation probabilities for some of the states of $S$. The pointwise and expected values of the most important dependability measures can be evaluated from
dependability concepts

the time-dependent occupation probabilities. The evaluation of combined metrics such as performability measures requires the definition of a reward structure, which assigns reward values to the states of the Markov chain model. We denote with \( W_i \) the reward associated to state \( i \), for each \( i \in S \), and with \( \mathbf{w} \) the vector that collects the reward values of all the states in \( S \). Quite informally, a reward \( W_i \) can be assigned to represent the gain achieved by the system for any unit of time spent in state \( i \). A performability measure can be computed as the total reward cumulated as the model evolves over time. This evaluation requires the derivation of the state occupation probabilities, too. For a discrete-time homogeneous Markov chain, the reward cumulated over the first \( n \) steps is denoted with \( W(n) \), and is evaluated as follows:

\[
W(n) = \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{\pi}_i
\]  

And the reward \( W(t) \) cumulated during the time interval \([0, t]\) for a continuous-time homogeneous Markov chain is computed as follows:

\[
W(t) = \int_0^t \mathbf{w} \cdot \mathbf{\pi}(t) dt
\]  

1.7.3 Automated Generation of Markov Chains

When Markov chains are applied to analyze the reliability, availability, and performance of a system the primary procedure consists of the following steps: abstracting the physical system at first, constructing the Markov chain, and then setting up ordinary differential equations (for transient solutions) or linear equations (for steady-state solution) manually, and finally writing a program for the numerical solution to the equations. It is a rather tedious and error-prone procedure, especially when the number of states become very large. Since late 1980s, some researcher have been developing new modeling formalism and software package for the automated generation and solution of Markovian stochastic systems based on Petri nets.

Petri nets are a promising formalism for describing and studying systems that are characterized as being concurrent, asynchronous, distributed, parallel, non-deterministic, and/or stochastic [33]. The ability to describe the qualitative and quantitative aspects of complex systems, and their intuitive and appealing graphical representation made the success of this modeling paradigm.

Firstly, the ordinary Petri nets are described. Subsequently, various kinds of stochastic Petri nets and their application are discussed.
1.7.4 Place Transition Petri Nets

Petri nets were originally introduced by C. A. Petri in 1962. Formally \[12\] \[29\], a place-transition Petri net (PN) is a 5-tuple \( PN = (P, T, A, M, \mu) \), where:

- \( P = \{P_1, P_2, ..., P_n\} \) is a finite set of places (draw as circle).
- \( T = \{t_1, t_2, ..., T_n\} \) is a finite set of transitions (draws as bars).
- \( A \subseteq (P \times T) \cup (T \times P) \) is a set of arcs connecting \( P \) and \( T \). Arcs going from a place to a transition are called input arcs, and arcs directed from a transition to a place are called output arcs.
- \( M : A \rightarrow \{1, 2, 3, \ldots\} \) is the multiplicity associated with the arcs in \( A \).
- \( \mu : P \rightarrow \{0, 1, 2, \ldots\} \) is the marking that denotes the number of tokens (drawn as black dots or a positive integer) for each place in \( P \). The initial marking is denoted with \( \mu_0 \).

The places that are linked to transition \( t \) by an input arc are called the input places of the transition. Similarly, those places linked to transition \( t \) by an output arc are called the output places of the transition. In the graphical representation of the Petri net model, places are drawn as circles and transitions are drawn as bars, with the input and output arcs linking them. Places may contain tokens, which are represented as black dots. The state of the Petri net model is a vector \( (m(P_1), m(P_2), ..., m(P_n)) \) called the marking of the net, and is defined by the number of tokens \( m(P_i) \) in each place \( i \) of the model. Transitions model activities which can occur (the transition fires) and change the state of the system (the marking of the Petri net). Transitions are only allowed to fire if they are enabled, which means that all the preconditions for the activity must be satisfied (there are enough tokens available in the input places). When the transition fires, it removes tokens from its input places and adds some at all of its output places. The number of tokens removed/added depends on the cardinality of each arc. A place may be an input place for more than a single transition. This leads to a competition of the enabled transitions for the tokens contained in the input places. When two enabled transitions share an input place and the number of tokens therein is not sufficient for both of them to fire, the transitions are said to be in conflict, and a selection rule must be employed to break the competition in favor of one of them. Usually, a priority is assigned to solve the conflicts. A system can be modelled by representing its states as markings of the Petri Nets. Tokens can be used to represent entities of the
system, such as tasks to be executed, messages to be sent. Transitions model activities or synchronization constraints of the system, and the firing rules define the preconditions to be satisfied for the activities to be executed or the synchronization to be completed, respectively. The absence of time in the class of place-transition Petri nets does not allow quantitative analysis of the modelled systems. This formalism was mainly introduced to model qualitative aspects of systems (concurrency, parallelism) and to verify its structural properties (like, for example, the absence of deadlocks, a given order in the actions performed etc.).

1.7.5 Stochastic Petri Nets

A very popular timed extension of the place-transition Petri nets is the class of stochastic Petri nets [34], denoted as SPNs in the following. In a SPN model, each transition \( t \) has an associated random firing delay whose probability density function is a negative exponential. The introduction of the firing delay imposes the definition of a more precise firing rule. The enabling of a transition is exactly defined as it was in the case of the place-transition Petri net models. As soon as a transition \( t \) gets enabled, a random firing time is sampled from the distribution associated to \( t \), and a timer starts counting form that time down to zero. Transition \( t \) fires if and only if it remains continuously enabled until the timer reaches zero. When \( t \) fires, the tokens are removed from its input places and added to the output places in a single atomic and instantaneous operation (atomic firing rule). It is interesting observing that in the time interval between the enabling and the firing of \( t \), other transitions sharing some input places with \( t \) can get enabled and fire without disabling it, provided that there is a sufficient number of tokens in the common input places. On the contrary, in the case of a conflict, the transition whose timer reaches zero the first is the one that fires (race model). It is also important to notice that the use of exponential distribution relieves the user from the specification of the behavior of those transitions that do not fire after having been enabled. Indeed, thanks to the memoryless property of the exponential distribution, whether the memory of the time they have already been enabled is kept or not, the remaining time to the firing is exponentially distributed with the same rate. The evolution of a SPN model can be represented by a continuoustime homogeneous Markov chain, whose state space elements are in a one-to-one correspondence with the elements of the reachability set, and whose transitions rates among states are equal to the firing rates of the transitions that produce the corresponding marking change in the SPN. An SPN model can be solved in terms of the marking occupation probabilities by performing the analysis of the associate
Markov chain. A reward structure can be associated to the markings of a SPN model, to evaluate some Performability metric of interest. The reward values assigned to the various markings need to be translated into the corresponding rewards over the state associated Markov chain.

1.7.6 Generalized Stochastic Petri Nets (GSPNs)

The class of generalized stochastic Petri nets, denoted by GSPNs [29], relaxes the assumption that all the transitions have an exponentially distributed delay, and allows for exponential transitions, and for instantaneous transitions as well, that is transitions that once enabled fire in zero time. Conflicts among timed transitions are solved according with the same race model as in the case of SPNs, whereas conflicts among instantaneous transitions are solved by a priority assignment, and by associating weights (or probabilities) to instantaneous transitions at the same priority level. The solution of a GSPN model resorts again to that of an associate Markov chain. However, for GSPN models, the reachability set elements are not in a one-to-one correspondence with the state of the associate Markov chain. Indeed, because of the instantaneous transitions, some of markings in the reachability graph have a zero sojourn time, that is the GSPN model spends a zero time therein. These markings are called the vanishing markings of the GSPN model, whereas the nonvanishing markings are often called tangible markings. Nevertheless, it is possible to operate a reduction of the reachability graph to eliminate the vanishing markings, and to obtain the reduced reachability graph. The reduced reachability graph is isomorphic to a Markov chain, and the reduction procedure does not affect the equivalence between the non vanishing marking occupation probabilities of the GSPN and the state occupation probabilities of the Markov chain. Therefore, the Markov chain associated to the reduced reachability graph can be solved to study the GSPN model evolution over time.

1.7.7 Stochastic Reward Nets (SRNs)

The specification of a system using GSPN’s can be tedious and troublesome. To remedy this, Ciardo and al. [35] introduced several structural extensions to GSPNs. Variable multiplicity arcs, enabling functions (also known as guards) for transitions, marking dependent arc multiplicities and timed transition priorities. The resulting net with all these extensions and capability of assigning a real value reward to any marking is termed as a stochastic reward net (SRN).
1.7.8 Deterministic and Stochastic Petri Nets (DSPNs)

The standard definition of GSPNs (and SRNs) implies that all the timed activities associated to the transitions have an exponentially distributed duration, so that the evolution of the model can be mapped into a continuous-time homogeneous Markov chain. As we already pointed out, this constraint implies that an approximation is introduced into the models whenever an activity with nonexponential duration must be represented. Recently, several classes of Petri nets have been defined, which allow for transitions whose firing times can be drawn from non-exponential distributions. These new classes of Petri nets are collectively called non-Markovian Petri nets [36]. Among non-Markovian Petri nets, we consider particularly interesting for the purposes of this dissertation the class of deterministic and stochastic Petri nets [37], denoted by DSPNs. DSPNs have been introduced as an extension of GSPNs, to allow the modeling of events having deterministic occurrence times [38]. The set of transitions of a DSPN can be partitioned into three disjoint sets: the set of instantaneous transitions, represented by a thin bar, the set of transitions having exponentially distributed firing times, represented by empty rectangles and the set of transitions with deterministic firing times represented by filled rectangles. This enriched set of possible transitions offered by DSPNs allows the exact modeling of a wider set of system features, such as timeouts and the message propagation delays in synchronous systems. Repair delays represent another example of activities that are typically more accurately modelled by deterministic transitions than by exponential ones. Unfortunately, the analytical solution of a DSPN model is not possible in general. Indeed, the deterministic distribution does not enjoy the Markov memory-less property, and the time-dependent evolution of the model requires keeping track of much additional information [39], which greatly complicates the analysis. However, the analytical tractability is guaranteed for the subset of DSPN models whose structure satisfies the following assumption: “at most one deterministic transition is enabled in each of the possible markings of the DSPN”. This hypothesis severely limits the expressiveness of DSPN models, nevertheless, quite recently some attempts have been made to relax it [40], [37].

1.7.9 Stochastic Activity Networks (SANs)

Stochastic extensions to Petri nets have been widely recognized as a powerful, versatile, tool for efficiently describing complex discrete event systems. For example, stochastic activity networks (SANs) have been demonstrated as an effective framework for discrete event system modeling. Stochastic Activ-
ity Networks (SANs) were first introduced in [41] and are currently employed in evaluation tools such as METASAN [42] and UltraSAN [22] Möbius [24]. These tools provide both analytical solvers and simulators. The underlying stochastic process of analytical models is the same as GSPN, when all timed transitions (activities, in the SAN language) are exponentially distributed, or DSPN when at most one deterministic activity is enabled at the same time.

The primitive elements used to construct a SAN are places, activities, arcs, and gates [43]. Places in SANs have the same interpretation as in Petri nets. Places hold tokens, which represent system resources, such as customers in a queue, working components, etc. The number of tokens in a place is called the marking of that place. The vector consisting of the marking of each place in a SAN is called the marking of the SAN. There are two types of activities, timed and instantaneous. Timed activities are used to represent delays in the system that affect the performance/dependability measure of interest, while instantaneous activities are used to abstract delays deemed insignificant relative to the performance and dependability measure. Uncertainty about the length of the delay represented by a timed activity is described by a continuous probability distribution function called the “activity time distribution function”. Given a timed activity \(a\), and a marking \(\mu\), the activity time distribution function of \(a\) in \(\mu\) is denoted by \(F(\cdot; \mu; a)\). The semi-colon notation used here indicates \(F\) is parameterized by the activity. Activities can have cases. Cases are used to represent uncertainty about the action taken upon completion of an activity. \(C(\cdot; \mu; a)\) is the case distribution of \(a\) in \(\mu\), a discrete probability distribution over the cases.

Gates connect activities and places, defining interdependencies among system resources and system delays. There are two types of gates. Input gates have a predicate, a Boolean function of the markings of the connected places, and a output function. Input gates are connected to one or more places and the input of a single activity. When the predicate is true, the gate holds.

Output gates are connected to one or more places, and the output side of an activity. If the activity has more than one case, output gates are connected to a single case. Output gates have only a function. Gate functions provide flexibility in defining how the markings of connected places change when the delay represented by an activity expires.

Arcs in SANs are default gates, defined to duplicate the behavior of arcs in Petri nets. Thus, arcs are directed. Each arc connects a single place and a single activity. The arc is an input arc if it is drawn from a place to an activity. An output arc is drawn from an activity to a place. An input arc holds if there is at least one token in the connected place. The function of an input arc removes a token from the connected place, while the function
of an output arc adds a token to the connected place.

An activity is *enabled* when all of its input gates hold. A marking in which only timed activities are enabled is called a *stable marking*. If instantaneous activities are enabled, the marking is *unstable* or *vanishing*. We represent the set of enabled activities in a stable marking, $\mu$, by $en(\mu)$.

### Activation and Reactivation Functions

When an activity becomes enabled, it is *activated*, that is, scheduled to *complete*. Activation also occurs when an activity completes, but remains enabled. The marking at activation time is known as the *activation marking*. The time between activation and the scheduled completion of an activity, called the *activity time*, is sampled from the activity time distribution, $F(\cdot; \mu; a)$, where $\mu$ is the activation marking. The completion of an activity in a SAN is analogous to the firing of a transition in a Petri net. Upon completion of an activity, input gate functions are executed first, followed by output gate functions. These functions operate on the current marking of the SAN to produce the next marking. Activities that are activated are not required to complete. An activity is *aborted* when the SAN moves into a new stable marking in which one or more of the activity’s input gates no longer hold. *Reactivation* occurs when the SAN enters a stable marking that is an element of the set of *reactivation markings of $a$* in $\mu$, denoted $G(\mu; a)$. The marking argument of $G$ refers to the activation marking of $a$. $G(\mu; a)$ returns the set of reactivation markings for $a$ when $a$ is activated in $\mu$. When an activity is reactivated, it is aborted and activated. Given a timed activity, $a$, and reactivation marking of $a$ in $\mu$, $\mu$, the new activity time of $a$ is sampled from $F(\cdot, \mu; a)$

### Reward and Impulse Measurements

Upon completing the model of the system, one must specify the performance/dependability measures of interest in terms of the model. In the SAN modeling framework, performance/dependability measures are specified in terms of *reward variables* [44]. Unformally, given $\{X_t, t \geq 0\}$ the stochastic process modeled and $M$ the set of all possible states, a reward variable collected at an instant of time conventionally denoted by $V_t$ is defined as

$$V_t = \sum_m R(m)P(X_t = m) + \sum_a C(a)I^n_t$$

where $R(m)$ is the reward earned when the marking is $m \in M$, $C(a)$ is the reward earned upon completion of transition $a$, and $I^n_t$ is the indicator of
The event that $a$ was the activity that completed to bring the SAN into the marking observed at time $t$. More precisely, $R(m)$ is the rate at which the reward accumulated in state $m$. $R(m)$ is specified by one or more couple predicate–function. When the predicate is true, the reward is accumulated at rate defined by the function. The steady-state reward can be obtained as $V_{t \rightarrow \infty}$. Variable of the interval of time can be considered. In this case, the reward accumulated is related both to the number of times each activity completes and the time spent in a particular marking during an interval.

\[ Y_{[t,t+l]} = \sum_{m} R(m)J_{[t,t+l]}^m + \sum_{a} C(a)N_{[t,t+l]}^a \] (1.7)

where $J_{[t,t+l]}^m$ is a random variable representing the total time that the SAN is in the marking $m$ during $[t,t+l]$ and $N_{[t,t+l]}^a$ is a random variable representing the number of completions of activity $a$ during $[t,t+l]$.

1.8 Model Power-Hierarchy

There exists a hierarchy [45] among the combinatorial and state–based model types. In fact, counter assertions can be provided those show that there exists an instance of a model type for which no equivalent instance of another model type exists. Modeling power of a model type is determined by the kinds of dependencies within subsystems that can be modeled and the kind of dependability measures that can be computed. Under the assumption that failure and repair time distributions of system components are exponentially distributed the power hierarchy shown in Figure 1.4 holds. However tradeoffs exist between Markov and combinatorial model types. In fact, whereas analysis of combinatorial model types does not put any restrictions on the nature of the distributions, the Markov model types can be extended but usually became intractable under non exponential distributions. Semi–Markov chains have often been used in reliability modeling to allow for non exponential distributions [46] or Markov Regenerative Models [37].

1.9 Simulation Techniques

Although Petri nets have many good properties, which make them superior to other formalisms, they also have some serious limitations. As noticed by many authors (e.g., [47]), one main problem is that when the complexity of the systems to be modelled increases, the corresponding reachability graph
of the Petri net will become very huge (because of the explosion in the number of states). This makes the problem of analysis unmanageable, due to the size of the Markov process that is generated. Moreover, when very general distributions are required for modeling a system, to proceed with an analytical solution may become impossible. In many cases, simulation techniques can be employed for the evaluation of the overall behavior of a model and for quantitative analysis, but they have to be handled with care because they are computationally expensive and can give approximate results. Stochastic activity networks can be solved by both analysis and simulation, depending on system characteristics and the tool at hand. UltraSAN provides this feature and that has been used extensively as solution method for the GPRS random access procedure model presented in the following Chapters.

Simulation provides a powerful way to predict performance and dependability if the system under study has not been implemented. Among the variety of simulations techniques we recall: *emulation*, *trace-driven simulation*, and *discrete-event simulation*. An example of emulation is the use of one available processor to emulate the instruction set of another processor (target) which is not available or under design, so to provide very detailed information about the behavior of the target system. A trace-driven simulation system consists of two components: an event generator (or trace generator) and a simulator. The event generator produces a trace of execution events which are used as input to the simulator. The simulator uses the traced data and simulates the target architecture to estimate, for example, the time to perform each event on the architecture under study. Traces are usually obtained by monitoring the executions of programs but they can also be “artificially” given by some dedicated models. The term discrete event refers to the fact that state variables change instantaneously at distinct points in time.
In a continuous simulation, variables change continuously, usually through a function in which time is a variable. The basic building blocks of a model for discrete event simulation are the possible states and events, one or more simulation clocks for recording the passage of (simulated) time, a mechanism for randomly generating the different kinds of events, and a mechanism for generating state transitions.

Large simulations take enormous amounts of time on sequential machines, which limits the application of the technique. Moreover several problems can arise also generating independent random numbers in large simulation and analyzing output data [48].

1.9.1 Importance Sampling Techniques

Importance Sampling is a powerful technique to speed up simulation time. This is more important in the dependability field, where reliability/availability models with repair policies have a big difference of their transitions, so the time to solve the model can be huge. Indeed, they can have a large state space dimension that adds difficulties to handle it numerically or analytically with classical resolution techniques (see [43], [49] for importance sampling in UltraSAN and [50], [51], [52], [53], [54], [55] for importance sampling applied in the field of dependability evaluation).

1.10 Conclusions

This Chapter has introduced the main concepts and the modeling formalism and techniques used throughout this dissertation. Both analytical modeling and simulators, based on Petri Net formalisms, has been used in the following chapters.

Chapter 2 will introduce the target systems. Chapter 3, 4, and 5 deal with the GPRS random access procedure. In this case the SAN formalism has been used for the model specification and the UltraSAN tool for its solution. Chapter 6, and 7 are dedicated at the problem of database maintenance. In this case DSPN formalism has been used and the tool DEEM for the model solution. In the first case a simulative scheme solution has been necessary for its flexibility in modeling complex phenomena, while in the second one an analytical approach has been pursued, thanks to mathematical tractability of the problem.
Chapter 2

Analysis of Telecommunication Systems

2.1 Introduction

It has been proved that today’s cellular infrastructures of 2nd and 3rd generation have failed to address critical situations efficiently so far, a fact that was dramatically demonstrated during, e.g., earthquakes, news year’s eve, public event, etc. Mobile network operators received strong criticism on their failure to address the communications need of their users. Innovative technologies and mechanisms joined with their deep analysis in terms of both dependability and performance, have to be applied to cope with optimization in terms of survivability, user satisfaction, and operator revenue of the above-mentioned systems.

