A User-Perceived Availability Evaluation of a Web Based Travel Agency

Mohamed Kaâniche, Karama Kanoun, and Magnos Martinello[*]
LAAS-CNRS — 7 Avenue du Colonel Roche
31077 Toulouse Cedex 4 — France
{Mohamed.Kaaniche, Karama.Kanoun, magnos} @laas.fr

Abstract

A hierarchical modeling framework for the dependability evaluation of Internet-based applications is presented and illustrated on a travel agency example. Modeling is carried out considering four levels, namely: user, function, service and resource levels. The first level describes how the users invoke the application and the three remaining levels detail how the user requests are handled by the application at distinct abstraction layers. The user perceived availability measure considered in this paper takes into account the combined impact of performance-related failures and traditional software and hardware failures. For illustration purposes, several sensitivity analysis results are presented to show the impact on this measure of various assumptions concerning, e.g. the users operational profile, the travel agency architecture and the fault coverage.

1. Introduction

Growing usage and diversity of applications on the Internet make the issue of assessing the dependability of the delivered services as perceived by the users increasingly important. The Internet is often used for transaction based money critical applications such as online banking, stock trading, reservation processing and shopping, where the temporary interruption of service could have unacceptable consequences on the e-business [1-3]. Given the critical nature of such applications, the assessment of the user perceived quality of service is a key issue for e-business service providers and developers. Indeed, it is important for the developers to analyze during the architecture design phase how hardware, software and performance related failures of the infrastructure supporting the delivered services might affect the quality of service perceived by the users.

Internet based applications are implemented on largely distributed infrastructures, with multiple interconnected layers of software and hardware components, involving various types of servers such as web, application, and database servers [4, 5]. Three key players are typically involved in the provision of the services delivered by such applications: 1) the users (i.e., the customers), 2) the e-business provider (eBP), who implements the e-business functions invoked by the users; these functions are based on a set of services and resources that are internal to the eBP site(s) or are provided by external suppliers, and 3) the external suppliers.

Generally, the eBP has a full control of its own architecture. Therefore, a detailed dependability modeling and analysis of this architecture can be carried out to support design architectural decisions. However, only limited information is generally available to analyze the dependability of the external suppliers services. In this context, remote measurements can be used to evaluate some parameters characterizing the dependability of these services [6-9]. These parameters can then be incorporated into the models describing the impact of eBP component failures and repairs on the user perceived dependability.

The discussion above shows that several issues should be taken into account when modeling the user perceived dependability of Internet based applications. Due to the complexity of the target system and the difficulty to combine various types of information (users behavior, failure-recovery behaviors of the supporting infrastructure), a systematic and pragmatic approach is needed to support the construction of such dependability models [10]. In our work presented in [11], we proposed a hierarchical framework for modeling the user perceived dependability of e-business applications. Modeling is done in two steps: 1) identify the functions and services provided to the users and the resources contributing to their accomplishment, and characterize how the users interact with the application, and, 2) based on this, build model(s) to assess the impact of component failures and repairs on the quality of service delivered to the users.

This paper is aimed to illustrate the main concepts of this modeling framework using a web-based travel agency (TA) as an example. The objectives are: 1) to show how to apply our framework based on the decomposition of the target application according to four levels: user, function, service and resource levels, and 2) to present typical dependability analysis and evaluation results obtained from modeling, to help the service providers in making objective design decisions. The user perceived dependability measure takes into account the combined impact of performance related failures and traditional software and hardware failures. For illustration purposes, several sensitivity analysis results are presented to show

[*] M. Martinello is supported by a fellowship from CAPES-Brazil. This work was partially supported by the European Community (Project IST-1999-11825-DSoS)

0-7695-1959-8/03 $17.00 (c) 2003 IEEE

the impact on the user perceived availability of various assumptions concerning, e.g. the users operational profile, the travel agency architecture and the fault coverage.

Section 2 recalls the main concepts of the modeling framework. Sections 3 and 4 present the travel agency example and its hierarchical description. Section 5 gives some examples of dependability evaluation results. Finally, Section 6 concludes the paper.