The performance of any cellular network is dependent on many different factors related to e.g. traffic workload, geographical area and propagation conditions. One way to assess the efficiency of a network is to use a real network and load it with different service types etc. and measure indicators related to its performance. Often, the unavailability of such a network makes that approach impossible and even if a network is available, it is not easy or even possible in some cases to try different scenarios in the network and evaluate its performance. That is the case, for example, of scenarios where faults affecting the system are rare events and observing their effects could take very long time.

Both simulation-based approach and analytical modeling offer another way of tackling the problem of system performance assessment. They have, of course, its drawbacks compared to real system (e.g. accuracy problems,
computational burden etc.) but in most cases they can provide indicative results and the possibility to test such scenarios that occur very seldom in real networks but still might cause a lot of trouble when they occur.

This Chapter introduces the target systems and their issues involved in the dependability and performability analysis shown in the remainder of this dissertation. In particular, Section 2.3 describes the GPRS random access procedure, while Section 2.4 the database maintenance problem.

2.2 The Needs for Dependability

Dependability planning of telecommunication systems is one of the major concern for the operating companies. In fact, the system has to be available as long as possible to increase operator revenue and reduce users’ dissatisfaction dues to dropping of a call or blocked service.

At standardization level, the CCITT (Comité Consultatif International Telegraphique et Téléphonique), now ITU-T (International Telecommunication Union, Telecommunication Standardization Sector), indicates the guidelines on models and methods for dependability planning in the Recommendation E.862 [56]. There, analytical modeling is indicating as the lead way to assess dependability, both from the point of view of the operating companies and for the customers. The basic principle of dependability planning is to find a balance between the customers’ needs for dependability and their demand for low costs. The dependability is described in terms of availability, reliability, and maintainability measures as defined in [57]. Quantitative models are recommended to express fault consequences in terms of money (as additional cost factors) in planning and cost optimization. The cost factors reflect the customer’s experience of faults in the network, quantified in term of money, as well the operating company’s costs for low traffic revenue and corrective maintenance. Modeling analysis can help operating companies to reduce the risk of customer’s complaints and loss of business to competitors as well the risk of unnecessary investments. Thus, there exist needs by operating companies to integrate dependability as a natural part of planning of telecommunication networks.

That is proven by several European projects where network operators are involved in to improve and assess dependability of the present and future personal wireless networks, e.g. [58], [59], and [60].

This dissertation represents a step toward this direction: our study contributes to better understand the performance and performability (when we also consider the effects of outages) of some aspects of wireless systems. Our aim is to analyze, on one side, the performance and the congestion of
the GPRS in the air interface due to outages (random access procedure to GPRS, users’ QoS perceptions) and, on the other side, to propose a new scheme for database maintenance tailored for maintenance of databases supporting wireless systems. We characterize both the working load scenario of such systems and their behavior during outages. These analyses can be also useful in devising system resource management as, for example, in [58].

2.3 General Packet Radio Service

General Packet Radio Service (GPRS) is being specified as a part of the GSM phase 2+ . The principal objectives of GPRS are high data rate, flexibility and efficiency utilization of scarce bandwidth across the air interface. The introduction of GPRS is a first step towards the full deployment of packet-data wireless networks. The use of the GSM circuit-switched transmission mode with data traffic, typically characterized by frequent alternation between activity and idle periods of the data source, results in an inefficient use of the scarce radio resources. In fact, in circuit switching allocation mechanisms, with high set-up time as in GSM, it is necessary to allocate a channel to a Mobile Station (MS) for all its transmission time without taking into account its real activity during this time. The GPRS introduces a packet oriented data service for GSM with a more efficient packet switching allocation mechanism. An important goal of the GPRS technology is to make it possible for GSM license holders to share physical resources on a dynamic, flexible basis between packet data services and other GSM services. The following Sections summarize the main characteristics of the GPRS, emphasizing the radio link protocol [61] [62].

2.3.1 A System Point of View

Figure 2.1 illustrates the GPRS system architecture ([61], [3]). Both GSM [63] and GPRS rely on databases for user mobility management and billing purpose. The Visitor Location Register (VLR) is a database storing actual user related information of the users currently served in a given area and the Home Location Register (HLR) holds further user information, like the actual location and the subscription data of the users. Besides the standard Base Transceiver Station (BTS) and Base Station Controller (BSC) components already present in GSM, GPRS will include two new network elements in order to create an end-to-end packet transfer mode. In fact, packet routing and transfer within the Public Land Mobile Network (PLMN) is supported by a new logical network node called the GPRS Support Node.
(GSN). The Gateway GPRS Support Node (GGSN) acts as a logical interface to external packet data networks. It provides inter–working with external packet–switched networks, and is connected with SGSN via an IP–based GPRS backbone network. The Serving GPRS Support Node (SGSN) is responsible for the delivery of packets to the MSs within its service area. It is at the same hierarchical level as the GSM component Mobile Switching Center (MSC), which keeps track of the individual MSs’ location and performs security functions and access control. The SGSN is connected to the Base Station Subsystem (BSS) with Frame Relay. Within the GPRS network, Protocol Data Units (PDUs) are encapsulated at the originating GSN and decapsulated at the destination GSN. In between the GSNs, Internet Protocol (IP) is used as the backbone to transfer PDUs. This whole process is defined as tunneling in GPRS. The GGSN also maintains routing information used to tunnel the PDUs to the SGSN that is currently serving the MS. All GPRS user related data needed by the SGSN to perform the routing and data transfer functionality is stored within the HLR. Notice that the BSS is shared between GPRS and GSM network elements, to maintain compatibility and to keep low the investments needed to introduce the GPRS service. In fact, as far as the BSS subsystem is considered, the introduction of GPRS over an existent GSM network only requires a software upgrade.

Figure 2.2 shows a simple example of routing in a mobile originated transmission. In order to access GPRS services, an MS first makes its presence
known to the network by performing a GPRS attach. This operation establishes a logical link between the MS and the SGSN, and makes the MS available for SMS via GPRS, paging via SGSN and notification of incoming GPRS data. To send and receive GPRS data, the MS activates the packet data address it wants to use. This operation makes the MS known in the corresponding GGSN, and interworking with external data networks can commence. The serving SGSN of the source mobile (SGSN−S), encapsulates the packets transmitted by the MS and routes them to the appropriate GGSN (GGSN−S). Based on the examination of the destination address, packets are then routed to the GGSN-D through the packet data network. The GGSN-D checks the routing context associated with the destination address and determines the serving SGSN (SGSN−D) and relevant tunneling information. Each packet is then encapsulated and forwarded to the SGSN-D which delivers it to the destination mobile.

2.3.2 The Protocol Architecture

Figure 2.3 shows the transmission plane up to the network layer according to the ISO/OSI reference model. Above the network layer, widespread standardized protocols may be used. The selection of these protocols is outside of
the scope of the GPRS specification. Between two GSNs, the GPRS Tunnel Protocol (GTP) tunnels the PDUs through the GPRS backbone network by adding routing information.

Below the GTP, the Transmission Control Protocol/User Datagram Protocol (TCP/UDP) and the Internet Protocol (IP) are used as the GPRS backbone network layer protocols. Ethernet, ISDN, or ATM based protocols may be used for L2 (Data Link Control Layer) depending on the operator’s network architecture.

Between the SGSN and MS, the Subnetwork Dependent Convergence Protocol (SNDCP) maps network level protocol characteristics onto the underlying logical link control and provides functionalities like multiplexing of network layer messages onto a single virtual logical connection, encryption, segmentation and compression.

Radio communication between the MS and the GPRS network covers the physical and data link layer functionality. Between the MS and the BSS, the data link layer has been separated into two distinct sublayers: the Logical Link Control (LLC) and the Radio Link Control/Medium Access Control (RLC/MAC) sublayers. The LLC layer is the higher sublayer of the data link layer. It operates above the RLC/MAC layer and provides a logical link between the MS and SGSN.

The RLC/MAC layer provides services for information transfer over the physical layer of the GPRS radio interface. It defines the procedures that enable multiple MSs to share a common transmission medium which may consist of several physical channels.

The RLC layer is responsible for the transmission of data blocks across the air-interface and the Backward Error Correction (BEC) procedures consisting...
Analysis of Telecommunication Systems

of the selective retransmission of uncorrectable blocks (ARQ).

The MAC layer itself is derived from a slotted ALOHA protocol and operates between the MS and BTS. It is responsible for access signaling procedures for the radio channel governing the attempts to access the channel by the MSs, and the control of that access by the network side. It performs contention resolution between channel access attempts, arbitration between multiple service requests from different MSs, and medium allocation to individual users in response to service requests.

The physical layer (GSM RF in Figure 2.3) is split up into a Physical Link Sublayer (PLL) and a Physical RF Sublayer (RFL). The PLL provides services for information transfer over a physical channel between the MS and the network. These functions include data unit framing, data coding, and the detection and correction of physical medium transmission errors. The PLL uses the services of the Physical RFL. The PLL is responsible for:

- Forward Error Correction (FEC) coding allowing detection and correction of errors in transmitted codewords and the signaling of uncorrectable codewords.

- Rectangular interleaving of one Radio Block over four bursts in consecutive TDMA frames (as defined in GSM 05.03).

- Procedures for detecting physical link congestion.

The RFL is part of a complete GSM system that delivers a range of services including GPRS. The RFL performs the modulation and demodulation of the physical waveforms and conforms to the GSM 05 series recommendations which specify among other things:

- Carrier frequencies and GSM radio channel structures (GSM 05.02);

- Modulation of transmitted waveforms and the raw data rates of GSM channels (GSM 05.04);

- The transmitter and receiver characteristics and performance requirements (GSM 05.05).

In the network, the LLC is split between the BSS and SGSN. The BSS functionality is called LLC Relay. Between the BSS and SGSN, the BSS GPRS Protocol (BSSGP) conveys routing and QoS (Quality of Service) related information, and operates above Frame Relay.

Figure 2.4 shows the Radio Block Structures for user data and control. Each Radio Block consists of a MAC Header, a RLC Data Block or
RLC/MAC Control Block and a Block Check Sequence (BCS). It is always carried by four Normal Bursts.

The MAC Header consists of the Uplink State Flag (USF), the Block Type Indicator (T), and the Power Control (PC) fields. The RLC Data Block consists of the RLC Header and the RLC Data. The RLC/MAC Control Block contains the RLC/MAC signaling information elements.

Channel coding is currently specified in GSM 05.03. Four different coding schemes, CS-1 to CS-4, are defined for the Radio Blocks carrying RLC data blocks.

Figure 2.5 shows the coding procedure, where the payload is defined in Figure 2.4. The details of the codes are shown in Table 2.1.

### 2.3.3 GPRS Air Interface Protocol

The GPRS air-interface protocol [61] is concerned with communications between the MS and the BSS at the physical, the Medium Access Control (MAC) and Radio Link Control (RLC) protocol layers, indicated in the Figure 2.3. The RLC/MAC sublayers allow efficient multiuser multiplexing on the shared data channel(s) and utilize a selective ARQ protocol for reliable

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Code Rate</th>
<th>Payload</th>
<th>Data Rate (Kb/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS-1</td>
<td>1/2</td>
<td>184</td>
<td>9.05</td>
</tr>
<tr>
<td>CS-2</td>
<td>≈ 2/3</td>
<td>268</td>
<td>13.4</td>
</tr>
<tr>
<td>CS-3</td>
<td>≈ 3/4</td>
<td>312</td>
<td>15.6</td>
</tr>
<tr>
<td>CS-4</td>
<td>1</td>
<td>428</td>
<td>21.4</td>
</tr>
</tbody>
</table>

Table 2.1: GPRS coding schemes
transmissions across the air-interface.

The physical channel dedicated to packet data traffic is called a *Packet Data Channel* (PDCH). A cell which supports GPRS may allocate one or more shared PDCHs which are taken from the common pool of physical channels available to the cell and otherwise used as *Traffic Channels* (TCHs). The allocation of TCHs and PDCHs is done dynamically according to the “capacity on demand” principles described below.

**Master–Slave Concept**

At least one PDCH (mapped on one physical time slot), acting as a master, accommodates *Packet Common Control Channels* (PCCCHs) that carry all necessary control signaling for initiating packet transfer as well as user data and dedicated signaling. The others, acting as slaves, are only used for user data transfer.

**Capacity on Demand**

In order to allow GPRS service in cells where there are few (or no) GPRS users without the need for any permanently allocated resources, the concept of capacity on demand has been introduced. The operator can decide whether to dedicate some PDCHs for GPRS traffic. Load supervision is done in the MAC layer to monitor the load on the PDCH(s), and the number of allocated PDCHs in a cell can be increased or decreased according to demand. Unused channels can be allocated as PDCHs to increase the overall quality of service for GPRS. If other services with higher priority request resources, de-allocation of PDCHs can take place. However, the existence of PDCH(s) does not imply the existence of PCCCH. When no PCCCH is allocated in a cell, all GPRS attached MSs automatically camp on the existing GSM CCCH as they do in the idle state. When a PCCCH is allocated in a cell, all GPRS attached MSs camp on it. The PCCCH can be allocated either as the result of the increased demand for packet data transfer or whenever there are enough available physical channels in a cell. If the network releases the PCCCH, the MSs return to the CCCH.

**Multiframe Structure for PDCH**

A multiframe structure is needed for the PDCH in order to accommodate paging groups and possibly blocks for broadcasting of the GPRS system information. The multiframe structure of both 51 TDMA frames and 52 TDMA frames are specified in GSM 05.01.
2.3.4 Data Flow

The Network layer Protocol Data Units (NPDUs or packets) which are received from the network layer are transmitted across the air-interface between the MS and the SGSN using the LLC protocol. First, the SNDCP transforms packets into LLC frames. The process includes optional header/data compression, segmentation and encryption. The maximum amount of user data in an LLC frame is 1600 Bytes.

An LLC frame is then segmented into RLC Data blocks which are formatted into the physical layer. Each block comprises 4 normal bursts in consecutive TDMA frames. Figure 2.6 summarizes the data flow in GPRS.

GPRS Logical Channels

Table 2.2 lists the GPRS logical channels and their functions. A detailed description for each channel is presented as follows.

Packet Broadcast Control CHannel (PBCCH)

The Packet Broadcast Control CHannel transmits system information to all GPRS terminals in a cell.
### Packet Common Control Channels (PCCCH)

The **Packet Random Access Channel** is used by MSs to initiate packet transfers or to respond to paging messages. On this channel MSs transmit access bursts with long guard times. On receiving access bursts, the BSS assigns a Timing Advance to each terminal.

The **Packet Paging Channel** is used to page an MS prior to downlink packet transfer.

The **Packet Access Grant Channel** is used in the packet transfer establishment phase to send resource assignment to an MS prior to the packet transfer.

The **Packet Notification Channel** is used to send a Point to Multipoint-Multicast (PTM-M) notification to a group of MSs prior to a PTM-M packet transfer. The notification has the form of a resource assignment for the packet transfer.

### Packet Traffic Channels (PDTCH)

The **Packet Data Transfer Channel** is a channel allocated for data transfer. One MS may use more than one PDTCH in parallel (multislot operation) for individual packet transfers.

The **Packet Associated Control Channel** is used to convey signaling information related to a given MS such as Acknowledgments (ACK), and Power Control information (PC). It also carries resource assignment and re-assignment messages, either for the allocation of a PDTCH or for further occurrences of a PACCH. One PACCH is associated to one or several PDTCHs that are concurrently assigned to one MS.
2.3.5 The Random Access Procedure

As it is observed during massive congestion events, such as New Year's Eve, the blocking on the PRACH (e.g. during the contention for channel reservation) may become a bottleneck of the system. This is the specific aspect of the GPRS addressed in the remainder of this dissertation, which therefore deserves a more detailed description \[62\].

The MSs get the access control parameters by listening to the Packet Broadcast Control Channel (PBCCH). Such parameters are the number of maximum retransmissions $M$, the persistence level $P$ and the parameters $S$ and $T$. The MS is allowed to make a maximum of $M + 1$ attempts to send a Packet Channel Request message. At the beginning of the procedure a timer is set (to 5 sec). At the expiry of this timer, the procedure, if still active, is aborted and a failure is indicated to the upper layer. The first attempt to send a Packet Channel Request can be initiated at the first possible TDMA frame containing PRACH. For each attempt, the mobile station extracts a random value $R$, and only if $R$ is bigger than, or equal to, the persistence level $P$ the station is allowed to send a Packet Channel Request. After a request is issued, the MS waits for a time, dependent on $S$ and $T$. If it does not receive the Packet Downlink Assignment (or a Packet Queuing) a new attempt is tried, if it is still allowed to make one, otherwise a failure is notified to the upper layer. From parameters $S$ and $T$, the MS also determines the next TDMA frame in which a new attempt is possible, should the previous one be unsuccessful and a new attempt still allowed. Under normal workload conditions, this retry mechanism is able to make the MS request to reach the BSS subsystem with a very high probability. Once the MS request successfully reaches the BSS subsystem, traffic packet data channels, called slave PDCH, are allocated if available in the cell to transport users’ data and transmission signaling. For what concerns data transfer, uplink and downlink channels allocation is completely independent and a MS can operate uplink and downlink data transfer simultaneously. Should the selected cell be not immediately able to allocate the PDCHs, the MS request may be put in a queue to wait for the first available resources. In case the request cannot be accommodated, a reject message is sent to the MS.

2.4 Database Maintenance

Databases supporting telecommunication systems, such as the previously introduced HLR and VLR used by GPRS, contain persistent and temporary
information needed in operations and management of the networks and its services. Such databases exhibit stringent reliability, availability and performance requirements since they store system-related as well as clients-related information, and provide basic services to the application process, such as read, write and search operations. Data concerning the status, the access rights and features available to the users, and routing information for dispatch calls are all examples of data contained in the database. The database is subject to corruption determined by a variety of hardware and/or software faults, such as internal bugs and transient hardware faults. The occurrences of such faults have the potential of yielding to service unavailability. Because of the central role played by such database in assuring a correct service to clients, means to pursue the integrity/correctness of data have to be carried out.

Actually, to ensure the consistency of database subsystems involved in communication systems (e.g., telephone systems), appropriate scheduled maintenance policies are necessary. Audit operations, consisting in periodic checks and recovery actions, are typically employed in databases to cope with runtime faults which may affect the dependability and quality of service of the overall system. With the term data audit it is commonly indicated a broad range of techniques to detect errors and recover from them. The kind of checks performed on the data to test its correctness highly depends on the specific application at hand, on the system components, and environmental conditions which determine the expected fault model. Both commercial off-the-shelf and proprietary database systems are generally equipped with utilities to perform data audits, (e.g. see [64]).

Our study on database maintenance aims at investigating on appropriate tuning of audit operations, so as to find optimal balances between contrasting requirements, namely satisfactory database availability and low overhead due to audits.

2.5 Conclusions

This Chapter focused on the aspects of GPRS system on which this dissertation is based on. The remainder of this thesis is split in two main topics. In the first one, Chapter 3 describes our GPRS analysis without outages, evaluating performance measures both at network and user levels. Chapter 4 characterizes the GPRS behavior in presence of outages while a dependability analysis is carried out in Chapter 5. Concerning the second topic, Chapter 6 introduces the models for database maintenance analysis and their solution is carried out in Chapter 7.
Part 2
Chapter 3

GPRS Analysis without Outages

3.1 Introduction

In this Chapter an analysis of the GPRS in absence of outages is presented. Thus, measures of QoS and performance are evaluated. At first, the focus is only the random access procedure, during which users compete to get a free channel. Its solution gives some QoS indicators at network (packet) level. Actually, initially users are supposed to send simple requests, composed just by one packet, without any other constraint. Afterwards, our analysis is enriched; the random access model is joined with a refined user model to account for a more complex user behavior and we obtained how such indicators at network level affects the QoS perceived by users. Also, our methodology defines a way to compose the previous defined models.

All models are derived using Stochastic Activity Networks (SAN) [42] and solved using the simulator provided by the UltraSAN tool [22]. The nature of the measures and the order of magnitude of the results we are looking for, make a simulation approach appropriate for studying the system. At the same time, we could represent real system conditions better than by using analytical approaches (we could choose distribution functions resembling the occurrence of specific phenomena, and not be forced to the exponential distribution). Moreover, using graphical primitives, SANs provide a high level modeling formalism with which detailed performance, dependability, and performability models can be specified relatively easy. Throughout this dissertation, numerical evaluations have been determined with 95% confidence interval and a relative confidence interval of 5% (if not differently specified).
3.2 Analyzing QoS at Network Level

We present here the evaluation of performance indicators of the GPRS by analyzing its behavior during the contention phase where users compete for channel reservation. In this analysis, we suppose that each user follows an elementary behavior, making requests that fit just one LLC frame. The following Sections present the measure selected for our evaluation study, the SAN model and the scenarios considered.

3.2.1 Measures of Interest

We have identified three measures to characterize the expected performance of GPRS systems during the contention phase for channel reservation. They are:

1. The probability that a user request is not successful, $P_{block}$;
2. The average time $D$ between two successful requests made by the generic user;
3. The average number of busy channels in the system, $C_{busy}$.

The first two can be regarded as performance indicators with a direct relation with the QoS as perceived by the user. The third measure is of major interest of the service supplier to balance between costs to be afforded and user satisfaction. Evaluation of such indicators has been performed through a steady-state analysis to get values indicative of the average behavior of the system.

In the following, the random access procedure model is defined under the assumption concerning the configuration of the GPRS stated in Section 3.2.2. The derived model is described in Section 3.2.3.

3.2.2 Modeling Assumptions

The random access procedure model has been defined under the following assumptions concerning the configuration of the GPRS:

1. Only one cell has been taken into account, containing a constant number of users, whose contexts are permanently retained (no “attach” and “detach” procedures are considered in our study);
2. All users belong to the same priority class, they are indistinguishable from the point of view of generated traffic;
3. User requests fit in one LLC frame and, from the user’s viewpoint, once a request has been made, he cannot abort it but has to wait until the service is provided;

4. The radio channel is considered faultless, meaning that no retransmissions are necessary at the LLC and RLC levels. To keep consistent, the coding scheme considered is the CS−1, characterized by a 1/2 code rate, payload of 184 bit per RLC block. This is the most robust coding scheme among the four accounted for by the standard;

5. One radio frequency is at maximum devoted to the GPRS traffic (8 time slots);

6. Only one MPDCH, for signaling and control information, is assumed, carrying 1, 2, or 4 PRACHs;

7. Each traffic channel is allocated to a single user at a time, who will retain it until the completion of his data transmission;

8. It is allowed to queue the request at BTS side through an Access Grant Reservation when all channels are occupied.

3.2.3 The SAN Model

The model of the random access procedure is shown in Figure 3.1 and its description is briefly sketched:

- Tokens in the place Served represent those users that have sent successfully their up−link data. After some time, accounted for by the timed transition to req, a user issues a new request and a token is moved from Served to the place nup.

- The block starting with the instantaneous activity req and ending with the input gate control represents the dynamics of the random access procedure. req states the maximum number of attempts a user is allowed to make in sending an Access Burst. It has one case for each possibility; the associated probabilities have been derived on the basis of the parameters M, P, S, T and the timer. Tokens in places ready1, ..., ready8 represent the number of users allowed to make a maximum of 1, ..., 8 attempts, respectively. The instantaneous activities check_p1, ..., check_p8 model the persistence level. If the user passes the persistence level, he can send an Access Burst and moves into the place tryi, otherwise he moves into the correspondent place faili.
Figure 3.1: SAN model of the Random Access Procedure of the GPRS system
Should a user consume all his assigned attempts to make his request, or should the time-out regulating the maximum allowed time for making a request (set to 5 sec) expire, the user is moved into the place block. A blocked user will do a new attempt after a time sampled from the timed activity $b_{to\_n}$, having exponential rate and taking into account Automatic Retransmission Time (ART). The place $w5$ and the activity $wait\_5$ take into account those users that haven’t been assigned any attempt, because they will always fail the persistence level. According to the standard specification, they have to wait 5 seconds before moving in the place block.

- The instantaneous transition $check\_capture$ checks, stochastically, if there is a successful receipt of one Access Burst; if yes, a token is placed in $one\_accepted$, unless the queue is full and there is no available traffic channel, in which case a token is put in $all\_discarded$. The instantaneous transition $who\_is\_passed$ fires when there is a token in $one\_accepted$ and it allows to choose which level the accepted Access Burst comes from, placing a token in one of the places $p1$, ..., $p8$ (each Access Burst at each level has the same probability to be the accepted one). The input gate $control$ and the activity $control\_act$ properly update the places recording the residual tries made available to the other concurrent requests (places $ready1$, ..., $ready8$, $try1$, ..., $try8$, $fail1$, ..., $fail8$, $wait\_a0$, ..., $wait\_a7$ and $p1$, ..., $p8$).

- When there is a successful receipt of one Access Burst and there is a free channel (that is at least a free pair between $ch1\_a1$, ..., $ch7\_a7$), the output gate $choose\_channel$ puts a token in one of the places $ch1$, ..., $ch7$ otherwise it puts a token in the place $queue$. The timed activities $su1$, ..., $su8$ simulate the set-up time of a radio link to send user data. The timed activities $exp1$, ..., $exp7$ and $d1$, ..., $d7$ simulate the data sending time in case it follows a truncated exponential distribution. Otherwise those activities are replaced by transitions $u1$, ..., $u7$ (e.g. in Figure 3.11) if we assume that the data sending time follows a uniform distribution. Their actual values depend on the scenario considered.

- The sub-net enclosing the timed activities $PRACH\_available$ and $slot\_available$, and the places $en$ and $enable$, models the multiframe on the MPDCH.

- A token in the place $queue$ represents a pending request waiting for up-link channel reservation. The immediate transition $q\_control\_a$ fires when a channel is released and there are pending requests in the queue.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-PDCH</td>
<td># Slave Packet Data CHannels</td>
<td>1...7</td>
</tr>
<tr>
<td>Users</td>
<td># Users in the cell</td>
<td>10...190</td>
</tr>
<tr>
<td>PRACH</td>
<td># Packet Random Access CHannel</td>
<td>1, 2, 4</td>
</tr>
<tr>
<td>$T_{req}$</td>
<td>Time interval between two request</td>
<td>10-76.4 sec</td>
</tr>
<tr>
<td>$P_{length}$</td>
<td>Data packet length</td>
<td>Trunc. Exponential, Uniform</td>
</tr>
</tbody>
</table>

Table 3.1: Relevant parameters and their default values

When transition $q_{control\_a}$ fires, the input gate $q_{control}$ moves a token from queue to a place $chn (ch1, ch2, ..., ch7)$, corresponding to the available channel;

A detailed description of the case probability of the instantaneous transition $req$ and the dimensioning of the queue length can be found in [65].

### 3.2.4 Settings for Numerical Evaluation

We identified two representative scenarios with different workload characteristics. In Scenario 1, the traffic model used is the Railway model, according to the ETSI document for evaluation criteria [66]. In this model, users’ data are generated using a truncated negative exponential distribution, with an average of almost 170 bytes and maximum value of 1000 bytes. On average, users make requests for data transmission every 10 seconds. This scenario seems to be adequate for Web browsing applications. In the second scenario, data traffic is generated using a uniform distribution in the interval [1000, 1600] bytes (note that an LLC frame may contain up to 1600 bytes; for the sake of simplicity, we limited data packets to this maximum size to require only one successful random access to complete the transfer of a user request). On average, users issue requests every 76.4 seconds. In Scenario 2 requests are less frequent than in the previous one, but each request involves more data to be transferred such as in case of short email or filled forms in WEB applications.

### 3.2.5 Numerical Evaluation

Table 3.1 summarizes the main parameters of the system taken into account in the evaluation, together with the default values used in the subsequent numerical analysis (unless otherwise specified). System parameters
P, T and S have been assigned average values in the respective intervals, as defined by the standard, while M has been assigned the maximum value (i.e., 7).

The selected indicators are derived from the SAN model as follows:

1. $P_{\text{block}}$ is determined as the ratio between the rate of the transition $b\_to\_n$ and the sum of the rates of $b\_to\_n$ and $t_{\text{req}}$;

2. $C_{\text{busy}}$ is the sum of the marking of the places $chi$ and $ai$ with $i = 1,...,7$;

3. $D$ is computed as the ratio between the number of the served users (number of tokens in the place $Served$) and the rate of the transition $to\_req$.

**A First Scenario**

The first study relative to scenario 1 concerns the evaluation of $P_{\text{block}}$ and $C_{\text{busy}}$ at varying values of users and traffic channels (SPDCH, from 1 to 7), keeping fixed the number of PRACHs to 2. In this scenario it is important to consider that the average time occupation of a channel after a successful request is shorter than the time between two successive PRACHs.

Figure 3.2 illustrates the obtained results. It can be immediately observed that using a higher number of SPDCHs always improves $P_{\text{block}}$; however,
while the improvement is very sensible for low numbers of SPDCH, when such number exceeds 4, it can be noticed only for a high number of users. In fact, a higher availability of traffic channels does not bring any significant advantage as long as the number of PRACH per multiframe does not grow (thus becoming a critical point for accepting user requests). The effect on $C_{busy}$ is slightly different (see Figure 3.3). The overall influence of the growing number of SPDCH has the same trend as for $P_{block}$, but the improvement, even for low numbers of SPDCH, tends to decrease towards the right part of the figure, starting from a number of users which depends on the available SPDCH. This is again due to the limiting effect of the small number of available PRACHs, which raises the probability of collision on user requests when the number of users becomes too big.

Figure 3.4 and Figure 3.5 illustrates the behavior of $P_{block}$ and $C_{busy}$ at varying values of the number of users with 1, 2 and 4 PRACHs and 1 and 3 SPDCHs. Looking at the curves relative to $P_{block}$, it can be appreciated the combined positive effects of having an higher number of SPDCHs and PRACHs in satisfying a growing number of users. The plots relative to $C_{busy}$ make more evident than in the previous analysis the need of a proper balance between the number of SPDCHs and the number of PRACHs. Therefore, the message for the service supplier is that employing a higher number of traffic channels to better satisfy a higher number of user requests is worthwhile only if an adequate number of PRACHs for the random access competition is also
Figure 3.4: Plots of $P_{\text{block}}$ in scenario 1 with SPDCH = 1 (a), and SPDCH = 3 (b)

Figure 3.5: Plots of $C_{\text{busy}}$ in scenario 1 with SPDCH = 1 (a), and SPDCH = 3 (b)
Figure 3.6: Plots of $D$ in scenario 1 with PRACH = 1 (a), and PRACH = 2 (b)

made available.

Figure 3.6, relative to the study of $D$, completes the analysis relative to the first scenario. In the figure, the influence of using a different number of traffic channels is studied for two configurations of PRACH (1 and 2). The first observation is that, as expected, the values of $D$ grow at increasing values of the number of users. Second, it can be immediately appreciated the high influence of the number of PRACHs, while using more than 2 traffic channels does not bring any significant advantage. As for the previous analysis, especially that related with $P_{\text{block}}$, the short service duration and the relatively high service request rate considered in this scenario are responsible for the higher influence of PRACHs over SPDCHs.

A Second Scenario

With respect to scenario 1, scenario 2 is characterized by less frequent user requests, but each sending a larger amount of data. The results of the analysis conducted on scenario 2 are illustrated in Figures 3.7, 3.8 and 3.9. On all the three measures, $P_{\text{block}}$, $C_{\text{busy}}$, and $D$, the influence of the number of used PRACHs is smaller whereas the most significant parameter is the number of traffic channels. The rationale behind is the different kind of user workload considered here: the bigger amount of data to transfer with the consequent longer occupation of the channels makes channels more precious than PRACHs. In general, the higher the number of traffic channels used, and the better are the measured figures. Figure 6(a) shows that moving from 1 to 3 SPDCHs makes a huge difference (e.g., $P_{\text{block}}$ changes from 0.9 to almost 0 in case of 100 users) while having 1, 2 or 4 PRACHs is almost irrelevant. Figures 3.7, 3.8 and 3.9 show the plots relative to $C_{\text{busy}}$, and $D$.
Figure 3.7: Plots of $P_{\text{block}}$ in scenario 2 with SPDCH = 1 (a), and SPDCH = 2 (b) respectively. In Figure 3.8 the value of PRACH is 2, but very similar results have been obtained also in case of PRACH = 1 and PRACH = 4. Both figures show that employing a number of traffic channels higher than 3 does not bring any further improvement.

### 3.3 Analyzing QoS of GPRS from a User’s Perspective

This work contributes to the analysis on the service accomplishment level perceived by GPRS users. The proposed modeling approach builds separately the GPRS and user models; the focus is on the GPRS access random procedure on one side, and different classes of users’ behavior on the other side. The overall model is obtained composing the basic submodels. Quantitative analysis, performed using a simulation approach, is carried out, showing the impact of user’s characteristics and network load on identified indicators expressing the QoS as perceived by users. Two main user characteristics neglected so far are accounted for here:

- The traffic generated by users is heterogeneous from the point of view of the requests size; specifically, user requests may require a varying number of LLC frames to be sent. This way, the representation of a population of users requesting service of different kinds is possible (e.g., web–browsing, e–mail);

- In its general form, GPRS may perform a varying number of automatic retries to provide the service, in case the user request fails the channel contention procedure or finds all the data channels busy. A simplistic
CHAPTER 3

Figure 3.8: Plots of $C_{busy}$ in scenario 2

Figure 3.9: Plots of $D$ in scenario 2 with PRACH = 1 (a), and PRACH = 2 (b)
case is to assume that such automatic retries are carried on until the service is provided. In more realistic scenarios, a user sets an a-priori maximum time interval he is prepared to wait for having his request accepted (user timeout), after which the user deliberately gives up.

In addition, a threshold mechanism is introduced to cope with congestion phenomenon. For each ongoing request transmission, at a pre-defined time instant an intermediate check on the speed of the network is made. If the request part still to be sent is bigger than a pre-defined amount, that request is aborted: the chance for it to be completed on time is considered too low, and the resources would be better used for completing other concurrent requests.

3.3.1 Measures of Interest

The four measures selected for our evaluation study account for important aspects of the QoS offered by the network as perceived by users. They are:

1. The mass probability of the delay incurred by a user request;
2. The average delay incurred by a user request;
3. The probability that a request fails to be satisfied within the user timeout (timeout probability);
4. The total probability of transmission failure due to both exceeding the user timeout and not satisfying the threshold condition, in case the threshold mechanism is applied (failure probability).

Of course, in order to estimate the overall user satisfaction on the service provided, other factors such as pricing, refund policy, repair time, etc., have to be taken into account as well.

3.3.2 Modeling Assumptions

In particular, the assumptions 2 and 3 on the simple user behavior made modeling only the random access procedure of Section 3.2.2 are relaxed here introducing a more realistic and detailed model of the user behavior. The following assumptions hold with respect to the user model:

- Following the standard, a user has to make a number of random accesses to the network equal to the number of LLC frames necessary to complete the request transmission. We assume that, in case of multiple LLC requests, the user can try the random access for sending the
next LLC frame only after having received the acknowledgement on the successful transmission of the previous LLC frame;

- Users are partitioned in classes; inside the same class, users issue requests of the same size, with the same inter-request rate, and apply the same user timeout;

- Should a user timeout elapse before completing the request transmission, that user waits for his next inter-request time instant for issuing a new request

### 3.3.3 Overall Model

As stated before, our modeling approach consists in combining the network model and several user models, representing the random access to the network and the user behavior, respectively. The overall model, shown in Figure 3.10, is obtained from the sub-models defined above by joining them through the UltraSAN Join and Rep operators. In particular, different classes of users can be modeled (User1, ..., UserN) and the overall user population is reached replicating each user model by means of the Rep operator.

A detailed description of the random access procedure (Network, in Figure 3.10) has been already given in Section 3.2.3. The previous model allowed making analysis of the channel contention phase of GPRS assuming a simple user behavior.

In order to account for the new user characteristics described before, we followed a two-step approach in developing the overall model. The modeling
approach consists in combining the random access procedure model described in Section 3.2.3 and the users’ models. In order to be combined with the users’ models, the model of Figure 3.1 has been slightly modified. Such modifications concern how the places **Served** and **block** are connected to the rest of the network and a new place called **new** is introduced. Such places became common places between all models. In this way, it is possible to join them through the UltraSAN Join operator. Another difference is in the part of the random access procedure model devoted to model the transmission of an LLC frame over the air interface.

Actually, if we want to allow users in a single session to transmit more data than one LLC frame can carry on, we need to modify the random access model; the new places **am1**, ..., **am7** and the transitions **Max1**, ..., **Max7** are added, as shown in Figure 3.11. When the amount of data to transmit in a given session exceeds one LLC frame space, data are split on more LLC frames. Thus, all the LLC frames to be sent are full LLC frames except for the last one that must hold a random amount of data with a given distribution. We also assume that the last LLC frame to be sent follows an uniform distribution from a minimum to a maximum value (lesser than the dimension of an LLC frame). This distribution is specified in the random access model by means of transitions **u1**, ..., **u7**. Thus, the random access model has to recognize, on statistical bases, if the channel is used to transmit a full LLC frames (which takes, in an error free environment, a deterministic time) or the latest of series (which transmission time depends on the chosen distribution). This choice is taken by means of the case probability of transitions **su1**, ..., **su7**. The probability to transmit a full LLC frame is evaluated as follows. For example, let suppose that an user is allowed to transmit up to 3 LLC frames per session with the probability of sending 1, 2 or 3 LLC frames of $P_{1\text{LLC}} = 0.5$, $P_{2\text{LLC}} = 0.3$, and $P_{3\text{LLC}} = 0.2$, respectively. The probability to transmit a full LLC frame is:

$$P_{\text{full LLC}} = P\{P_{\text{full LLC}}/ 1 \text{ LLC to be sent}\} \times P_{1\text{LLC}} + P\{P_{\text{full LLC}}/ 2 \text{ LLC to be sent}\} \times P_{2\text{LLC}} + P\{P_{\text{full LLC}}/ 3 \text{ LLC to be sent}\} \times P_{3\text{LLC}}.$$  

For the example above $P_{\text{full LLC}} = 0*0.5+0.5*0.3+0.66*0.2$, because of $P\{P_{\text{full LLC}}/ 1 \text{ LLC to be sent}\}$ is always zero, $P\{P_{\text{full LLC}}/ 2 \text{ LLC to be sent}\} = 0.5$, and $P\{P_{\text{full LLC}}/ 3 \text{ LLC to be sent}\} = 2/3 = 0.666$.

Figure 3.12 shows the model which represents the behavior of a single user. It is configurable, so as to adapt to different classes of users, by changing the inter-request time, the number of LLC frames composing each user request and the user timeout. The model description is the following:
CHAPTER 3

Figure 3.11: Modified random access model

Figure 3.12: User model
A token in the place *idle* means that the user has no pending service requests. The exponential transition *request* represents the user idle time. The users not in the idle state are allowed to take a token from place *Served* or place *block*.

The exponential transitions *catch* and *catch2* allow to catch a token from the places *Served* (LLC frame successfully transmitted) or *block* (LLC frame not successfully transmitted). The transitions *finish* and *finish2* and places *wait* and *wait2* allow introducing a delay in catching a token from the places *Served* and *block*, respectively: a user can not take a token from place *Served* until the place *wait* is empty (when transition *finish* fires). In the same way, a user can not take a token from place *block* until the place *wait2* is empty.

Input gates *contr* and *contr2* synchronize all the operations to catch the tokens from places *Served* and *block*. The gate *contr* also controls if the user has finished to transmit all his pending tokens to complete his request (traced in the place *pending*); should this be the case, the user is put back into the idle state, by adding a token in the place *idle*.