2. Dependability modeling framework [12]

The information needed for dependability modeling and analysis is structured into four levels:

- The *user level* describes the user operational profile in terms of the types of functions invoked and the probability of activation of each of them.
- The *function level* describes the set of functions available at the user level.
- The *service level* describes the main services needed to implement each function and the interactions among them. Two categories of services are distinguished: those delivered by the eBP (internal services) and those provided by external suppliers (external services).
- The *resource level* describes the architecture on which the services identified at the service level are implemented. At this level, the architecture, and fault tolerance and maintenance strategies implemented at the eBP site are detailed. However, each service provided by an external supplier is represented by a single resource that is considered as a black box.

The dependability modeling and evaluation step is directly related to the system hierarchical description. This is illustrated in Fig. 1 where the dependability measure considered is availability. The outputs of a given level are used in the next immediately upper level to compute the availability measures associated to this level (denoted by $\mathcal{A}$(x) where x is a user, a function, a service or a resource). Accordingly, at the service level, each service availability is derived from the availability of the resources involved in its accomplishment. Similarly, at the function level, the availability of each function is obtained from the availability of the services implementing it. Finally, at the user level, the availability measures are obtained from the availability of the functions invoked by the users.

Various techniques can be used to model each level: fault trees, reliability block diagrams, Markov chains, stochastic Petri nets, etc. The selection of the right technique mainly depends on the kinds of dependencies between the elements of the considered level and on the quantitative measures to be evaluated. In Section 4, we mainly make use of block diagrams and Markov chains to evaluate the availability of the travel agency.

As regards the state of the art, the proposed hierarchical description builds on some of the concepts proposed in [13] to analyze the performance of e-business applications. However, as our framework focuses on dependability, we have adapted these concepts and refined them to fulfill the objectives of our study.

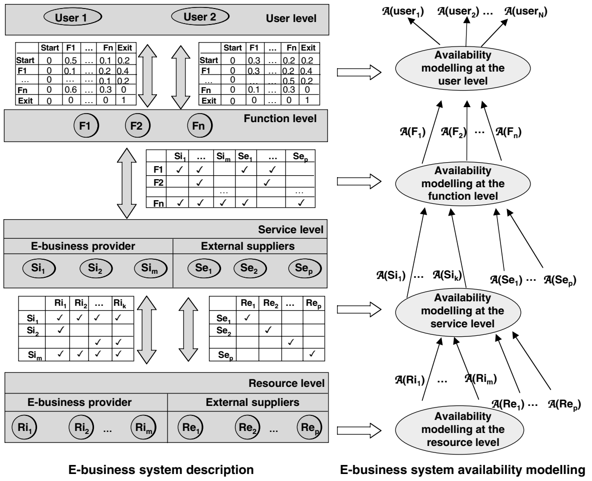
User 1
User 2
User level
Start
F1
…
Fn
Exit
Function level
F1
F2
Fn
Si1 … Sim Se1 … Sep
Service level
E-business provider
External suppliers
Si1 Si2 Sim
Se1 Se2 Sep
Ri1 Ri2 … Rik
Re1 Re2 … Rep
Resource level
E-business provider
External suppliers
Ri1 Ri2 … Rim
Re1 Re2 … Rep
A(user1) A(user2) … A(userN)
Availability modelling at the user level
A(F1) A(F2) … A(Fn)
Availability modelling at the function level
A(Si1) … A(Sik) A(Se1) … A(Sep)
Availability modelling at the service level
A(Ri1) … A(Rim) A(Re1) … A(Rep)
Availability modelling at the resource level
E-business system description
E-business system availability modelling

Figure 1. Hierarchical availability modeling framework

The evaluation of quantitative measures characterizing user-perceived dependability for web-based applications is widely recognized as highly important to faithfully reflect the impact of failures from the business point of view [14], However, there is still a lack of modeling examples illustrating how to address this issue. The work presented in this paper intends to fill this gap by illustrating on a simplified example how the modeling can be carried out and what kinds of practical results can be derived.

3. The Travel Agency (TA) example

The TA is designed to allow the users to plan and book trips over the web. For this end, it interacts through dedicated interfaces with several flight reservation systems (AF, KLM, …), hotel reservation systems (Sofitel, Holiday Inn, …), and car rental systems (Hertz, …).