The output gate *send* puts in the place *pending* a number of tokens corresponding to the number of LLC frames the user request consists of.

The subnet including *start*, *no_more*, *count_one*, and *acc_count* represents a timer. Such timer keeps track of the time requirements set by the user; should such timer expire without completing the request sending, the residual operations relative to that request are aborted and the user is put back in the place *idle*. Note that the user who fails to have his request successfully transmitted because of the deadline violation, does not know if the failure was due to continuous collisions with other concurrent requests or because the system is experiencing a possible outage. The gate *no_more* also controls the threshold mechanism conditions, when such mechanism is employed.

The places *idle_app*, *abort*, and *abort_app* and the transition *abort* are exclusively used for measuring purposes, as specified later on.

### 3.3.4 Settings for the Numerical Evaluation

The model in Figure 3.10 has been numerically solved using the simulator provided by the UltraSAN tool [22]. In the following experiments, GPRS
configuration are kept fixed. We take advantage of the work of the previous Section for the proper identification of a satisfactory GPRS configuration in terms of PRACHs and PDCHs. Precisely, we consider a GPRS configuration consisting of 5 SPDCH (i.e., traffic channels) and 3 PRACH (i.e., logical access channels). Two system scenarios have been identified and studied. In the first scenario, the users’ population is partitioned into two groups, with different characteristics of the workload offered. Specifically, the first group issue requests fitting in 1 LLC frame (i.e., consisting of up to 1600 bytes), with a short inter-request time, while requests issued by the other group are composed of 6 LLC frames, with a longer inter-request time. The two groups also differ in the maximum time a user is prepared to wait to have his request satisfied before giving up. Table 3.2 summarizes the numerical values assigned to such user characteristics for the two groups. The total load offered to the system is variable from 75% to 90% of the whole system capacity. Different compositions of the load in terms of users belonging to Group \(_1\) and Group \(_2\) are considered, going from a load made only by users of Group \(_1\) or of Group \(_2\), to some intermediate proportions between both of them. Consequently, the number of users requesting network services varies in accordance with the considered total load and the considered composition of users.

An explanation of how the workload offered to the network is tied to some parameters of the random access model and to the number of users for each group is given in the following. These considerations are still valid for other evaluations throughout this dissertation. In the model of the random access procedure, the traffic capacity for a channel is evaluated dividing the length (byte) of one LLC frame, 1600 byte, by the time to transmit it (e.g. 1.29248 sec, by using the CS-1 code scheme plus the time to setup the channel 0.036928 sec. (equals to 8 TDMA frames) [65]). Thus, the traffic capacity for each channel is: 1600/(1.29248+0.03928) = 1201.42 byte/sec.

Let suppose that users are partitioned in classes and users belonging to the same class make requests of the same size with the same rate. Thus, the

<table>
<thead>
<tr>
<th>Group</th>
<th>Inter_request_time</th>
<th>Request size (in LLC frames)</th>
<th>Timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>(_1)</td>
<td>10</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>(_2)</td>
<td>120</td>
<td>6</td>
<td>18</td>
</tr>
</tbody>
</table>

Table 3.2: User parameters
input workload offered to the network due to a generic class of users is:

\[
\text{NumberOfUsers} \times \frac{\text{MeanNumberOfBytes}}{\text{InterRequestTime}}
\]

where \(\text{NumberOfUsers}\) is the number of users belonging to the same class, \(\text{MeanNumberOfBytes}\) is the mean size (in byte) of a user session, and \(\text{InterRequestTime}\) is the rate of the requests for such class of users. From the previous considerations can be easily achieved Table 3.3 which reports the further settings for numerical evaluation shown in the following Section.

The second scenario considers only users belonging to Group\(_2\), in the case of total load of 90% and 110% of the network capacity. Now, the attention is moved to the analysis of the effect of a threshold mechanism, introduced to control the congestion phenomenon. Associated to each user there is a couple \((T_{thr}, P_{thr})\), \(T_{thr}\) is the time, after the issuing of a request in which a check is made on whether at most \(P_{thr}\) packets have to be still delivered by the network in order for that user to maintain connected. The rationale behind such an intermediate check is aimed at reducing the system congestion. If the chance to complete the transmission of a request by the user’s required time is too low, it is better to abort that request, thus diminishing the competition on system resources (and therefore raising the chance of completing other currently active transmissions). For this scenario, we concentrate on a sensitivity analysis to find good values to assign to the couple \((T_{thr}, P_{thr})\). The above discussed values, assigned to the parameters in the experiments, are not taken from any specific application, although requests fitting in 1 LLC frame could be representative of typical Web browsing appli-
cations while the longer ones (fitting in 6 LLC frames) could well represent short/medium emails or filled forms on the web. Actually, the aim of this numerical analysis is to show the utility of our approach and to point out trends of the evaluated measures in presence of heterogeneous user behavior, rather than providing absolute QoS values.

3.3.5 Numerical Evaluation

This Section presents the numerical results obtained solving the SAN model with steady state analysis. The measures of interest are derived from the SAN model as follows:

- The mass probability function of the delay incurred by a user request is evaluated by means of the counter implemented by the SAN elements $\text{start}$, $\text{no\_more}$, $\text{count\_one}$, and $\text{acc\_count}$ of Figure 3.12; we break up the range of the values covered by the counter in disjoint adjacent intervals and we measure the proportion of time the value of the counter falls in such intervals deriving the wanted histogram. Place $\text{idle\_app}$ serves to measure only the delay of the user requests successfully ended.

- The average delay incurred by a user request is straightforwardly obtained from the mass probability function of the delay.

- The timeout probability is given by the probability that the previously described counter reaches the user timeout.

- The failure probability is obtained by the fraction of the number of times the transition $\text{abort}$ fires (e.g. transmission fails due to both exceeding the user timeout and not satisfying the threshold condition, in case the threshold condition is applied) on the number of times transition $\text{request}$ fires in given amount of time, e.g. one hour.

A First Scenario

The analysis in the first scenario concentrates on the delay incurred by a request to be successfully transmitted and on the probability to abort the request because of exceeding the user’s deadline. The mass probability distribution of such delay for users belonging to $\text{Group}_1$ and $\text{Group}_2$ are shown in Figure 3.13 and 3.14, respectively. Different combinations of the two kinds of users, including the extreme cases are considered. The plots in the Figure are obtained interpolating the histograms points determined with a step of 0.25 seconds in Figure 3.13 and of 1 second in Figure 3.14.
Figure 3.13: Delay distribution for users belonging to Group_1

Figure 3.14: Delay distribution for users belonging to Group_2
Figures 3.15 and 3.16 plot the probability of unsuccessful transmission of the user request by the deadline imposed by the user (timeout probability), and the mean transmission delay, respectively, at varying the percentage of users belonging to Group 2, for two different settings of the network total load. Looking at the curves relative to users of Group 1, it can be observed that the presence of users of Group 2 does not impact significantly the measures under evaluation. In fact, some appreciable variation is observed only in the first part of Figure 3.13. The probability of completing the request in the interval [1-1.25] seconds decreases by 20% from the case where the whole load is made of Group 1 users to that where Group 1 users account for 25% of the load. In Figures 3.15 and 3.16 the variations at changing the percentage of users of Group 2 are quite negligible.

Moving to the curves relative to Group 2, the effects of having a mixed load with users belonging to the two groups are a bit more noticeable. This is especially true in Figure 3.13, in particular for the 75% system load, where the worst values are obtained when the load is made only of Group 2 users. The impact of considering different values of system workload is very intuitive for both kinds of users: the higher is the workload and the higher is the probability of not completing the transmission by the imposed deadline. Similar trend shows in Figure 3.14.

Not surprisingly, the service perceived by Group 2 users are worst than for the other group. The rationale behind this observed behavior is easy to
understand. In fact, traffic generated by users in Group 2 is bursty (each request consists of 6 LLC frames), and several studies in the literature have already observed that the correlation among packets in such bursty cases have negative effects on the network performance e.g., [67].

**A Second scenario**

As already discussed, in this scenario we concentrate the attention on the users belonging to Group 2 only, to investigate the impact on the users perceived QoS of a threshold mechanism to perform intermediate checks on how quickly/slowly the transmission is going. If by time $T_{thr}$ more than $P_{thr}$ packets have still to be sent in order to complete the transmission, that request transmission is aborted, thus favoring the timely transmission of other concurrent requests. How to find good values for $(T_{thr}, P_{thr})$, so as to trade between avoiding the abort of requests with good chance to be completed by the deadline and not to wait too long to decide that a request will be unsuccessful, is not an easy task. Here, we conduct a sensitivity analysis on how to find a good balance, so as the overall QoS is improved.

Figure 3.17 plots the mass probability function of the delay incurred by a user request when the load is 90%, the time threshold $T_{thr}$ is fixed to 11 seconds, for different choices of the packet threshold $P_{thr}$. It gives an indication on how the sending of the remaining packets is accomplished, for
Figure 3.17: Threshold’s effect on the mass probability function of the service time.

Figure 3.18: Failure Probability and Delay at varying of $P_{th}$ under 90% load.
Figure 3.19: Failure Probability and Delay at varying of $T_{thr}$ under 90% load

those requests in which at most $P_{thr}$ packets have still to be sent at time $T_{thr}$. Figure 3.18 shows the failure probability and the mean time delay of a user’s connection when the time threshold is kept fixed and the packet threshold varies. The straight lines represent the reference values obtained in the previous analysis (Figures 3.15 and 3.16 with 100% of $Group_2$ users). Figure 3.19 plots again the failure probability and the mean delay, keeping the packet threshold $P_{thr}$ fixed and varying the time threshold $T_{thr}$.

It can be observed that in both Figures 3.20 and 3.21, the mean transmission delay follows a monotonic behavior. In fact, the less the packet or time thresholds are, and the less is the congestion in the network so the time to accomplish a successful request improves. Instead, the probability to abort a request follows a bathtub curve: at first the abort probability is high due to a too poor choice of the threshold (too many requests are deliberately stopped), while in the end, is high due to a too high congestion in the network. Of course, the best choices of the threshold value stay in the middle. Finally, it is worthwhile to observe the increase of performance obtained with the insertion of a threshold mechanism. Figures 3.18 and 3.19 show that choosing $T_{thr} = 11$ and $P_{thr} = 3$ leads to a reduction of almost 10% of the failure probability and of about 3.5% of the average delay with respect to the case without threshold. Figures 3.20 and 3.21 plot the same measures as Figures 3.18 and 3.19, respectively, but in the case where the network load is 110%, so as to investigate on the effects of the threshold mechanism.
Figure 3.20: Failure Probability and Delay at varying of $P_{thr}$ under 110% load

Figure 3.21: Failure Probability and Delay at varying of $T_{thr}$ under 90% load
in overload condition. From Figure 3.21, it can be noted that the optimal choice of the two threshold parameters $T_{thr}$ and $P_{thr}$ is different from that made in the previous case where the network load was 90%. This is a clear indication that the network load influences the choice optimal values for the threshold parameters. However, despite the setting $T_{thr} = 11$ and $P_{thr} = 3$ is not optimal in such a case, still adopting the threshold mechanism with these values leads to an improvement of about 18.3% for the failure probability and 8.5% for the delay, with respect to the case without the threshold mechanism. It can be concluded that such a threshold mechanism is very helpful, and the benefits it brings grow with the increase of the network load.

3.4 Conclusions

This Chapter has presented a study on modeling and analysis of the Quality of Service (QoS) provided by the GPRS network system both at network level and from a user’s perspective. Firstly, we have modeled the behavior of the GPRS system and some indicators of the QoS, related to individual packets, have been evaluated. In this preliminary study, has been supposed that all users have the same behavior, their requests fit one LLC frame and they never give up in case of congestion of the network. Afterwards a detailed user behavior has been considered. Such users can make request of different sizes, lengths and rates. We considered two categories of users (in order to focus on specific characteristics of the user population) and we combined them with the random access model to make quantitative estimations of properly identified QoS indicators. Thus, this work accounts for differences in the offered traffic, the possibility to deliberately abort a request after a prefixed time interval, and the possibility to use timing and packet thresholds to perform intermediate checks on the evolution of the transmission in order to take decisions on staying connected or not. The selected QoS measures are mainly the delay time for the network to successfully accomplish a user request, and the probability to abort the transmission because of violation of the user deadline or violation of threshold conditions. The purpose of this study was to understand the most impacting user’s characteristics, to help the system designer in devising satisfactory system configurations. The numerical evaluation and sensitivity analysis carried out on representative system scenarios actually show helpful indications in this direction.
Chapter 4

Characterization of the GPRS Behavior in Consequences of Outages

4.1 Introduction

Focusing on the contention phase, where users compete for channel reservation, this Chapter analyzes the GPRS with the objective to understand its behavior under critical conditions, as determined by periods of outages, which significantly impact on the resulting dependability. In fact, during outages (service unavailability), users trying to access the service accumulate, leading to an overload of the system. When the system resumes its operations, the accumulated users determine a higher probability of collisions on resources assignment (and therefore a degradation of the QoS perceived by the users). Our analysis, performed using a simulation approach, allowed to gain insights on the impact of outages on the QoS and of the overload that GPRS systems have to face after outages.

4.2 Modeling the Effects of Outages on the GPRS Uplink Contention

This Section presents the numerical evaluation performed solving the SAN model of the random access procedure in a transient period, extending from the outage occurrence to an interval of time following the system repair. The focus is on a single outage of varying duration; its impact on the identified indicators (degradation during the outage and recovery time) is analyzed under
varying users’ population, users’ characteristics and system workload. The presented results will help the reader in better understanding the evaluations shown in Chapter 5.

4.2.1 Measures of Interest

The effects of outages on the GPRS services have studied through specific indicators which are:

1. The time necessary for the system to reach its steady-state behavior, following the end of an outage (recovery time);

2. A measure of the congestion induced by the outage in the system, as an indication of the service degradation, both during the outage and during the recovery time.

4.2.2 Modeling Assumption

The random access procedure of Figure 3.1 has been defined under the assumptions stated in 3.2.2 plus the following one:

- During an outage the network is completely blocked and the on-going data transmissions are interrupted; such a kind of outage could be caused, for example, by a malfunction in the antenna of the cell itself, which therefore stops to send and receive signals. The outage occurrence follows an exponential distribution, while the repair time is assumed to follow a deterministic distribution (if not specified differently).

4.2.3 The SAN Model

The model of the random access procedure is substantially that of Figure 3.1. However, it has been still slightly modified to account for outages as in Figure 4.2. In particular, the subnet of Figure 4.1 has been added and is briefly described in the following.

- Places work, out_serv and ok are used to represent the occurrence of outages, and the consequent repair of the system. A token in the place work represents the correct running of the network. A token in the place out_serv represents the unavailability of the network because an outage happened and a token in place ok that the network has been repaired.
Characterization of the GPRS Behavior...

Figure 4.1: Subnet which models the outage effects

Figure 4.2: Modified random access model
- The firing of the timed activity *outage* represents the occurrence of an outage. This activity has a deterministic time, chosen in such a way to model the occurrence of an outage after the network has already reached the steady-state.

- The output gate *control_fault* simulates the effect of the fault. An occurrence of a fault consists in the inhibition of the immediate activity *req* and the moving of the tokens of the whole net in the place *nup*. This form of outage could be caused, for example, by a malfunction in the antenna of the cell itself, which therefore stops to send and receive signals. The time necessary to have all the tokens in *nup* represents the outage duration to reach the worst system conditions (i.e., the maximum congestion). The timed activity *rip* represents the time necessary to repair the system and moves a token in the place *ok*.

In such model, at growing values of the outage duration (transition *outage*), more and more tokens move from place *Served* to place *nup*, meaning that more users are making new service requests. Some time is necessary to complete this process (which depends on how frequently served users issue new requests). Therefore the longer is the outage, the higher the number of served users affected by the outage, yielding a higher degradation of the QoS.

### 4.2.4 Settings for Numerical Evaluation

We evaluated two scenarios: Table 4.1 and Table 4.2 reports the values assigned to the main parameters in the numerical evaluation in Scenario 1 and Scenario 2, respectively. The varying parameters in the experiment are the duration of outages, the request size, the number of users in the cell, and the user inter-request time. The scenarios are chosen in a way to offer to the network the same workload being equal the number of user in the cell. The length of the queue is also different to ensure that users do not wait in the queue more than 5 seconds, as suggested by ETSI standards. Details on how the length of the queue is chosen can be found in [65].

### 4.2.5 Numerical Evaluation

The measure of the congestion induced by the outage in the system is derived from the SAN model simply observing the marking of the place *Served*. The recovery time is obtained from the previous one, observing the time necessary for the system to get close enough to its expected, steady-state behavior, following the end of an outage.
### Table 4.1: Relevant parameters and their values (Scenario 1)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPDCH</td>
<td>Number of Slave Packet Data CHannel</td>
<td>3</td>
</tr>
<tr>
<td>Users</td>
<td>Number of users in the cell</td>
<td>75-150</td>
</tr>
<tr>
<td>PRACH</td>
<td>Number of Packet Random Access CHannel</td>
<td>2</td>
</tr>
<tr>
<td>( Q_L )</td>
<td>Length of the queue</td>
<td>2</td>
</tr>
<tr>
<td>( T_{req} )</td>
<td>User inter-request time, following an exponential distribution</td>
<td>60 sec. (average)</td>
</tr>
<tr>
<td>( T_{out} )</td>
<td>Outage duration, following a deterministic distribution</td>
<td>20-300 sec.</td>
</tr>
<tr>
<td>( R_{size} )</td>
<td>Request size, following a uniform distribution</td>
<td>[1000-1600] byte</td>
</tr>
</tbody>
</table>

### Table 4.2: Relevant parameters and their values (Scenario 2)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPDCH</td>
<td>Number of Slave Packet Data CHannel</td>
<td>3</td>
</tr>
<tr>
<td>Users</td>
<td>Number of users in the cell</td>
<td>75-150</td>
</tr>
<tr>
<td>PRACH</td>
<td>Number of Packet Random Access CHannel</td>
<td>2</td>
</tr>
<tr>
<td>( Q_L )</td>
<td>Length of the queue</td>
<td>6</td>
</tr>
<tr>
<td>( T_{req} )</td>
<td>User inter-request time, following an exponential distribution</td>
<td>30 sec. (average)</td>
</tr>
<tr>
<td>( T_{out} )</td>
<td>Outage duration, following a deterministic distribution</td>
<td>20-220 sec.</td>
</tr>
<tr>
<td>( R_{size} )</td>
<td>Request size, following a uniform distribution</td>
<td>[500-800] byte</td>
</tr>
</tbody>
</table>
Figures 4.3 and 4.4 and figures 4.4 and 4.6 show the effects of outage on the number of served users, both during the outage and during the recovery time, at varying of the users acting in the cell for Scenario 1 and Scenario 2, respectively. Several curves have been plotted for different values of the outage duration. At growing values of the outage duration, more and more tokens exit the place Served, meaning that more users are making new service requests. This is traced through the descendant line in the Figures. Then, the ascendant lines show the time necessary for obtaining the “normal” number of tokens in the place Served (as determined by a steady state analysis under normal system conditions, i.e. in absence of outages, and indicated by the upper horizontal line). These Figures show also how the behavior of the GPRS during outages and recovery depends on the kind of traffic offered to the network. In Figures 4.3 and 4.5 it can be observed that the decrease rate of the number of tokens in place Served depends on the number of users acting in a cell; however, the time to empty that place is the same in both cases. This is not surprising, since users issue requests with the same rate. It can be noticed, also, that the degradation time is much longer in Scenario 1 than in Scenario 2 due to the different behavior of the users (the users inter-request time is higher). The recovery time is also much longer compared to Scenario 2. Summarizing, in the Scenario 1 the system degradation is slower than in Scenario 2, and also the time to come back to the “normal” working conditions is longer.

Besides the offered traffic, the recovery time depends on the duration of outages, as shown in Figure 4.7. For low values of the outage duration the recovery time varies significantly, becoming almost independent from it when outage duration gets high values (in the figure, greater than 160 seconds).

### 4.3 On the effects of faults with/without queuing mechanism

An evaluation of the effectiveness of a queuing mechanism (for uplink channel reservation) used to reduce the effects of collisions has been carried out. The queue mechanism is already included in the model of the random access procedure of Chapter 3 (see Figure 3.1). In this Section the benefits of this queue mechanism are pointed out.
Characterization of the GPRS Behavior ...

Figure 4.3: Behavior during outages and recovery (Scenario 1, 150 users)

Figure 4.4: Behavior during outages and recovery (Scenario 1, 75 users)
Figure 4.5: Behavior during outages and recovery (Scenario 2, 150 users)

Figure 4.6: Behavior during outages and recovery (Scenario 2, 75 users)
4.3.1 Measures of Interest, Modeling Assumption and SAN Model

The evaluated measures, and the modeling assumptions are the same of Sections 4.2.1 and 4.2.2. The solved model is the same of Figure 3.1 with slight modifications when the queueing mechanism is not accounted for. In this case, the SAN elements that implement the queue (place queue, the immediate transition q\textunderscore control\_a, transition q\textunderscore control\_a, and the input gate q\textunderscore control), are excluded.

4.3.2 Settings for the Numerical Evaluation

The settings for the numerical evaluations are specified in the following Table 4.3.

4.3.3 Numerical Evaluation

Figure 4.8 compare the system behavior, in presence of outage, with configurations employing and not employing the queue mechanism. Two couples of plots are shown in the figure, obtained varying the outage duration; $T_{out}$ has been assigned a low and a high value with respect to significant outage duration (note that the maximum outage duration to bring the system at its
maximum congestion level, i.e., the time to all users are attempting to access to the network, is the same for both systems, with and without the queue, since it only depends on the number of user’s population and on the users $T_{req}$). As a first observation, employing the queue leads to an increase in performance, quantified by the higher average marking of the place $Served$, represented by the horizontal lines (enclosed by two dotted lines at a distance of $\pm 3\%$). This result confirms the benefits expected from the queue mechanism. The rate with which tokens exit the place $Served$ during the outage (left descending curves) is, of course, higher for the configuration using the queue, since the average marking of this place (that is, at the time the outage starts) is higher for this configuration.

The evaluation of the recovery time, necessary to bring the system to its average operational level after the end of the outage is shown in Figure 4.9 completes our study. It shows the recovery time as a function of the outage duration, for two configurations which only differ for the queue mechanism. The better behavior exhibited by the configuration which uses the queue, already pointed out in the previous Figure 4.8, is now extremely clear. The sensitivity of the recovery time to $T_{out}$ is especially noticeable in the left part of the figure (low values of $T_{out}$). Summarizing, the results shown by this analysis re-enforces the benefits of adopting the queue mechanism, also as a measure to better cope with outage occurrences.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPDCH</td>
<td>Number of Slave Packet Data CChannel</td>
<td>2</td>
</tr>
<tr>
<td>Users</td>
<td>Number of users in the cell</td>
<td>150</td>
</tr>
<tr>
<td>PRACH</td>
<td>Number of Packet Random Access CHannel</td>
<td>1</td>
</tr>
<tr>
<td>$Q_L$</td>
<td>Length of the queue</td>
<td>2</td>
</tr>
<tr>
<td>$T_{req}$</td>
<td>User inter-request time, following an exponential distribution</td>
<td>rate $1/(76.4 \text{ sec.})$</td>
</tr>
<tr>
<td>$T_{out}$</td>
<td>Outage duration, following a deterministic distribution</td>
<td>20-300 \text{ sec.}</td>
</tr>
<tr>
<td>$R_{size}$</td>
<td>Request size, following a uniform distribution</td>
<td>range $[0.812-1.292]$</td>
</tr>
</tbody>
</table>

Table 4.3: Relevant parameters and their values
Figure 4.8: Effects of outages with/without queueing mechanism

Figure 4.9: Effects of the queue on the recovery time at varying of outage duration
4.4 Conclusions

This Chapter has presented a study on modeling and analysis of the GPRS random access procedure (where users compete for the channel reservation) in presence of outages. An estimation of the effect of outages on the QoS as perceived by users has been performed. Outages constitute a dependability-critical condition for communication systems as the GPRS, implying unavailability of the system and a consequent accumulation of users asking for the service. As a consequence, as soon as the GPRS comes back in operation, the high level of requests leads to a higher probability of collisions. We have therefore analyzed the system during and in an interval after outages, to understand the degradation of the QoS (in terms of unavailability of the service) and the time necessary to restore the steady-state behavior (recovery time), which has been in turn properly evaluated. It has been shown that depending on the number of users acting on a cell and their traffic pattern, the outage effects can be different, both for what concerns the transient period during which the system experiences an outage and during the recovery time to come back in the expected “normal” conditions. Also, the achieved results show the benefits on the above measures of introducing a queue mechanism of users’ requests.
Chapter 5

GPRS Dependability Analysis

5.1 Introduction

In the previous Chapter the impact of a single outages on the GPRS network has been analyzed and has been shown that its effects on users’ QoS varies over the time and does not follow a simplistic on-off behavior. Usually, availability is defined as the fraction of time a system is working properly on the overall observation time. However such a definition neglects the inertia of the GPRS systems at varying of its status and of users’ behavior. Indeed, in the previous Chapter it has been shown that the outage effects can be different at varying of the number of users acting on a cell and their traffic pattern.

Here, the whole GPRS network architecture has been considered. Each block of the GPRS architecture is modeled considering their failure rate/repair behavior, the redundancy, and the number of spare components through the end-to-end transmission from MS up to external IP networks. This leads to the introduction of a model, which take into account the effects of hardware configuration of the GPRS system on the nature of the outages that can occur during the lifetime of the system. After, an accurate way to estimate availability of GPRS has been proposed.

On the other hand, the effects of outages on user QoS perspective are investigated. In fact, standard network QoS analysis usually accounts for the infrastructure performance/availability only, with scarce consideration of the user perspective. Users that can differentiate for different sizes of requests sent to the network at different rates and subject to timing constraints are now considered.
5.2 GPRS Architecture Availability Model

This Section introduces the network models employed in the evaluation of the availability of the whole GPRS network. It presents an architectural view of GPRS availability, by building a model that represents the failure/repair behavior of the end-to-end transmission path from the MS up to the external IP networks. We accounted for redundancies of network elements for our evaluation. In this Section, we did not consider the effects of outages on the uplink contention by user, yet. In fact, is likely that after an outage, users try to access to the network at the same time. This hypothesis will be relaxed in the following.

5.2.1 Measures of Interest

For the moment, we are interested in the classical availability of the architecture of GPRS expressed as the fraction of time a system is working properly on the overall observation time. In Section 5.3 a more accurate availability estimation is derived. Particularly, in this study we evaluate:

1. The probability distribution of outages,
2. The classical availability,
3. The rate of the outages,
4. A sensitivity analysis of the availability at varying values of the duration of the repairs of faulty components and their rate of failures.

5.2.2 System Availability Related Aspects

The availability of the overall GPRS system basically depends on the availability of the MS, BTS, BSC, SGSN and GGSN network elements. i.e., the entities providing the service in the external packet data networks, plus some other GSM network components, used to authenticate and localize the mobile user, such as the Home Location Register (HLR) and Visitors Location Register (VLR) databases. Moreover, the availability of the communication links between these units is also necessary for the system to be available from a user perception. These communication links include:

- Radio links between the MS and the BTS across the Um interface;
- E1 spans between a BTS and a BSC supporting the Abis interface;
- E1 spans from a BSC to a SGSN providing Gb interface;
5.2.3 Modeling Assumptions

To limit the scope of the modeling, we will consider one single GPRS cell, and do not take into consideration the mobility of users. These restrictions will limit the number of users that need to be taken into consideration. Moreover, the following assumptions have been made for the sake of building a manageable availability model of GPRS:

- The Mobile Station is fault free;
- The reliability of the radio links will not be taken into consideration in this model;
- The pure GSM network elements, such as HLR and VLR, are fault free, as well as all the links supporting the interface towards them (shown in Figure 2.1);
- The external packet data network (Internet) is fault free, as well as the links supporting the Gi interface;
- The Gp interface is not supported.

Therefore, our availability model will consider the path shown in Figure 5.1 from the MS up to the GGSN, taking into consideration the reliability and the repairs of the networks elements along the path Figure 5.1, as well as the links supporting the communications across the depicted interfaces.

It is worthwhile remarking that several of the elements determining the network availability of GPRS are in fact redundant components. We will
shortly describe in the following the various types of redundancy that may be deployed in the architecture of the system.

The Gb interface between the BSC and the SGSN is deployed in a redundant fashion, with two Frame Relay switches as shown in Figure 5.2. This is done to tolerate the outages typically caused by network-independent events (links cut). Only one of the links is actively working, whereas the second one is a spare, ready to be put into operation in case the first link stops working. Both the SGSN and the GGSN network elements are deployed in a redundant $N + M$ configuration, in which $N$ modules are actively working in a load-sharing mode, and $M$ modules are held in a cold spare state. In case of failure of one or more active modules, the spare modules are enabled to switch over and take the role of active modules. Failed elements or links are mainly repaired through replacement. It is important observing that links repair time is usually among the major contributors to telecommunications systems downtime.

5.2.4 Network Availability Model Definition

We build the availability model of GPRS by using a Stochastic Activity Network approach. For each of the network elements introduced in the previous section, we will define a separate SAN subnet. Joining the various subnets through the JOIN operator of UltraSAN allows defining in quite a natural way the overall GPRS availability model.

SAN Subnets of the Non Redundant Elements

The simple SAN subnet in Figure 5.3.1 models the availability of the BTS network component. This same subnet (apart from obvious changes in the SAN elements) will be used to model all the other network elements for
which no redundant deployment is foreseen, e.g. BSC, Abis E1 link, Gn E1 link. The initial marking of place BTS$_{up}$ is equal to 1, meaning that the modeled component is available in the initial state. Firing of activity BTS$_{fail}$ represents a failure of the component. The firing time of BTS$_{fail}$ follows a negative exponential distribution. At firing time, the token is removed from place BTS$_{up}$ and put into place BTS$_{down}$, representing the unavailability of the component. Firing of BTS$_{repair}$ represents the repair completion of the component, which occurs after a deterministic amount of time after the failure. The token is moved back to the BTS$_{up}$ place when BTS$_{repair}$ fires.

Place UNAVAILABLE is used to keep track of the availability of the BTS component together with that of the other GPRS components. In fact, for the non-redundant models, the UNAVAILABLE place is completely redundant, its marking being always equal to that of the down place of the subnet. The usefulness of this place will be explained in the following. The firing time distributions and the parameters used to instantiate the distributions are shown in Table 5.1.

### SAN Subnet of the Redundant Frame Relay Links

The SAN subnet showed in 5.4 models the availability of the redundant links across the Gb interface. The SAN in Figure 5.4 is quite similar to the one presented in Figure 5.3. The difference is found in the number of tokens that circulate in the model, which is equal to two to represent the

---

**Figure 5.3: SAN model of the BTS**

<table>
<thead>
<tr>
<th>Activity</th>
<th>Distribution</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BTS$_{fail}$</td>
<td>Exponential</td>
<td>rate = 1/MTTF$_{BTS}$</td>
</tr>
<tr>
<td>BTS$_{repair}$</td>
<td>Deterministic</td>
<td>value = MTTR$_{BTS}$</td>
</tr>
</tbody>
</table>

Table 5.1: Activity distributions for the BTS model
GbLINK_UP
GbLINK_fail
GbLINK_up
GbLINK_repair
GbLINK_repair_all
UNAVAILABLE
GbLINK_DOWN
GbLINK avail
GbLINK_repair
GbLINK_repair_all
UNAVAILABLE

Figure 5.4: SAN model of the redundant Gb link

<table>
<thead>
<tr>
<th>Activity</th>
<th>Distribution</th>
<th>Parameter values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$GbLINK_fail$</td>
<td>Exponential</td>
<td>rate = $1/MTTF_GbLINK$</td>
</tr>
<tr>
<td>$GbLINK_repair$</td>
<td>Deterministic</td>
<td>value = $MTTR_GbLINK$</td>
</tr>
</tbody>
</table>

Table 5.2: Activity distribution for the Gb link model

double redundant Frame Relay link. Only one Frame Relay is active at the time. The active link fails according to the exponential failure rate of activity $GbLINK\_fail$. When the active link fails, the second one becomes active and therefore subject to fail. To model that, each firing of $GbLINK\_fail$ removes only one token from place $GbLINK\_up$ and puts it into place $GbLINK\_down$. Activity $GbLINK\_repair$ fires with a deterministic time. A repair removes all token present in place $GbLINK\_down$ and puts them back into place $GbLINK\_up$ through the output gate $GbLINK\_repair\_all$.

The output gate $GbLINK\_avail$ puts a token in place $UNAVAILABLE$ when no more Frame Relay connections are available. The token is removed when the subnet has two tokens in place $GbLINK\_down$, and the repair gets completed. The subnet distributions and their parameters are listed in Table 5.2. The definition of the function executed in the output gate of the model is given in Table 5.3.

SAN Subnet of the SGSN and GGSN Redundant Network Elements

The SAN subnet in Figure 5.5 models the SGSN component, according to the redundancy management description given in the previous subsection. The SAN model for the GGSN is the same (apart from a renaming of the subnet elements) as that of the SGSN in Figure 5.5.

Exactly $N$ and $M$ tokens are put in place $SGSN\_up$ and $SGSN\_spare$ to
### Table 5.3: Output gate definition for the GbLINK subnet

<table>
<thead>
<tr>
<th>Gate</th>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GbLINK_avail</td>
<td>Output</td>
<td>if (MARK(GbLINK_up)==0) MARK(UNAVAILABLE)++;</td>
</tr>
<tr>
<td>GbLINK_repair_all</td>
<td>Output</td>
<td>MARK(GbLINK_up)++; if (MARK(GbLINK_down)==2) MARK(UNAVAILABLE)--; MARK(GbLINK_down)=0;</td>
</tr>
</tbody>
</table>

Figure 5.5: SAN Model of SGSN
CHAPTER 5

<table>
<thead>
<tr>
<th>Activity</th>
<th>Distribution</th>
<th>Parameter values</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGSN_fail</td>
<td>Exponential</td>
<td>Rate = \text{MARK(SGSN_up)}/\text{MTTF_SGSN}</td>
</tr>
<tr>
<td>SGSN_repair</td>
<td>Deterministic</td>
<td>Value = \text{MTTR_SGSN}</td>
</tr>
<tr>
<td>SGSN_switch</td>
<td>Exponential</td>
<td>Rate = (N-\text{MARK(SGSN_up)})/\text{MTTS_SGSN}</td>
</tr>
</tbody>
</table>

Table 5.4: Activity distributions for SAN model of SGSN

initialize the subnet. Activity \text{SGSN\_fail} models the failure of one active SGSN component. At firing time, one token is removed from place \text{SGSN\_up} and put into place \text{SGSN\_down}. As soon as the number of tokens in place \text{SGSN\_up} becomes less than \(N\), activity \text{SGSN\_switch} gets enabled through input gate \text{SGSN\_act\_switch}. Firing of activity \text{SGSN\_switch} moves one token from \text{SGSN\_spare} to \text{SGSN\_up}, representing the activation of a spare component. The firing of \text{SGSN\_repair} models the completion of a repair activity. At firing time, output gate \text{SGSN\_after\_repair} puts a token in place \text{SGSN\_up} or \text{SGSN\_spare} depending on the current SGSN model marking. The token is put into place \text{SGSN\_up} if and only if its marking is less than \(N\) (i.e. more active SGSN are needed), or in place \text{SGSN\_down} otherwise (i.e. the newly repaired component becomes a spare one). A token is put in place \text{UNAVAILABLE} whenever one active SGSN fails. Tokens are removed from \text{UNAVAILABLE} by the output gates \text{SGSN\_avail} and \text{SGSN\_after\_repair}, according to the rules specified in Table 5.4. The rationale behind this token game is that the token remains in place \text{UNAVAILABLE} only during the switch-on time of spare components, or when no more spares can take over and the SGSN has to wait for the repair. The full definition of distributions and gates for the subnet is given in Table 5.4 and Table 5.5, respectively.

5.2.5 UltraSAN Composed Model

The overall network availability model is obtained from the subnets defined above by joining them with the SAN Join operator. The UltraSAN composed model is shown in Figure 5.6. All the subnets are joined over the common place \text{UNAVAILABLE}. This implies that a single \text{UNAVAILABLE} place will exist in the model, with all subnets using that same place.

As a result, the number of tokens in place \text{UNAVAILABLE} will represent the number of unavailable network elements of the GPRS system. The advantage of this definition of the common place is the easiness in the definition of the availability measures. Indeed, it is enough to check the marking of place \text{UNAVAILABLE} to verify whether the GPRS infrastructure is available or not. This same place will be exported to the upper-level model of
<table>
<thead>
<tr>
<th>Gate</th>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$SGSN_{act_switch}$</td>
<td>input</td>
<td>enabled iff $\text{MARK}(\text{SGSN}_{up}) \geq N$</td>
</tr>
<tr>
<td>$SGSN_{avail}$</td>
<td>output</td>
<td>if ($\text{MARK}(\text{SGSN}_{up}) = (N-1)$) $\text{MARK}(\text{UNAVAILABLE})$ - -;</td>
</tr>
<tr>
<td>$SGSN_{failed}$</td>
<td>output</td>
<td>if ($\text{MARK}(\text{SGSN}_{up}) = (N-1)$) $\text{MARK}(\text{UNAVAILABLE})$ ++;</td>
</tr>
<tr>
<td>$SGSN_{after_repair}$</td>
<td>output</td>
<td>if ($\text{MARK}(\text{SGSN}_{up}) = (N-1)$) $\text{MARK}(\text{UNAVAILABLE})$ - -;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>else $\text{MARK}(\text{SGSN}_{spare})$ ++;</td>
</tr>
</tbody>
</table>

Table 5.5: Gate definitions for the SGSN subnet

Join

BTS AbisLINK BSC GbLINK SGSN GnLINK GGSN

Figure 5.6: Composed model for the network availability model
the random access procedure to GPRS.

5.2.6 Settings for the Numerical Evaluation

We conducted a numerical evaluation of the availability of the GPRS infrastructure, by solving the composed network availability model of Figure 5.6. A simulation approach is adopted, achieving a steady-state solution. In this experiment, as well as in subsequent numerical evaluations, results have been determined with 95% confidence interval and a relative confidence interval lower than 1%. Table 5.6 summarizes the main parameters involved in the analysis, together with the numerical values assigned to them in our experiments. We recall that MTTF, MTTR, and MTTS stand for Mean Time To Failure, Mean Time To Repair, and Mean Time to Spare, respectively. Moreover, $N$ is the number of modules that are actively working in a load-sharing mode, and $M$ the number of modules that are held in a cold spare state. The details of the mining of the other parameters can be found in Sections 5.2.2 and 5.2.4.

5.2.7 Numerical Evaluation

The measures of interest are derived from the SAN model as follows:

- The probability distribution of outages is evaluated by means of an additional SAN models (described in Appendix A) which implement a counter for measuring the length in time of an outage; we break up the range of the values covered by the counter in disjoint adjacent intervals and we measure the proportion of time the value of the counter falls in such intervals deriving the wanted histogram.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>5</td>
<td>MTTR_AbisLINK</td>
<td>4 hours</td>
</tr>
<tr>
<td>$M$</td>
<td>1</td>
<td>MTTR_BSC</td>
<td>1 hour</td>
</tr>
<tr>
<td>MTTF_AbisLINK</td>
<td>1 year</td>
<td>MTTR_BTS</td>
<td>2 hours</td>
</tr>
<tr>
<td>MTTF_BSC</td>
<td>6 months</td>
<td>MTTR_GGSN</td>
<td>1 hour</td>
</tr>
<tr>
<td>MTFF_BTS</td>
<td>4 months</td>
<td>MTTR_GbLINK</td>
<td>4 hours</td>
</tr>
<tr>
<td>MTTF_GGSN</td>
<td>6 months</td>
<td>MTTR_GaLINK</td>
<td>1 hour</td>
</tr>
<tr>
<td>MTTF_GbLINK</td>
<td>1 year</td>
<td>MTTR_SGSN</td>
<td>1 hour</td>
</tr>
<tr>
<td>MTTF_GnLINK</td>
<td>1 year</td>
<td>MTTS_GGSN</td>
<td>5 seconds</td>
</tr>
<tr>
<td>MTTF_SGSN</td>
<td>6 months</td>
<td>MTTS_SGSN</td>
<td>5 seconds</td>
</tr>
</tbody>
</table>

Table 5.6: Settings for numerical evaluation
The classic availability (or unavailability) is determined observing the marking of the place \textit{UNAVAILABLE}.