The TA can be seen as composed of two basic components: the client side and the server side. The client side handles user's inputs, performs necessary checks and forwards the data to the server side component. The latter is the main component of the TA. It is designed to respond to a number of calls from the client side concerning e.g., availability checking, booking, payment and cancellation of each item of a trip. It handles all transactions to, and from, the booking systems, composes items into full trips, converts incoming data into a common data structure and finally handles all exceptions.

Starting from this very high-level description, we will further detail it according to the various aspects required for the hierarchical description. We will first focus on the function and user levels together, then the service and function levels before addressing the resource level.

3.1. Function and User levels

The behavior of the users accessing the TA web site is characterized by the operational profile example presented in Figure 2. The nodes "Start" and "Exit" represent the start and end of a user visit to the TA web site, and the other nodes identify the functions invoked by the users during their visit. For the sake of illustration, we have considered five functions for the TA example:

- *Home:* invoked when a user accesses the TA home page.
- *Browse:* the customer navigates through the links available at the TA site to view any of the pages of the site. These links include the weekly promotions, help pages, frequent queries, etc.
- *Search:* the TA checks the availability of trip offers corresponding to the criteria specified by the customer. A user request can be composed of a flight, a hotel and a car reservation. Based on the information provided by the user, the TA converts the user requests into transactions to several hotel, flight and car reservation systems and returns the results of the search to the user.
- *Book:* the customer chooses the trip that suits his request and confirms his reservation.
- *Pay:* the customer is ready to pay for the reservation fees for the trips booked on the TA site.

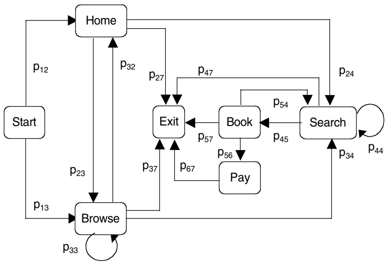

Figure 2. User operational profile graph

The transitions among the nodes and the associated probabilities p_{ij} describe how the users interact with the TA web site. A given class of users is defined by a specific set of p_{ij}. These probabilities can be obtained by collecting data on the web site (see e.g., [15]).

The operational profile defines all user execution scenarios (or shortly, user scenarios) when visiting the TA web site. Each scenario is defined by the set of functions invoked and the probability of activation of each function in the corresponding scenario. The “Start” and “Exit” nodes denote the beginning and end of a user scenario.

The identification of the most frequently activated scenarios gives useful insights into the most significant scenarios to be considered when evaluating the user perceived dependability. Indeed, the higher the activation probability of a given scenario, the higher its impact on the dependability perceived at the user level. Such measure is affected by the dependability of the functions, services and resources involved in this scenario.

Table 1 lists the user execution scenarios derived from Figure 2 and gives the probabilities of these scenarios (in terms of percentage), associated to two customer profiles (denoted as user class A and user class B). The notations {Home - Browse}*and {Search-Book}* mean that these functions are activated more than once in the corresponding scenarios, due to the presence of cycles in the graph.

User scenario	π_i, Class A	π_i, Class B
1: St-Ho-Ex	10.0	10.0
2: St-Br-Ex	26.7	6.6
3: St-{Ho- Br}*-Ex	11.3	4.2
4: St-Ho-Se-Ex	18.4	13.9
5: St-Br-Se-Ex	12.2	20.4
6: St-{Ho- Br}*-Se-Ex	7.6	9.7
7: St-Ho-{Se-Bo}*-Ex	3.0	4.7
8: St-Br-{Se-Bo}*-Ex	2.0	6.9
9: St-{Ho- Br}*-{Se-Bo}*-Ex	1.3	3.3
10: St-Ho-{Se-Bo}*-Pa-Ex	3.6	6.4
11: St-Br-{Se-Bo}*-Pa-Ex	2.4	9.4
12:St-{Ho-Br}*-{Se-Bo}*-Pa-Ex	1.5	4.5

St: Start Ho: Home Br: Browse Se: Search Bo: Book Pa: Pay Ex: Exit

Table 1. User scenario probabilities (in%)