The number of faults per hour (the rate of outages) is obtained from an additional SAN model (described in Appendix A) simply by counting the number of times a given transition fires in given amount of time, e.g., one hour. This transition fires whenever the marking of place \textit{UNAVAILABLE} becomes greater than zero.

Figure 5.7 shows the distribution of the outage duration. Since the repair times of faulty components are deterministic, there are only four possible values of outage duration spanning from seconds to hours (see Table 5.6). From the figure, the most probable outage has a duration of 5 seconds, which corresponds to the Mean Time to Spare (MTTS) of SGSN and GGSN components. Figure 5.8 shows the availability at varying values of the Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) of hardware components listed in Figure 8. Let $K_F$ and $K_R$ be two constants representing a scaling factor of MTTF and MTTR, respectively. The availability implied by $K_F = K_R = 1$ is that in case Figure 5.7. Setting $K_F > 1$ implies reducing the probability of hardware failure while $K_R < 1$ leads to faster components repair. Then, not surprisingly, the case $K_F = 2$ and $K_R = 0.8$ results in the highest availability in the example depicted in Figure 5.8.
5.3 The GPRS Accurate Availability Estimation

In this section the GPRS architectural model and the random access model, analyzed separately in Sections 5.2 and 4.2 respectively, are here combined and solved together. Evaluations on the join model are carried out to gain some insight on the effects of system availability and outages on the service provided by the GPRS system. At last, a sensitivity analysis shows how the frequency and duration of outages, given a fixed availability, affects the performance of GPRS.

5.3.1 Measure of Interest and Modeling Assumptions

We have identified in the average number of served users per hour an accurate availability measure. We evaluated this measure both with and without keeping into account for the effects of outages on the uplink contention.

The following model has been defined under the assumption already stated about the model of the random access procedure in presence of outages and the GPRS network availability model in Sections 4.2.2 and 5.2.3, respectively.
5.3.2 The Overall Model

We now illustrate how the GPRS architectural model and the random access model have been composed in order to provide an accurate availability estimation of the GPRS system. The composed model allows taking into account both the availability of the components of the GPRS infrastructure, and the effects of outages on the user’s perceived quality of service.

The composition of the previously described models is actually quite straightforward, as shown in Figure 5.9, since it consists in joining the overall architectural model (shown in Figure 5.6) with the random access model (as specified in the following) using the SAN Join operator. All the subnets are joined together having the place UNAVAILABLE as the only common place through which all the interactions take place.

Random Access Model

The version of the random access model employed to build the model of the whole GPRS network is nearly similar to that of Figure 3.1 with the only extension that the subnet of Figure 5.10 is added. This subnet is used to represent the occurrence of outages, and the consequent repair of the system. The mining of the net elements is explained in the following.
A token in the place \textit{UNAVAILABLE} represents the unavailability of the network. Note that this place is in common with the underlying network availability model;

- The input gate \textit{inp\_{out}} enables the transition \textit{outage} without removing the token from the place \textit{UNAVAILABLE};

- The firing of the timed (but very fast) activity \textit{outage} triggers the effects of an outage by means of the output gate \textit{outage\_effects}. This gate simulates the effects of the fault through the inhibition of the immediate activity \textit{req} and the gradual moving of the tokens of the whole net in the place \textit{new\_req}. For the sake of simplicity, the others Petri net elements, which assure that the \textit{outage} transition fires only once when there is a token in place \textit{UNAVAILABLE} without removing it, have been omitted. Further modeling details can be found in Appendix A.

The modified random access model is shown in Figure 5.11.
5.3.3 Settings for Numerical Evaluation

The settings for numerical evaluation are the same of those specified in Tables 5.6 (for what concern the architectural availability model of Figure 5.6) and 4.1 (for what concern the random access procedure). The only one exception is the parameter $T_{out}$ of Table 4.1 that is meaningless in the following evaluations.

5.3.4 Numerical Evaluation

The measure *averaged number of served users per hour* is derived from the SAN model simply by counting the number of times the transition $to\_req$ of the random access procedure (Figure 5.11) fires in given amount of time, e. g. one hour.

Figure 5.12 shows the reduction (percentage) of users served per hour, at varying inter request time, while Figure 5.13 shows the change of availability when the service dependability model is considered with respect to the network availability as estimated in subsection 5.2.6. The second x-axis shows the corresponding network traffic load as percentage of the network capacity. The dashed line reports the values obtained from the network model as estimated in the subsection 5.2.6. This is a constant value, because it does not depend on the transient effect of outages on the service model. The solid line represents the “real” availability as perceived by the class of users modeled.

Looking at the Figures one may observe that for high load the user perceives a better QoS than just considering the network availability model. An explanation for such behavior is that the smooth degradation of the user perceived QoS when outages start out–weights the slow recovery after the system resumes operations. Actually the system is “always” overloaded so the overload at restarting does not cause any particular harm. Very different is the situation at lower loads. In these cases the effect of the slow restart determines the users to perceive a loss of QoS due to the congestion at system restart. In the present parameters setting, for inter request times of 80 seconds the degradation is the highest leading to a worsening of availability of about 20%. We can conclude by saying that the bare estimation of QoS derived by the sole network model is an underestimation of what really users perceive, and that accounting of the transient effect brought about outages is necessary.
Figure 5.12: Percentage reduction of served users per hour with and without considering the transient effects

Figure 5.13: Availability reduction with and without considering the transient effects
5.4 On the Impact of Outage Frequency and Duration

In the previous Section we performed the evaluation of a specific GPRS architecture with its network availability, on which we obtained a more accurate estimation of the service availability provided to GPRS users. Additionally, we are interested in performing some sensitivity analysis at varying the frequency and duration of outages periods and for different values of network availability. Thus, in this subsection we consider a model that allows performing such detailed analysis of service unavailability. We replace all the architecture submodels with a very simple submodel with two places (Available and Unavailable) and two timed transitions to model failure and repair of the architecture. The firing times of these transitions are derived once we fix the network availability and the Mean Time To Outage (MTTO) or the Mean Time To Repair (MTTR). The settings of the GPRS parameters are the same shown in Table 4.1.

Figure 5.14 shows the reduction (percentage) of users served per hour at varying MTTR for inter request time of 80 sec., and fixing the network availability to 0.998516 (which is the availability of the system studied in Section 5.2). We can observe that if the MTTR is short we have actually a gain in the QoS provided to users. The shorter (i.e. more frequent) the outages the
Figure 5.15: Served users per hour as a function of MTTR for $\text{inter\_request\_time} = 60$ sec

Figure 5.16: Served users per hour as a function of MTTR for $\text{inter\_request\_time} = 80$ sec
better the user perceived QoS for the same “basic” system availability. Figures 5.15 and 5.16 plot the served users per hour at varying the MTTR and the availability, for inter request times fixed to 60 and 80 seconds, respectively. They show how QoS improves as availability improves (as expected) but also as MTTR becomes shorter. Last, Figure 5.17 shows the served users per hour as a function of availability for different distributions of the MTTF, just to point out how not only the MTTF is important in determining the QoS, but also the distribution of the time to failure.

5.5 Effects of Outages on the QoS of GPRS Network Under Different User Characterizations

This Section presents a study on modeling and analysis the effects of outages on the Quality of Service of GPRS network system explicitly accounting for different user behaviors. In fact, we consider that the user viewpoint cannot be neglected in such kind of analysis, being user satisfaction the ultimate goal of system suppliers.
5.5.1 Measure of Interest

As in the previous Section, we concentrate on the averaged number of served users per hour distinguishing between different user classes.

5.5.2 Model Assumptions

For the user behavior, we defined two slightly different models, to allow capturing a wider spectrum of user characteristics. One, named Hom_User, models a class of users, inside which users share the same characteristics (homogeneous user). Of course, the Hom_User Model has been developed in a parametric way such that it can be customized in accordance with relevant characteristics of the user class under analysis. Through this model, it is possible to analyze separately users accessing the GPRS network through different applications (e.g., the class of users operating web browsing, that of users operating e-mails, etc.). The other, named Het_User, models a class of users inside which users may behave differently (heterogeneous user). It allows representing realistic scenarios where, from time to time, the same single user may access the GPRS through a variety of applications, thus making requests of varying size. Users’ models has been defined under the assumptions already stated in 3.3.2 with the following modifications:

- For the Hom_User model only: should a user timeout elapse before completing the request transmission, that user waits for his next inter-request time instant for issuing a new request;
- For the Het_User model only: users are grouped in just one non-homogeneous class, inside which users may differ in the length of requests they may issue, in the inter-request time, and in the user timeout.

5.5.3 The SAN Models

We followed a compositional approach, built on basic sub-models of the network and user behaviors. The overall models, shown in Figure 5.18, are obtained by joining the random access model (indicated as “Network” in the figure), defined in Chapter 3, and different user models through the UltraSAN Join operator. As in the previous Section 5.4 the random access model, shown in Figure 5.19, has been extended with two places (Available and Unavailable), two timed transitions to model failure and repair of the architecture, and an output gate to trigger the outage effects [68]. Other modifications concern the places Served, new, and block that become common
Combining such basic sub-models, the overall model has been solved through simulation, in order to make quantitative estimations of a properly identified QoS indicator. The specific user characteristics this work focuses on are differences in the offered traffic and the possibility to deliberately abort a request after a prefixed time interval.

Actually two overall models are obtained, in accordance with the usage of the homogeneous user sub-model (Hom_User, Figure 5.18.a) and of the heterogeneous user sub-model (Het_User, 5.18.b). Note that in Figure 5.18.a the multiple boxes labeled Hom_User allow to represent a number of classes of homogeneous users (e.g., the class of users using the e-mail application, that of users using web-browsing, etc.). The Rep operator allows replication of the user sub-model, to properly represent the full population of system users.

Figure 5.20 shows the model which represents the behavior of a single user. The complete model on the left is the Hom_User model. Replacing the subnet enclosed in the dotted box with the separated subnet in the right part of the figure the Het_User model is obtained. The Hom_User model is used to represent classes of homogeneous users. It is configurable, so as to adapt to different classes of users, by changing the inter-request time, the number of LLC frames composing each user request and the user timeout. The model description of the Hom_User is the same as in Section 3.3.3 (see Figure 3.12) without some unnecessary SAN elements for the purpose of this Section (e.g places abort, abort_app, idle_app and the transition abort of figure 3.12). The Het_User model differs from the Hom_User one in allowing a user to vary, from one request to the next one, the length of his requests.
Figure 5.19: Modified random access model

Figure 5.20: User models
(in terms of number of LLC frames), the inter-request time and the timeout. In particular:

- The instantaneous activity select, allows to account for varying requests size and inter-request time; it has a case for each possibility allowed to the user. For simplicity, in the represented model they are just two (activated with probability $p$ and $(1-p)$, respectively), but could be easily extended to a higher number.

- Places to_req1, and to_req2, represent two possible choices for the inter-request time: short request more frequently or a long request less frequently, respectively. In the first case, the user idle time is modeled by the timed activities request1 and the number of LLC frames the user request consists of is controlled by the output gate send1. Similarly, for the second case the transition request2 and the gate send2 are used. Again, the model can be easily extended to include a higher number of possibilities.

### 5.5.4 Settings for the Numerical Evaluation

The analysis focuses on the expected served users per hour, under varying user, load and outages parameters. The purpose is to point out the difference between the evaluation conducted considering the effect of outages at system restart (in terms of the increased collision phenomenon due to accumulated requests during the system downtime), and the usual simplistic vision of an on-off system.

Being our study directed to the user viewpoint, in the following experiments we vary the user characteristics, and keep fixed the GPRS configuration. In choosing an adequate configuration, we took advantage of previous work [69], which focused on the performance of GPRS systems: 5 traffic channels (SPDCH) and 3 logical channels (PRACH) are assumed in the following analysis. Both user models have been considered, thus originating and studying two system scenarios. In the first scenario, the model Hom_User has been exercised. Specifically, the user population is partitioned into two classes. In the first class, users issue requests fitting in 1 LLC frame, with a shorter inter-request time (10 seconds), while requests issued by the other group require 6 LLC frames, and have a longer inter-request time (120 seconds). The two groups also differ in the maximum time a user is prepared to wait to have his request satisfied before giving up. Table 5.7 summarizes the numerical values assigned to such user characteristics for the two classes.

We are mainly interested in analyzing system behavior under critical conditions; therefore the experiments are conducted under high load situations,
Table 5.7: User parameters in Scenario 1

<table>
<thead>
<tr>
<th>User characteristic</th>
<th>User requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter_request_time</td>
<td>Request_size</td>
</tr>
<tr>
<td>class_1</td>
<td>Exponential dist.</td>
</tr>
<tr>
<td></td>
<td>$\lambda = 1/10$</td>
</tr>
<tr>
<td>class_2</td>
<td>Exponential dist.</td>
</tr>
<tr>
<td></td>
<td>$\lambda = 1/120$</td>
</tr>
</tbody>
</table>

Table 5.8: User parameters in Scenario 2

<table>
<thead>
<tr>
<th>User characteristic</th>
<th>User requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter_request_time</td>
<td>Request_size</td>
</tr>
<tr>
<td>$p$</td>
<td>Exponential dist.</td>
</tr>
<tr>
<td></td>
<td>$\lambda = 1/10$</td>
</tr>
<tr>
<td>$(1-p)$</td>
<td>Exponential dist.</td>
</tr>
<tr>
<td></td>
<td>$\lambda = 1/60$</td>
</tr>
</tbody>
</table>

from 90% to 110% of the whole system capacity. Users generating such load belong to class_1 and class_2 in a fixed proportion of 50% for each class. Consequently, the total number of users requesting network services varies in accordance with the considered total load. Outages are first considered of fixed duration (1000 seconds), but of varying frequency (by changing the availability value). Then, in a second experiment, the overall availability is kept fixed, while variations of the duration (and consequently, the frequency) of outages are explored, thus performing a sensitivity analysis to outage parameters. The distributions used to characterize the outage are the deterministic one for the duration, and the exponential one for the frequency. In the second scenario, the user model Het_User is exploited. Therefore, users are grouped in one, non-homogeneous class. Dynamically, a triplet consisting of a request size, an inter-request time and a timeout are assigned to users waiting for issuing a new request, on the basis of a probabilistic parameter $p$. Similarly to the previous scenario, two values, 1 and 6 LLC frames, are considered for the request size: with a probability $p$ the request size is 1 LLC frame, and with probability $(1-p)$ it requires 6 LLC frames. Then, the percentage of short and of long requests varies, in accordance with the value assigned to the parameter $p$. The workload is fixed to 100% of the system capacity. In order to keep it fixed at varying the probability $p$ of choosing a request size, the inter-request time assigned to short requests (1
LLC frame) is 10 seconds, while that associated to longer requests (6 LLC frames) is 60 seconds. The user timeout assumes the same values as in the previous scenario; Table 5.8 summarizes the user characteristics in scenario 2. Outages of fixed duration are considered, but having a variable frequency of occurrence (by varying the system availability). In all the experiments, results have been determined with 95% confidence interval and a relative confidence interval lower than 10%. As a final remark on the experiments setup, the selected parameter values are not tied to any specific real application, although short requests may be appropriate for typical web browsing applications while longer ones may fit categories of e-mails or filled forms on the web. Actually, despite the fact that real valued parameters are always preferable than artificial ones when available, studies which compare different situations in a relative way are very useful at system design stage in order to make appropriate decisions (on configurations, mechanisms, etc.), even if based on some arbitrary values. What is recommendable, in such cases, is some sensitivity analysis conducted on a range of values of the most impacting parameters. The value of our analysis stems from this observation, and its utility has to be regarded as a system designer support, rather than in providing absolute final estimations.

5.5.5 Numerical Evaluations

This Section presents the numerical evaluation performed solving the SAN model adopting a steady state analysis. The measure averaged number of served users per hour is derived from the SAN model simply by counting the number of times the transition request fires for the Hom_User and the sum of number of time transitions request1 and request2 fire for the Het_User (see Figure 5.20) in given amount of time, e.g. one hour.

A First Scenario

Two figures summarize the results obtained from experiments based on scenario 1.

The previous figures depicts the expected number of served users per hour for users belonging to class_1 and to class_2 (Figures 5.21 and 5.22, respectively). The measure has been determined at varying values of the GPRS availability, and for two values of the workload (90% and 110% of the system capacity, to explore critical situations from the overload point of view). In the figure, both the curves relative to the case where outages effects at system restart are accounted for and those in case of a simple on-off system are drawn. By comparing these two families of curves, it is immediate to
Figure 5.21: Served users per hour for class_1 users, at varying availability, for two values of the system load.

Figure 5.22: Served users per hour for class_2 users, at varying availability, for two values of the system load.
appreciate the extent of the error induced when performing the analysis using the simple model. Not surprisingly, looking at Figure 5.21, two phenomena are immediately captured. Firstly, there is an over-estimation in the served users when outage effects are not accounted for. Of course, the entity of such over-estimation depends on the availability value; for example, an availability of 0.95 leads to an error of about 10% when the load is 90%, which reduces at growing availability, becoming almost negligible from availability equal to 0.99. In this study, we keep fixed the mean time to repair (MTTR) after an outage occurs (which corresponds to the outage duration). This implies that, for the same load, at system restart the same accumulated requests have to be processed whichever be the availability assumed. However, higher availability means a lower number of outages experienced during the system lifetime, and therefore an enhancement of the evaluated measure. Secondly, all the curves degrade at increasing the system workload, and the error incurred relying on the “on-off” behavior increases. Similar comments apply to Figure 5.22; of course, being the study referred to the class of users issuing less requests of longer size, the derived numbers of served users per hour are significantly lower.

A second experiment in this scenario has been directed to investigate on the impact of different distributions of outages in terms of frequency of occurrence and duration, for a fixed availability value. In fact, availability
is an overall system measure that accounts for the total amount of time the system is up in the reference time interval. However, it is expected that different effects are produced on the QoS offered by the system, depending on whether that availability value has been determined by a higher number of shorter duration outages, or by a lower number of longer duration outages. The results are shown in Figure 5.23, which refers to an availability of 0.98, but where the time to repair the system after an outage (i.e., the outage duration) varies from 500 to 2500 seconds. From the figure, the following trend is observed: increasing the outage duration, which means decreasing the outage occurrence (being the availability at a fixed value), the expected served users per hour increase. The increase is noticeable in the first part of the figure (from MTTR = 500 sec to MTTR = 1000 sec, which means halving the number of outages), becoming almost constant in the second part. Therefore, outages occurring with lower frequency and lasting a longer time have less detrimental effects than more frequent, shorter ones. Note that values relative to class 1 are on the left Y-axis, while those of class 2 are on the right Y-axis.