For the class A profile, a high proportion of users are mainly seeking for information without a buying intention, whereas the class B profile is characterized by a higher proportion of users really seeking for booking a trip. Indeed, the percentage of transactions that end up with a payment of a trip is around 20% for user class B while it is almost 3 times lower for user class A. Moreover, it can be seen from Table 1 that, for the class B profile, 80% of user transactions lead to the invocation of the functions Search, Book or Pay. Such scenarios involve the external reservation systems in addition to the TA system. Therefore, the quality of the service offered by these reservation systems has a significant impact on the user perceived dependability. The percentage is lower (50%) when considering the class A profile.

These two examples of user classes are used in Section 4 to evaluate the user perceived availability.

3.2. Service and Function levels

The service level identifies the set of servers involved in the execution of each function and describes their interactions. This analysis requires a deep understanding of the business logic and the technical solutions implemented by the TA system provider.

For the sake of illustration, Table 2 gives a simplified example of mapping between the functions provided at the TA site, the internal servers directly controlled by the TA system provider and the external servers operated and controlled by external suppliers.

The external suppliers correspond to the flight, hotel, and car reservation systems that provide information on the corresponding items of a trip. Also, we assume that the TA provider uses the services of an external payment system for handling card-based transactions.

The internal services are supported by three types of servers: 1) Web servers that receive user requests and send back the requested data, 2) Application servers that implement the main operations needed to process user requests, and Database servers handling data related operations (for storing and retrieving information about flight, hotel and car reservation companies, as well as information on customer orders).

	Internal services			External services			
	Web	Appli-cation	Data base	Flight	Hotel	Car	Pay ment
Home	✓						
Browse	✓	✓	✓	✓	✓	✓	✓
Search	✓	✓	✓	✓	✓	✓	✓
Book	✓	✓	✓	✓	✓	✓	✓
Pay	✓	✓	✓	✓	✓	✓	✓

Table 2. Mapping between functions and services

The "Home" function execution involves the web server only. However, for the other functions several servers are required. In this case, it is necessary to analyze for each function the interactions among the servers involved and all possible execution scenarios (referred to as function scenarios). This is achieved through the interaction diagram dedicated to each function. Examples of interaction diagrams for the Browse, Search, Book and Pay functions are given hereafter.

<u>*Browse*</u>: Figure 3 describes the interactions among the servers involved in the accomplishment of the Browse function. The "Begin" and "End" nodes identify the beginning and the end of each function execution. Each path from the "Begin" node to the "End" node identifies one possible function scenario. The probability of activation of each scenario can be evaluated by taking into account the probabilities q_{ij} associated to the transitions involved in the corresponding scenario. Note that the probability of activation of non-labeled transitions is one.

We can identify three scenarios described as follows:

- 1→2→3: The user sends a request to the web server (node 2). The data requested is available in the local cache and returned back to the user (node 3).
- 1→2→4→5→6: The web server accepts the user request and sends it to the application server (node 4). In this case the requested data is not available in the local cache. The application server processes the request and returns a dynamically generated page to the web server (node 5). The latter is then forwarded to the user (node 6). The database is not involved in this case.
- 1→2→4→7→8→9→10: The application server requires some specific information that is on the TA database server (node 7). After the database server has answered the application server, the latter processes the user request (node 8) and sends the results to the web server (node 9). The latter generates an HTML page incorporating the corresponding outputs (node 10).

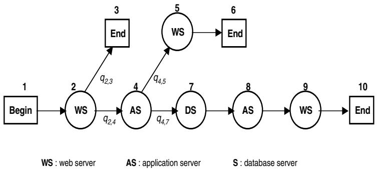

Figure 3. Interaction diagram of the "Browse" function

<u>*Search*</u>: The interaction diagram describing the execution of the Search function is decomposed into 9 stages (Figure 4). The input data provided in the search request issued by the user (node 1) are first processed by the web server WS (node 2). WS performs necessary checks, and then breaks down the user request into three individual requests corresponding to each aspect of the trip. If data is correct and in the right format, it is forwarded to the application server AS (node 4), otherwise an exception is sent to the user (node 3). AS uses the request information to formulate a query and asks the database server (node 5) for the list of booking systems to be contacted. Based on the answer received, AS sends a query (node 6) to the selected systems (identified by the Flight, Hotel and Car nodes). The AND operator means that the request is submitted to the three types of booking systems (nodes 7.a, 7.b, 7.c). The answers returned to AS are formatted (node 8) and sent to WS (node 9) that forwards them to the user (node 10).

The number of Flight, Hotel and Car reservation systems is not indicated in this figure. We assume that the TA always interacts with the same systems. A transaction is successful when, for each service (Flight, Hotel and Car reservation), at least one system responds.

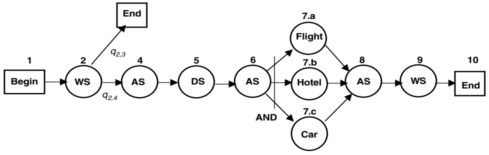

Figure 4. Interaction diagram of the "Search" function

<u>*Book:*</u> An example of interaction diagram of the *Book* function is given in Figure 5. The trip booking order received from the user through WS is processed by AS. Using the parameters embedded in the book order, AS interacts with the corresponding flight, hotel and car booking systems to book the selected trip. The booking references returned to the application server are then stored in the database, before a confirmation is sent to the user through the web server.

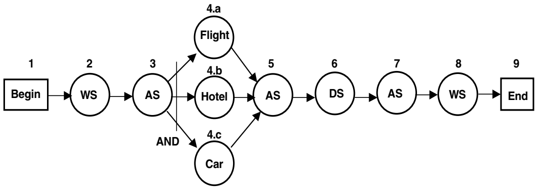

Figure 5. Interaction diagram of the "Book" function

<u>*Pay:*</u> The interaction diagram for the *Pay* function is presented in Figure 6. When a payment call is received through the web server, the booking data is first checked by the application server, then a call is sent to the payment server, for authentication and verification purposes, and also to accomplish the payment. Finally, the application server updates the information in the database concerning client orders, before sending a confirmation to the user.

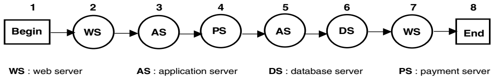

Figure 6. Interaction diagram of the "Pay" function

3.3. Resource level

The various services are mapped into the resources involved in their accomplishment. Therefore, we need to take into account the real hardware and software organization of the system. With respect to external services, as the architecture on which these services is not known, we associate to each external service a single resource that is considered as a black box. For internal services, it is possible to detail the organization of internal resources for which the architecture is known.

Various architectural solutions are possible for implementing the internal services, considering different organizations of the servers on the hardware support (e.g., dedicated hosts for each server, vs. multiple servers on the same host) or different fault tolerance strategies (non-redundant servers *vs.* replicated servers). Replicated servers can be located at one site or be geographically distributed at distinct sites. Also, fault tolerance can be applied to provide redundant accesses to the Internet or redundant communication links between internal resources. Additionally, the architecture solutions might be compared with regards to the maintenance strategy adopted by the TA provider (e.g., immediate *vs.* deferred maintenance, dedicated *vs.* shared repair resources).

For illustration purposes, we consider the two architectures presented in figures 7 and 8. We assume that the external resources are identical for both architectures. They correspond to Flight reservation, Hotel reservation, Car reservation and Payment. We assume that the flight, hotel and car reservation services are provided by respectively N_F, N_H and N_C components each.

The basic architecture (Figure 7) consists in allocating a dedicated host to each server and interconnecting these hosts through a LAN. The LAN is viewed as a single resource providing communication between the servers.