**A Second Scenario**

The results concerning the second scenario are illustrated in following figures. In it, the curves of the difference between the number of expected served users per hour obtained using the simplistic “on–off” model and that accounting for outages are shown. Varying parameters are the probability $p$ of assigning length 1 LLC frame to an issued request, and the availability. The results are shown separately for short and long requests (Figures 5.24 and 5.25, respectively). In both parts of the Figure, it can be observed that there is always a reduction in the number of served users per hour when the outage effects are considered, and such reduction is higher when lower availability values are considered. (Note that the MTTR, i.e. the duration of outages, is fixed; therefore lower availability means more frequent outages). Concerning the variations of the parameter $p$, the trend is that the highest is the probability of issuing requests of a certain type (between short and long), and the highest is the served users reduction for that type of requests. In fact, in Figure 5.24 the highest reduction is in correspondence of $p = 1$, which implies all requests are 1 LLC frame. In Figure 5.25, instead, the highest reduction is observed for $p = 0$, which means all requests require 6 LLC frames (being $1-p$ the probability of issuing long requests). In this last figure, the point corresponding to $p = 0$ seems to be a singular point, which does not respect this trend. Actually, this behavior could be imputed to the simulation process, since the simulation results of the two values of served
Figure 5.24: Reduction in the expected served users per hour without accounting and accounting for outage effects for short requests

users per hour originating this point have not disjoint confidence intervals.

5.6 Conclusions

In this Chapter we conducted the analysis of GPRS by providing a modeling approach to better understand the effects of outage periods on the service provision. The goal was to achieve accurate availability estimation and improve the behavior analysis of such systems, to gain insights on the user perception of the QoS. A modeling approach has been followed, adopting the powerful modeling capabilities of Stochastic Activity Networks. Starting from a quite classical availability study of GPRS, based on the dependability of the various components of the GPRS infrastructure, the network dependability figures are combined into a very detailed service model that is used to analyze the overload effect that GPRS has to face after outages. The result of this modeling is an accurate availability analysis, which goes beyond the classical network perspective that is commonly taken when studying the availability of telecommunications systems, by including in the analysis a user and application perspective of the dependability of GPRS services. A typical GPRS configuration has been deeply analyzed and evaluated, in terms of a few identified QoS indicators, namely the number of served users per hour.
Interesting results have been observed, which can be fruitfully exploited in devising GPRS configurations adequate to maintain an acceptable QoS also in critical, overload conditions. Additionally, some sensitivity analysis has been performed at varying the frequency and duration of outages periods and for different values of network availability. Again, the objective of this study has been to enrich the knowledge of the impact of the outage phenomenon, in order to better cope with it and improve user satisfaction.
Chapter 6

Database Audit Modeling

6.1 Introduction

This Chapter deals with database maintenance (by means of audit operations) analysis with reference to the communication systems. Two main factors characterize these systems, which impact on the definition of efficient maintenance policies: i) short-persistence of most of the data stored in the database (typically, of the same duration of the user call), and ii) the highly dynamic evolution of the environmental conditions (e.g., varying number of active calls) and the changes over time of the requirements and of the services offered from these systems.

This work aims at investigating on appropriate tuning of audit operations, so as to find optimal balances between contrasting requirements, namely satisfactory database availability and low overhead due to audits. For this purpose, a methodology to analyze the behavior of the database under scheduled maintenance is here suggested. Analytical models, essentially based on Deterministic and Stochastic Petri Nets (DSPN) and the identified dependability indicators, are here presented.

6.2 General Assumptions on the System Context

For the purpose of our study, we assume that a set of audit procedures to cope with data corruption are provided, each characterized by a cost (in execution time) and coverage (as a measure of its ability to detect and/or correct wrong data). From the point of view of coverage, we distinguish between partial audits, characterized by a coverage lower than 1, and complete
audit, which performs complete checks and recovery such that, after its execution, the system can be considered as good as a new one. The considered audits are activated at pre-determined time intervals, in accordance with a maintenance strategy performed by an audit manager. In fact, an audit manager selects the part of the database to check/recover, the detection/recovery scheme to apply, and the frequency with which each check/recovery operation has to be performed. The audit manager is therefore responsible for applying the maintenance strategy to cope with database corruption and therefore preventing system unavailability. To set up an appropriate maintenance strategy, the audit manager would need some support, which helps it in evaluating the efficacy of applying different combinations of the available audit operations. In this work, we focus on such evaluation component (strategy evaluator), by developing a methodology to proper tuning of audit operations. In Figure 6.1, the logical structure of the database subsystem and of the involved components is shown.

Records of the database tables also include fields that are used to reference records belonging to other tables. Such reference fields (pointers) have a dynamic content. Whenever a call is set up, a set of linked records is inserted in the database; these records store all the data relevant for the establishment and management of the on-going calls. Records allocated to store the information on a specific call are released when a call ends. The specific set of relations that identify the linked structure of the database defines the dependency scheme. A pointer may fail in two ways: out of range, i.e., its value incorrectly assumes the value of a memory location outside the database tables, or in range, when it wrongly points to a location memory inside the tables’ space. The latter kind of fault shows more dangerous in the
general case, since a record belonging to another dependency is erroneously deallocated; we therefore say that an in range fault generates a catastrophic failure, while an out of range fault results into a benign failure. In addition, although the single out of range fault is not catastrophic, its repeated occurrence (above a pre-fixed threshold) leads to a catastrophic failure. After a catastrophic failure, the system stops working. In this work, we concentrate on maintenance policies for enhancing pointer correctness, which is undoubtedly very critical for the application correctness; however, our approach is general methodology which can be easily adapted to take into account different specific database information.

6.3 A Methodology to Fine-Tuning of Audit Operations

Our goal is to identify a methodology to model and evaluate the relevant dependability attributes of scheduled audit strategies in order to derive optimal maintenance solutions. The main aspects of such a methodology are:

1. The representation of basic elements of the system and the ways to achieve composition of them;

2. The behavior of the system components under fault conditions and under audit operations to restore a correct state;

3. The representation of failure conditions for the entire system;

4. the interleaving of audits with on-going applications and their relationships;

5. The effects of (combinations of) basic audit operations on relevant indicators for the system performance, in accordance with application requirements.

Our approach is based on Deterministic and Stochastic Petri Nets (DSPN) (see section 1.7.8). Specifically, in accordance with the points listed above, we defined general models which capture the behavior of the database and of the maintenance policy checking it, to be easily adapted to specific implementations of databases and audit actions. The defined models allow investigating on the most relevant aspects in such system, related to both the integrity of the database and the overhead caused by the audit activities. For the analysis purpose, the basic elements of the database are the pointer fields of
the tables. In order to compact the basic information, one can represent in
the same model structure the pointers belonging to database tables which:

1. Have the same failure rate;
2. Share the same audit operations, applied at the same frequency.

We call the tables whose pointer fields share such characteristics as *homoge-
neous set*. Such compactness process has to be carefully performed in
accordance with the set of maintenance policies to be analyzed. To represent
the process of generation of pointers and of their next deletion at the end
of the user call, one needs to model also the applications working on the
database. This way, the events of system failure caused by erroneous point-
ers in dependencies at the moment of the end of a call are also captured.
Finally, the complete maintenance strategy has to be modeled, in the form
of alternation of pure operational phases with others where applications and
audits run concurrently. The presentation of such general models, as well as
the interactions among them, follows in the next section.

### 6.4 Modeling of Maintenance Policies

Before presenting the models, the relevant figures of merit defined for the
analysis purpose and the assumptions made in our study are described. In
performing the system analysis and evaluation, we consider that the system
works through missions of predefined duration [14].

#### 6.4.1 Measures of Interest

To our purpose, two measures have been identified as the most sensible
indicators, and the developed models have been tailored to them.

1. The reliability that should be placed on the database correctness, ex-
pressing the continuity of service delivered with respect to system spec-
ifications [7]. Actually, to better appreciate the effect of maintenance,
we will evaluate the unreliability, as a measure of the probability of not
surviving a mission of a pre-fixed duration.

2. A performability measure [70], which shows appropriate to evaluate
whether a certain maintenance strategy is “better” than another. Nec-
essary to performability is the definition of a reward model; we use
here, by way of example, a simple additive reward model that fits our
mission-oriented systems. We assume that a gain $G_1$ is accumulated
for each unit of time the system spends while performing operational phases, and a value $G_2$ is earned for each unit of time while audit operations are in execution, with $G_1 > G_2$. Finally, a penalty $P$ is paid in the case of failure, again for each time unit from the failure occurrence to the end of the mission.

### 6.4.2 Assumptions

The models and analysis have been developed under the following assumptions:

1. Pointers corrupt with an exponential rate $\lambda_c$. Pointer faults occur independently from each other, so the corruption rate for a dependence is the sum of the corruption rates of each pointer involved in that dependence;

2. Audit operations and applications share the same processor(s); when audits are in execution, a reduced number of user calls can be satisfied. The entity of such reduction, being related to audit operations, may vary significantly;

3. Audit operations are characterized by a coverage $c$, indicating the audit’s probability of successful detection/correction. Intuitively, the higher is $c$, the more complex (and time consuming) is the corresponding audit;

4. According to the kinds of pointer failure (i.e., *in range* or *out of range*), catastrophic or benign failures are possible: several benign failures lead to a catastrophic failure (as already discussed in Section 6.2);

5. Each active user call involves an element (record) in each database table.

6. The maximum number of pointers that can be corrupted simultaneously is limited to three. This assumption is necessary to alleviate the problem of the explosion of the state space dimension of the model carrying out the model solution. An evaluation of the small impact of this assumption in our analysis is given in Appendix A (Section A.2).

### 6.4.3 The Models

Exploiting the multiple–phased structure of our problem, we developed separate models to represent:
Figure 6.2: Model of the maintenance strategy

- The behaviour of the system through the alternation of operational and audit phases;
- The failure/recovery process of the system components.

Figure 6.2 shows the model of a generic maintenance strategy. It represents the alternation of a (variable) number of operational phases ($Op1, \ldots, Opn$) and audit phases ($Ma1, \ldots, Man$), determining a maintenance cycle, which is then cyclically re-executed. Only one token circulates in the net. The deterministic transitions $TOp1, \ldots, TOpn$ model the duration of operational phases, while the deterministic transitions $TMa1, \ldots, TMan$ model the duration of the corresponding audit phase. The places $S1, \ldots, Sn$ and the instantaneous transitions $TS1, \ldots, TSn$ allow to complete the recovery act on in the homogeneous set is (described later) before a new phase starts.

The main elements of the application sub-net, shown in Figure 6.3, are:

- The place $Call_{active}$ contains the number of the ongoing calls.
- The place $Corrupted$ contains the number of out of range corruptions of a dependence (benign failures); one token in the place $Failed$ represents the catastrophic failure of the system.
- The instantaneous transition $T_{active}$ allows updating the number of tokens in the homogeneous set: whenever a call is set-up, represented by token moving from $Call$ to $Call_{active}$, a token is added in the place $Table$ of each homogeneous set. The exponential transition $T_{idle}$ represents the duration of a call. When the system is in an operational phase, that transition fires with rate $\mu$; during an audit phase the rate is $x * \mu$, where $0 \leq x \leq 1$ accounts for the percentage of the power processing lost during an audit phase with respect to an operational one.
- The instantaneous transitions $I_{to\_S}$, $I_{to\_C}$, and $I_{to\_F}$ model the behavior of the database when a call ends. The choice of which of them fires depends on the marking of the places $actived$ and $failed1$ (out of range) or $failed2$ (in range) in the representation of a homogeneous set sub-net (see Figure 6.4).
Figure 6.3: The application model

Figure 6.4: The model of a homogeneous set
CHAPTER 6

Figure 6.4 shows the model of a homogeneous set, i.e., of the pointers belonging to database tables having the same failure rate and subject to the same audits, with the same frequency. The sub-nets of the application and of the homogeneous set have to be connected together, since pointers are created and deleted by user calls. The meaning of the main elements in Figure 6.4 is:

- The firing of the exponential transitions $T_{cor}$ models a pointer corruption. The instantaneous transitions $T_{out}$ and $T_{in}$ move a token in the places $Out$ and $In$ respectively to distinguish if a given pointer is corrupted out of range or in range with probability $p$ or $1 - p$, respectively.

- During a maintenance phase transitions $rec_{.out}$ and $rec_{.in}$ are enabled according to the audit specifications.

- The instantaneous transitions $no_{.ok.out}$, $ok_{.out}$, $no_{.ok.in}$, $ok_{.in}$ model the recovery actions performed at the end of audit phases. They are enabled when there is a token in the places $Sn$ of the maintenance submodel. The success or failure of a recovery is determined by the coverage $c$ of the applied audit.

- When a call ends, a token (a pointer) will leave the homogeneous set sub-net. In a probabilistic way and on the basis of the marking of the places $failed1$, $failed2$, and $activated$ the decision is made on whether the dependence associated with a call is corrupted (out of range or in range) or not. The instantaneous transitions $I_{to.S}$, $I_{to.C}$, and $I_{to.F}$ of the application sub-net (see Fig 6.3) operate such choice.

- The instantaneous transitions $to.I$, $to.I1$, $to.I2$, $to.I3$, and $to.I4$ are enabled when transition $T_{Idle}$ of the application sub-model fires and a token is moved in the place $PutOut$.

- The instantaneous transitions $flush_{activated}$, $flush_{failed1}$, and $flush_{failed2}$ fire when there are no tokens in the place $Idle$ and after the instantaneous transitions $I_{to.S}$, $I_{to.C}$ and $I_{to.F}$ of the application sub-net.

From the DSPN models, the measures we are interested in are derived as follows:

1. The Unreliability is the probability of having one token in the place $Failed$ (in the application model) or a given number of tokens in the place $Corrupted$. 
2. The Performability is evaluated with the following formula:
\[ G_1 \times \{\text{Operational time while the system works properly}\} + G_2 \times \{\text{Audit time while the system works properly}\} - P \times \{\text{Time while the system is failed}\}. \]

6.5 Case Studies

To illustrate the usefulness of our approach and to give the reader an idea of the relevance of our analysis, two case studies are set-up and evaluated. In this Chapter the case studies are introduced. Their analysis are in Chapter 7.

6.5.1 Case Study 1: Single Audit Strategy Evaluation

We consider a database supporting a hypothetical telephone system, to which both partial and total audits are periodically applied. The defined maintenance strategy consists in alternating partial checks on different sets of dynamic data (pointers) with operational phases for a certain number of times, after which a complete audit is executed which resets the database status to the initial conditions. We are interested in evaluating the unreliability
and performability between two complete audits; it is then straightforward to make forecasts on the system for any chosen interval of time. By applying our methodology and composing the model elements defined in the previous section, the model instance for our case study is derived, as sketched in Figure 6.5. The upper part of the model represents the maintenance strategy, which encompasses two operational phases interleaved with two executions of the some partial audit on two non-disjoint sets of data. Therefore, three homogeneous sets of pointers \((A, B \text{ and } C)\) are defined in the lower part of the model. The relationships with the application model are shown in the right side of the Figure 6.5. Moreover, has been supposed that there exists two cyclic dependencies between the homogeneous set, e. g. \(AB\) and \(BC\). Thus, the set \(B\), being shared by both dependencies, is the most critical and the corruption of pointers in it has a higher impact. \(B\) is also supposed to be more frequently checked than \(A\) and \(C\) because more accessed by the application.

### 6.5.2 Case Study 2: Comparison between Different Audit Strategies

A set of audit procedures is also defined: \(procedure1\) has the ability to detect and recover an out of range error in a dependency while \(procedure2\) is more powerful: it detects and recovers an in range error. This power is paid in terms of execution time: \(procedure2\) is slower than \(procedure1\). On such database structure a few different audit strategies shown in Figure 6.6 are defined and will be evaluated. Strategy (i) cyclically performs two operational phases interleaved with two audits ones. The audit phases always use \(procedure2\) applied to dependence \(AB\) first and \(BC\) later.

In strategy (ii) the first audit phase checks the entire database using \(procedure1\) while the second phase uses \(procedure2\). Strategy (iii) has 4 operational and 4 audit ones. The four audit procedures are as follows: \(procedure1\) is used to check dependence \(AB\), then \(procedure2\) is invoked on \(BC\), further \(procedure2\) is used on \(AB\) and, finally, \(procedure2\) works on \(BC\). Comparing the effectiveness of these strategies is not trivial at all without the help of analytical analysis. In fact, a lot of factors and parameters play a role in the database behavior: the most important are the mean duration of a call \(T_{Call}\), the duration of the procedures 1 and 2, referred here as \(T_{Proc1}\) and \(T_{Proc2}\), respectively, the pointer failure rate \(\lambda_c\), the duration of an operational phase \(T_{op}\), the coverage \(c\) of the audit procedures, the probability \(p\) that a failed pointer is corrupted out of range \((1 - p\) is the probability of in range corruption), and the \(Time\), at which the analysis has to be carried out.
Figure 6.6: Evaluated strategies
6.6 Conclusions

In this Chapter the modeling framework for the evaluation of database maintenance has been defined.
Chapter 7

Database Audit Evaluation

7.1 Introduction

Firstly, the analytical models and case studies introduced in the previous Chapter are here solved in terms of the identified dependability indicators. A sensitivity analysis wrt to the most affecting internal and external parameters is also performed. Second, a learning approach is presented to dynamically adapt the maintenance policy at varying database and environmental parameter values leading to select, in each time period, the optimal maintenance policy.

7.2 Strategy Evaluation

In this Section the case study 1 defined in Section 6.5.1 of the previous Chapter is evaluated.

7.2.1 Settings for the Numerical Evaluation

The variable parameters involved in our numerical evaluations and their description are summarized in Table 7.1. Notice that we observe the system behavior between two complete audits; this interval of time is indicated with the variable \textit{Time} in such Table.

7.2.2 Numerical Evaluations

The derived models are solved by the DEEM tool [25], which provides an analytical transient solver. DEEM (DEpendability Evaluation of Multiple
Table 7.1: Relevant parameters and their default values

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_c$</td>
<td>Pointer corruption rate</td>
<td>$5 \times 10^{-7} - 5 \times 10^{-8}$ per seconds</td>
</tr>
<tr>
<td>$T_{op}$</td>
<td>Duration of an operational phase</td>
<td>60 - 300 seconds</td>
</tr>
<tr>
<td>$T_M$</td>
<td>Duration of a maintenance phase</td>
<td>20 sec.</td>
</tr>
<tr>
<td>$p$</td>
<td>Probability that a failed pointer is</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>corrupted <em>out of range</em></td>
<td></td>
</tr>
<tr>
<td>$c$</td>
<td>Coverage factor of partial audits</td>
<td>0.8 - 0.999</td>
</tr>
<tr>
<td>$P$</td>
<td>Penalty of the reward structure</td>
<td>50 - 1000</td>
</tr>
<tr>
<td>$G_1$</td>
<td>Gain during correct system functioning</td>
<td>10</td>
</tr>
<tr>
<td>$G_2$</td>
<td>Gain during audit operations</td>
<td>5</td>
</tr>
<tr>
<td>$T_{time}$</td>
<td>Time interval between two complete audits</td>
<td>2 hours</td>
</tr>
<tr>
<td>$N$</td>
<td>Max number of user calls concurrently active</td>
<td>100</td>
</tr>
<tr>
<td>$T_{Call}$</td>
<td>Call duration</td>
<td>5 min.</td>
</tr>
<tr>
<td>$B_F$</td>
<td>Benign failures necessary to determine</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>a catastrophic system failure</td>
<td></td>
</tr>
</tbody>
</table>

phased systems) is a tool for dependability modeling and evaluation, specifically tailored for multiple phased systems and therefore very suitable to be used in our context. Figure 7.1 shows the performability as a function of the duration of the operational phase, for different values of the penalty associated to the failure condition of the system. For the chosen setting, it can be observed a noticeable influence of such penalty factor $P$ on the resulting performability. When $P$ is just a few times the value of $G_1$ (i.e. the gain in case the system is fully and correctly operating), increasing $T_{op}$ brings benefits to the performability. This means that in such a case, the main contribution to the performability is given by the reward accumulated over operational phases. However, for $P$ from 200 to 300, an initial improvement can be observed, which is then lost, although the performability degradation is not dramatic. When $P$ is two order of magnitudes higher than $G_1$, the cost of a failure is so big that increasing $T_{op}$ (which implies a higher risk of failure) results in a performability detriment.