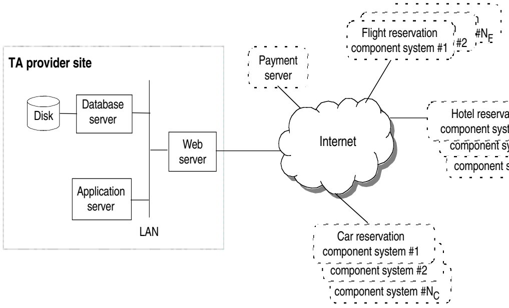

Figure 7. Basic architecture

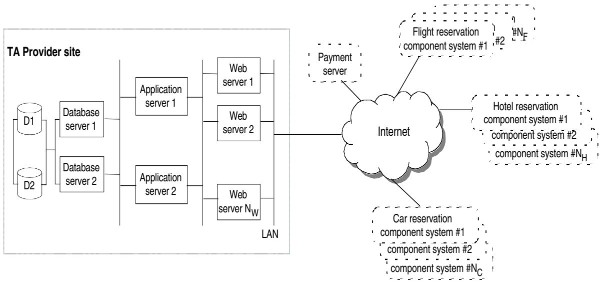

Figure 8. Redundant architecture

The basic architecture suffers from several weak points due to the lack of redundancy and scalability. The architecture described in Figure 8 applies redundancy in several places to improve the dependability and scalability of the basic architecture. It is based on a server farm configuration with load balancing, including N_W web servers, two application servers and two database servers with two mirrored disks. The servers are connected through a LAN. Indeed, several LANs are generally used to interconnect these servers, nevertheless we will assume that all of them are represented as a single LAN. Also, to simplify the modeling, the load balancers are not explicitly described in this architecture.

In the next section, we will model the availability of both the basic and redundant architectures.

4. TA availability modeling

The availability modeling of the TA will be carried out according to the hierarchical description of the system in four steps (see Figure 1), starting at the service level considering the two architectures of Figures 7 and 8.

4.1. Service level availability

At this step, we are concerned with the evaluation of external and internal service availabilities.

4.1.1. External services. Each external system is modeled as a black box that is assumed to fail independently of all the others.

Let us consider the following notations:

- $\mathcal{A}_{Fi}$, $\mathcal{A}_{Hj}$ and $\mathcal{A}_{Ck}$: Availabilities of a flight, hotel and car reservation system, ($i = 1, \ldots, N_F; j = 1, \ldots N_H; k = 1, \ldots, N_C$).
- $\mathcal{A}_{PS}$: Availability of the payment system.
- $\mathcal{A}_{net}$: Availability of the TA connectivity to the Internet.

Using the failure independence assumption and considering that the service is provided as long as at least one reservation system for each item of a trip (flight, hotel and car reservation) is available, the availability of the external services can be derived as in Table 3. It is worth mentioning that if the TA connectivity to the Internet is unavailable, none of these services is provided. Thus, the availability of the TA connectivity to the Internet will be accounted for by multiplying the user perceived availability expression by $\mathcal{A}_{net}$ (cf. Section 4.3).

$\mathcal{A}(Flight) = 1 - \prod_{i=1}^{N_F}(1-\mathcal{A}_{Fi})$	$\mathcal{A}(Hotel) = 1 - \prod_{i=1}^{N_H}(1-\mathcal{A}_{Hi})$
$\mathcal{A}(Car) = 1 - \prod_{i=1}^{N_C}(1-\mathcal{A}_{Ci})$	$\mathcal{A}(Payment\ service) = \mathcal{A}_{PS}$

Table 3. External service availability

4.1.2. Internal services. These concern the web, application and database services.

For both architectures of Figures 7 and 8, the communication between servers is achieved by a local area network (LAN). The LAN is assumed to be a single point of failure, i.e., when the LAN is unavailable, all internal services are unavailable. As a consequence, the LAN availability, denoted by $\mathcal{A}_{LAN}$, is a multiplying factor in all equations giving the various function availabilities (as will be seen in Section 4.2). $\mathcal{A}_{LAN}$ can be evaluated using for example the models discussed in [16, 17].

As the primary objective of this paper is to show the applicability of the proposed approach to the TA example, we make simplistic assumptions for the application and database services. More realistic assumptions are made for the web service, to illustrate the kind of more complex calculations that can be performed.

Application and database service availability: Let us denote by $\mathcal{A}(C_{AS})$ and $\mathcal{A}(C_{DS})$ the availabilities of the computer hosts associated to the application and database servers, respectively. The disk availability is denoted by $\mathcal{A}(Disk)$. To simplify the presentation we assume that the computer hosts and the disks fail independently of each other. The application and database service availability (denoted as, $\mathcal{A}(AS)$ and $\mathcal{A}(DS)$) are given in Table 4.

	Basic architecture	Redundant architecture
$\mathcal{A}(AS)$	$\mathcal{A}(C_{AS})$	$1-(1-\mathcal{A}(C_{AS}))^2$
$\mathcal{A}(DS)$	$\mathcal{A}(C_{DS})\mathcal{A}(Disk)$	$\left[1-(1-\mathcal{A}(C_{DS}))^2\right]\left[1-(1-\mathcal{A}(Disk))^2\right]$

Table 4. Application and database service availability

In the following, we focus on the evaluation of the web service availability considering the basic and redundant architectures, respectively.

Web service availability: We take into account:1) hardware and software failures that affect the computer host and lead to web server failure, and 2) performance-related failures that occur when the incoming requests are not serviced due to the limited capacity of the web servers

The web service is assumed to be available when neither of the above types of failures occurs.

The impact of both types of failures on the web service availability can be accounted for by adopting a composite performance and availability evaluation approach [18, 19]. The main idea consists in combining the results obtained from two models: a pure performance model and a pure availability model. The performance model takes into account the request arrival and service processes and evaluates performance related measures conditioned on the state of the system as determined from the availability model. The availability model is used to evaluate the steady state probability associated to the system states that result from the occurrence of failures and recoveries.

This approach is based on the assumption that the system reaches a quasi steady state with respect to the performance-related events, between successive occurrences of failure-recovery events. This assumption is valid when the failure/recovery rates are much lower than the request arrival/service rates, which is typically true in our context.

Basic architecture: It is composed of a unique computer host, CWS. Let us denote by p_K the probability that the web server input buffer (whose size is K) is full when a request is received. The evaluation of p_K is derived from the performance model and depends on the assumptions made about the request arrival process and the request service process. Let us assume that the request arrivals are modeled by a Poisson process with rate α and the request service times are exponentially distributed with rate ν. Then the web server behavior governed by the arrival and service processes can be modeled by an *M/M/1/K* queue.

The probability that an arriving request is lost due to buffer being full is given by (see e.g., [20]):

$$p_K = \rho^K \frac{1-\rho}{1-\rho^{K+1}} \text{ with } \rho = \alpha / \nu \tag{1}$$

The availability model is composed of two states: up and down states. The steady state probability of the up state corresponds to the system steady-state availability denoted $\mathcal{A}(C_{WS})$.

Thus, the availability of the web service is:

$$\mathcal{A}(Web\ service) = (1-p_K)\mathcal{A}(C_{WS}) \tag{2}$$

This definition of availability allows incorporation of the inherent dependence between performance and dependability in one equation.

Redundant Architecture: The redundant architecture is composed of N_W identical web servers. We assume that all component failures are independent and that the web service is provided as long as at least one of the redundant component systems is available.

The performance model associated to this architecture to evaluate $p_K(i)$, the probability that web requests are lost due to input buffer being full, is assumed to be described by an *M/M/i/K* queue, where i is the number of servers available and K is the size of the buffer.

For a system state with i operational servers, $p_K(i)$, is given by (see, e.g., [20]):

$$p_K(i) = \frac{\rho^K}{i^{K-i} i!}\left[\sum_{j=0}^{i-1} \frac{\rho^j}{j!} + \sum_{j=i}^{K} \frac{\rho^j}{i^{j-i} i!}\right]^{-1} \quad i \geq 2 \tag{3}$$

Note that $p_K(1)$ is given by equation 1.

With respect to the availability model, the aim is to model the redundant architecture behavior resulting from the occurrence of failures/repairs, in order to evaluate the steady state probability associated to system states i (i is the number of operational servers, as denoted above).

Two assumptions are made with regards to the coverage of web server failures: 1) perfect coverage, and 2) imperfect coverage.

Perfect coverage: In the Figure 9 model, it is assumed that each web server runs on a dedicated computer host. Web server failures occur with rate λ. The model assumes shared repair facilities with repair rate μ. When a server fails, it is automatically disconnected and the system is reconfigured (with probability 1) with the web servers that are still operational.