Figure 7.2 shows the performability keeping fixed the reward structure and at varying values of the coverage of the audit procedure and the length of the operational phase. Two effects can be immediately noticed. First, as expected, the performability improves with growing values of the coverage.
Figure 7.1: Performability at varying of $T_{op}$ and Penalty

Figure 7.2: Performability at varying of $T_{op}$ and Coverage
Figure 7.3: Performability at varying $T_{op}$ and $\lambda_c$

Figure 7.4: Unreliability at varying $c$ and $T_{op}$
Second, it can be observed a “bell shape” of the curves: the performability grows at growing values of the duration of the operational phase till a maximum value of $T_{op}$ after which the trend inverts. This is due to the fact that the higher reward obtained during a longer operational phase is at first the dominant factor in determining the performability, but lengthening $T_{op}$ also means exposing the system to a higher risk of failure, and the penalty $P$ to be paid in such a case becomes the most influencing parameter of the performability in the second part of the Figure 7.2. Figure 7.3 completes the analysis of the performability, at varying values of $T_{op}$ and for three different values of the pointer failure rate. The impact of $\lambda_c$ on the performability is noteworthy, and a behavior similar to that in Figure 7.1 is observed. Figure 7.4 shows the behavior of the unreliability at varying values of the coverage and for several values of $T_{op}$. Of course, the unreliability improves at increasing both the audits frequency (i.e., small $T_{op}$ and the coverage of the audits). It can be noted that same values of the unreliability can be obtained by adopting audits with a higher coverage or applying more frequently audits with a lower coverage. The analysis just discussed give a useful indication about the tuning of the major parameters involved in the database system. The optimal trade–off between the frequency of the audits and the investment to improve the coverage of the audits can be found, to match the best performability and dependability constraints.

7.3 Comparison between Different Strategies

In this Section the case study 2 defined in Section 6.5.2 of the previous Chapter is evaluated.

7.3.1 Settings for Numerical Evaluation

In the numerical evaluation reported here, we considered $T_{Call} = 5$ min, $T_{Proc1} = 15$ seconds, $T_{Proc2} = 30$ seconds, $\lambda_c = 5 \times 10^{-8}$ errors per second, $c = 0.95$, $p = 0.98$, and $Time = 5$ hours. With regard to the reward structure, we considered $G_1 = 10$, $G_2 = 5$, and $P = 500$ and we defined the unreliability as the probability of having one pointer failing in range failure, i.e. an ending application processes an in range corrupted pointer, or three accumulated out of range failures.
7.3.2 Numerical Evaluation

A comparison of the performability and of the unreliability of the system at varying values of $T_{op}$ for all the three strategies are shown in Figure 7.5 and 7.6, respectively. Figure 7.5 shows the performability; for low values of $T_{op}$ strategy (iii) (defined in Figure 6.6 of Chapter 6) is the best until $T_{op}$ reaches 3 minutes, for higher values the best is strategy (i), while strategy (ii) is always the worst. Figure 7.6 shows the curves of the unreliability and we can see that all the strategies show very good values for low values of $T_{op}$. It is noteworthy that strategy (iii) has a high performability despite it has at the same time the worst reliability. This means that the advantages due to the shorter maintenance duration with respect to the others strategies outweigh the penalty due to a lower reliability. We performed also a sensitivity analysis wrt. the pointer failure rate and the coverage of the data audits. We observed that the performability is greatly affected by the failure rate having also different effects on how the strategies perform, so that the best strategy depends on its value. In general, this modeling shows interesting results in the identification of an optimal trade–off between the cost of the audit procedures and the gain of availability of the system and for a comparison among audit strategies.
Towards an Optimal Maintenance Policy through the Reinforcement Q-Learning Approach

In this Section, a reinforcement Q-Learning [71][72] approach is suggested for a dynamic selection of the maintenance policy at varying database and environmental parameter values leading to select, at each given time period, the best maintenance policy. At first an overview is given on the reinforcement Q-Learning for an optimal behavior policy. Second, the suggested reinforcement Q-Learning approach is presented and discussed. Indeed, in this approach, an intelligent software agent must learn to maximize the performance (in our case in terms of performability and unreliability) of a wireless communication system using an appropriate maintenance policy.

7.4.1 Overview on Reinforcement Q–Learning for an Optima Behavior Policy

Learning from interaction is a foundational idea underlying nearly all theories of learning. Indeed, whether a human is learning to drive a car or to hold a conversation, he is acutely aware of how his environment responds to what he does, and he seeks to influence what happens through his be-
behavior. Thus, reinforcement learning is much more focused on goal-directed learning from interaction than other approaches to machine learning. Indeed, reinforcement learning is learning what to do (how to map situations to actions) so as to maximize a reward signal. The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics trial-and-error search and delayed reward are the two most important distinguishing features of reinforcement learning. One of the most widely used variants of reinforcement learning is Q-Learning. In this on-line reinforcement learning paradigm, the agent incrementally learns an action/value function \( Q(s,a) \) that it uses to evaluate the utility of performing action \( a \) while in state \( s \). Q-Learning leads to optimal behavior, i.e., behavior that maximizes the overall utility (performance) for the agent in a particular task environment \([73]\). The agent and environment interact at each of a sequence of discrete time steps, \( t = 0, 1, 2, \ldots \). At each time step \( t \), the agent receives some representation of the environment's state, \( s_t \in S \), where \( S \) is the set of possible states, and on that basis selects an action, \( a_t \in A(s_t) \), where \( A(s_t) \) is the set of actions available in state \( s_t \). One time step later, in part as a consequence of its action, the agent receives a numerical reward, \( r_{t+1} \in R \), and finds itself in a new state, \( s_{t+1} \). The Figure 7.7 illustrates this agent-environment interaction from which an agent can learn, using reinforcement Q-Learning, to maximize the reward leading to an optimal behavior policy. Thus, reinforcement learning allows an agent (the learner and decision-maker) to use its experience, from the interaction with an environment, to improve its performance over time.

### 7.4.2 Investigations on Reinforcement Q-Learning Approach for an Optimal Maintenance Policy

In its general form, reinforcement learning can be applied to learn a task from scratch; as a consequence, the learning process may require a prohibitively time-consuming amount of exploration of the state-action space in order to find a good policy. Even worse, in such situations intermediate actions chosen by the agent may result in very poor performance, especially in the first stages. As already said, wireless communication systems are availability critical; this implies that maintenance actions have to be performed so as to ensure a threshold performance level in order for the supplier not to incur in significant economical loss. Therefore, in our system, an agent with
lls following a pure trial-and-error paradigm cannot be acceptable. Our suggested approach to dynamically adapt the maintenance policy at varying database and environmental parameter values by using reinforcement learning is depicted in Figure 7.8. In this approach, an intelligent software agent learns to select an optimal action among defined relevant actions, in each time period ($T_P$), from the two following interactions:

- Interaction with the DEEM tool which evaluates a model of the real environment to predict a reward from a given state defined by a configuration of the relevant parameters;

- Interaction with the considered wireless communication system, the real environment, the behavior of which is sensitive to different configurations of the relevant parameters, for each value configuration of these relevant parameters corresponds a relevant state of the system.

Then, this intelligent software agent is trained on-line, using reinforcement Q-Learning, to learn the optimal maintenance policy to maximize the reward gained from the system. However, having the learning process completely performed at run-time risks to expose the system to violation of the requirement on a minimum system performance, especially during the first period. A way to cope with this problem would consist in performing some off-line training on the basis of (rough) a-priori knowledge on the system and
environment conditions. With reference to wireless communication systems, such a-priori knowledge could be obtained, e.g., from previous experience with similar systems and from information on the users population in the area the system is called to work. Then, when the whole system is put in operation, the basic learning already acquired by the agent would help in issuing reasonably good actions towards the system in response to environmental changes. At the same time, the agent continuously exercises the DEEM tool, on the basis of the last fresh values of relevant parameters provided by the real environment at each $T_P$ instant, thus refining its system knowledge towards the identification of the optimal maintenance policy. It is noteworthy to underline in this solution the utility provided by the environment model offered by DEEM, both during the off-line and the on-line activity of the software agent. Especially on-line, it helps exploring the utility of different maintenance solutions on the basis of knowledge collected so far, and to issue the best answer to the real environment, exploiting the agent’s knowledge on the real environment gained through the model. In this way, no exploration is done on the real environment, thus reducing the risk of actions negatively affecting the quality of service of the wireless communication system. Resorting to the same example the analysis of the previous Sections is based on, main relevant parameters representing changes in the real environment are: i) the mean number of calls $N_{\text{Call}}$ (Parameter_0, in Figure 7.8), and ii) the pointer failure rate $\lambda_c$ (Parameter_1). The time durations of the operational phases, the given audit operations, or different combinations of the time durations of the operational phases and given audit operations can be defined to be the relevant actions. For instance, four different time durations, $T_{op}$, of the operational phases can be considered as the relevant actions (Action_0, Action_1, Action_2, and Action_3) among which the optimal one must be selected in each time period ($T_P$). The system rewards evaluated by the environment model can be again mapped into performability and/or unreliability measures. The process of selecting a particular action for execution from a given set of possible actions depends on the current state of the real environment as well as the internal situation, $Q$ values, of the intelligent software agent. For example, for a given state $S$ corresponds $Q[S][\text{Action}_0]$, $Q[S][\text{Action}_1]$, $Q[S][\text{Action}_2]$, and $Q[S][\text{Action}_3]$; then the maximal $Q$ value determines the selection of the corresponding action, i.e., the optimal action.

This approach is still under investigations, and a number of problems have still to be understood in order to finalize the study. However, the results obtained from these first investigations seem to be quite promising. In particular, the main advantages of the presented reinforcement Q-Learning approach are:
Database Audit Evaluation

Figure 7.8: Intelligent software agent learning from interactions with the considered wireless communication system and the DEEM tool
• To keep under control both the early stage of learning, which is often deficient, and the random exploration of Q-Learning, with benefits on the performance of the wireless communication system;

• The ability of the intelligent software agent to reach the optimal maintenance policy is guaranteed by Q-Learning ensuring then the maximization of the performance (performability and unreliability) of the wireless communication system over its life-cycle, i.e., an efficient quality of service.

7.5 Conclusions

This work has focused on maintenance of dynamic database data in a communication system.

To achieve a good trade-off in terms of overhead and efficacy of the maintenance, it is necessary to properly choose which audit operations are to be applied and how frequently they should be used. We proposed a modular approach to model and evaluate the relevant dependability attributes of scheduled audit strategies. Our approach is based on Deterministic and Stochastic Petri Nets (DSPN) and on the DEEM tool. The major impact of this study is the definition of a general model for the evaluation of the effectiveness of audit strategies. Paramount criteria for our work have been the extensibility and flexibility in composing the audit strategies. Of course, in order for our models to be really useful for the selection of proper order and frequencies of audit operations, input parameters such as cost and coverage of the check and failure data are necessary and should be provided.

Aiming at deriving optimal maintenance strategies a learning approach is presented to dynamically adapt the maintenance policy at varying of database and environmental parameter values leading to select, in each time period, the optimal maintenance policy. The approach is still under investigations, and a number of problems have still to be understood in order to finalize the study. For example, one important issue to guarantee the success of the suggested approach is how to derive an efficient state space representation. Neural Network methods seem at the moment a viable solution to overcome this problem. Also, the integration of off-line training with the on-line refinement learning activity by the software agent has to be more deeply analyzed.

Simplified database audit scenarios have been defined, on which our modeling effort has been exercised to illustrate the practical utility of our modeling approach. However, the learning approach for dynamic adaptation of
maintenance policy still deserves further investigations and a (simplified) case study for this approach should be highly useful to better understand the adequacy of the approach in our context and its possible limitations.
Conclusions

There exists needs by operating companies to integrate dependability as a natural part of planning of telecommunication networks. Thus, dependability and performability evaluation are a crucial mean to help operating companies and system constructors and integrators to reduce the risk of customer’s complaints and loss of business to competitors as well the risk of unnecessary investments. In such systems the most impacting dependability requirements, both on user satisfaction and operators revenue, is the availability of the service. Actually, availability is an important aspect of telecommunication systems and it is tightly linked to the concept of Quality of Service (QoS).

This work concentrated on dependability and performability evaluation of wireless communication systems, particularly of the random access procedure to GPRS (General Packet Radio Service) at one side and data maintenance of databases supporting telecom systems at the other side. A modeling approach (both simulative and analytical) has been pursued throughout this thesis, based on Petri Net formalisms. The achieved results analyze the behavior of such systems in “normal” conditions and in presence of faults (in term of QoS related measures). Also, possible solutions to alleviate congestion phenomena and the effects of faults have been explored.

Seen as a whole, this work provides a modeling framework for evaluating dependability and performance measures of some aspects of telecommunication systems. A contribution of our approach is in the explicit separation between architectural, and user (services) concerns of a system. This methodology brings many advantages. Firstly, once the links among the above models are clearly identified and formalized, reuse/change of one or more submodels can be performed more easily than when the various models are not separated. Secondly, it is possible, for example, to evaluate comprehensive dependability and performability measures that are especially relevant to the end-user, in addition to specific measures that may be too detailed for the end-user but mainly of interest for the system manufacturer. Thirdly, the reusability of the submodels, allows comparing different architectures for
choosing the most suitable from the end-user perspective.

Concerning to GPRS random access procedure, our analysis spanned several aspects. Initially, the bare random access protocol to the GPRS network has been modeled. It allowed to explore the traditional QoS indexes (block probability, service time, and channels utilization) in “normal” system conditions (absence of faults) [69]. Then, we have introduced a characterization of the QoS perceived by the user. The users’ behavior is modeled in term of number of random accesses needed to accomplish their service requests, the frequency of these requests, and the time they are willing to wait for the service accomplishment. Moreover, a threshold mechanism has been implemented and evaluated to cope with congestion phenomena. The rationale behind this choice is to abort sessions that have a too low probability to be completed successfully within a given timeout. It has been shown that this mechanism leads to a substantial improvement of the overall quality of service by means of a better network utilization [74]. Then, the behavior of the random access procedure to GPRS in presence of faults (both during the outage and the recovery periods) has been analyzed. We also explored the benefits of a queue mechanism of the requests (that queue is not mandatory in the standard) in term of the previous measures [75]. After, a study of effects of outage on the users’ QoS perception has been carried out. This study allowed to distinguish between the traditional QoS at system level from what at user level both in “normal” conditions and in presence of faults [68]. Finally, an actual overall GPRS architecture has been modeled, from the BTS to the GGSN, for an accurate availability evaluation of the system. In fact, in this analysis the availability has been evaluated keeping into account the transient effects due to fault outage-recovery periods. On the other hand, classical availability evaluation is evaluated simply as the ratio between the time the system is working properly on the overall observation time [76], [77].

Concerning the database maintenance two main points have been investigated. Firstly, we have defined a methodology to identify an appropriate scheduling of database audit (maintenance policy). This methodology allowed to compare maintenance policies in term of different frequencies, order, and targeted fields of the database audits [78]. The major impact of this study is the definition of a general model for the evaluation of the effectiveness of audit strategies. Paramount criteria for our work have been the extensibility and flexibility in composing the audit strategies. Of course, in order for our models to be really useful for the selection of proper order and frequencies of audit operations, input parameters such as cost and coverage of the check and failure data are necessary and should be provided. Secondly,
in order derive an optimal maintenance strategies a dependability manager has been defined to dynamically adapt the maintenance policy at varying of database and environmental parameter values leading to select, in each time period, the optimal maintenance policy. This dependability manager utilizes the previously defined models as policy evaluator to take appropriate decisions [79], [80].

Finally, this thesis work offers relevant contribution to the understanding of impacts of faults on the performability of both the GPRS uplink radio access and databases supporting GPRS. These results can be useful for an accurate dimensioning of such complex systems which takes into account the effects of faults and congestion phenomena.

Concluding, the idea of dependability manager (introduced at the end of Chapter 7) based on the online solution of statistical Petri-like models, deserves to be further investigated. In particular, it represents a valuable opportunity in the field of scheduling maintenance and reconfiguration (as a fault treatment mean) of a system to cope with faults or new QoS constraints. Currently, we are pursuing this approach in the framework of the Caution++ IST Project “Capacity and network management platform for increased utilization of wireless systems of next generation” ([81] and related activities [82]).
References


REFERENCES


CONCLUSIONS

of Computer Systems, pages 245 – 246, Newport, Rhode Island, United States, 1992. ACM SIGMETRICS.


REFERENCES


REFERENCES


Appendix A

Further Modeling Details

In this section we illustrate some details of a few further models which have been omitted previously for the sake of simplicity.

A.1 Further Details of the SAN Model of Figure 5.6

Actually, the overall composed model of Figure 5.6 is represented in Figure A.1, where the new introduced subnets Counter and Measure are shown in Figure A.2 and A.3, respectively. These submodels are introduced to derive the identified measures of interest defined in Section 5.2.1. All submodels have in common the place UNAVAILABLE.

Figure A.2 implement a counter for deriving the mass probability function of the outage duration. Its details are explained in the following:

- The transition count_one, which is determinist and lasts one second, is
Figure A.2: *Counter* subnet

Figure A.3: *Measure* subnet
enabled by means of the input gate no_more when there is at least one token in place UNAVAILABLE. Tokens are accumulated in the place acc_cout up to the marking of UNAVAILABLE is greater than zero.

- When the marking of UNAVAILABLE becomes zero, the input gate no_more enables the transition no_more (which is very fast, with respect to count_one) that moves tokens from place acc_count to place acc_cont_app and resets the marking of place acc_count. In this way, it is possible measure the fraction of time that the marking of the place acc_count_app falls into a given interval, while the marking of place UNAVAILABLE is zero. To achieve the desiderata mass probability function of the outage durations remain to normalize the measure of the time that the marking of the place acc_count_app falls into a given interval for the time that marking of place UNAVAILABLE is zero.

Figure A.3 is explained in the following. Such submodel is devised to derive the number of faults per hour measure.

- Whenever an outage occurs (the marking of place UNAVAILABLE is greater than zero) and the marking of place already is zero, transition outage is enabled through the input gate inp_gate and puts a token in the place already (which is initially at zero).

- When the marking of UNAVAILABLE becomes zero and marking of already is one, transition outage2 is enabled and the marking of already is reset. The desiderata measure is obtained from the SAN model simply by counting the number of times the transition outage2 fires in given amount of time, e.g. one hour. This transition is assured to fire only one time whenever the marking of place UNAVAILABLE becomes greater than zero.

A.2 Model Validation in the Database Maintenance Problem

The database maintenance problem has been approached exploiting the DSPN modeling formalism and the model solution has been carried out by means of the analytical solver provided by DEEM [25]. A well known problem of Petri nets is the large dimension of the state space of the model describing complex systems; thus, some simplifying assumptions should be done to solve it. To avoid the problem of the state space explosion, the main assumption
we have done in our modeling approach concerns a limitation of the fault process of the pointers, e.g. the number of simultaneously corrupted pointers is assumed to be less or equal to 3. Figures A.4 and A.5 give an explanation of this choice and demonstrate the small impact that this assumption has in the model solution for the strategies of Figure 6.6. They are derived in the worst case for the fault process of the pointers, e.g. with the highest corruption rate of the pointers and the less frequency for the audit operations that we have considered, \( \lambda_c = 5 \times 10^{-7} \) and \( T_{op} = 5 \) min., respectively, during an assumed length of the mission of 5 hours. Figure A.4 shows the cumulated block time of the pointer fault process; we can see that is much less than one second at the end of the mission duration. Figure A.5 represents the block probability of the fault process of the pointers for the different maintenance strategies of Figure 6.6. All these pictures give an indication of the unavailability of the fault process, that seems to be negligible for all the evaluations carried out in Chapter 7. Notice that every \( T_{op} + T_M \) seconds (here \( T_M \) is the duration of a generic database audit and it depends on the audit strategy under consideration) the curves in Figure A.5 are discontinuous since a maintenance strategy is performed and some possible pointer failures have been recovered. A similar behavior can be observed also in Figure A.4, but is less evident because of the different scale of the x-axis.
Figure A.5: Block probability of the fault process of the pointers