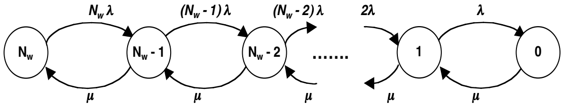

Figure 9. Markov model (perfect coverage)

Let us denote by Π_i the steady-state probability of state i, $i = 0, 1, \ldots, Nw$. In state i, $i \neq 0$, i web servers are available to process the input requests. (Π_0 corresponds to the state where all web server are down).

$$\Pi_i = \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i \Pi_o \quad \text{and} \quad \Pi_0 = \left[\sum_{i=0}^{Nw} \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i\right]^{-1} \qquad (4)$$

The availability of the web service is as follows:

$$\mathcal{A}(Web\ service) = 1 - \left(\sum_{i=1}^{Nw} \Pi_i p_K(i) + \Pi_o\right) \qquad (5)$$

where $p_K(i)$ is given by equation (3).

The expression between the brackets corresponds to the probability that a web request is not serviced either due a) to buffer being full or b) to web server unavailability.

Imperfect coverage: This assumption is included in the model presented in Figure 10, where from each state i, two transitions are considered:

1) After a covered failure (transition with rate $ic\lambda$) the system is automatically reconfigured into an operational state with (i-1) web servers.
2) Upon the occurrence of an uncovered failure (transition with rate $i(1\text{-}c)\lambda$), the system moves to a down state y_i, where a manual reconfiguration is required before moving to operational state (i-1). The reconfiguration times are exponentially distributed with mean 1/β.

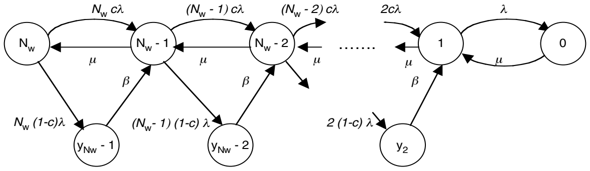

Figure 10. Markov model (imperfect coverage)

Solving Figure 10 model for steady-state probabilities leads to:

$$\Pi_i = \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i \Pi_o \qquad i=1,\ldots, Nw \qquad (6)$$

$$\Pi_{y_i} = \frac{\mu(1-c)}{\beta(i\text{-}1)!}\left(\frac{\mu}{\lambda}\right)^{i\text{-}1} \Pi_o \quad i=1,\ldots, Nw\text{-}2 \qquad (7)$$

$$\Pi_0 = \left[\sum_{i=0}^{Nw} \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i + \sum_{i=0}^{Nw\text{-}2} \frac{\mu(1-c)}{\beta(Nw\text{-}i\text{-}1)!}\left(\frac{\mu}{\lambda}\right)^{Nw\text{-}i\text{-}1}\right]^{-1} \qquad (8)$$

As states y_i correspond to down states, the availability of the web service is computed as follows:

$$\mathcal{A}(Web\ service) = 1 - \left[\sum_{i=1}^{Nw} \Pi_i p_K(i) + \sum_{i=1}^{Nw-2} \Pi_{y_i} + \Pi_o\right] \qquad (9)$$

where $p_K(i)$, is also given by equation (3).

Summary: Table 5 recalls the equations of the web server availability for the basic and redundant architecture, assuming perfect and imperfect coverage.

Basic architecture $\mathcal{A}(Web\ service) = \mathcal{A}(C_{WS})(1 - p_K)$ $p_K = \rho^K \frac{1-\rho}{1-\rho^{K+1}} \qquad \rho = \alpha/\nu$
Redundant architecture (perfect coverage) $\mathcal{A}(Web\ service) = 1 - \left(\sum_{i=1}^{Nw} \Pi_i p_K(i) + \Pi_o\right)$ $p_K(i) = \frac{\rho^K}{i^{K\text{-}i} i!}\left[\sum_{j=0}^{i\text{-}1} \frac{\rho^j}{j!} + \sum_{j=i}^{K} \frac{\rho^j}{i^{j\text{-}i} i!}\right]^{-1} \qquad \rho = \alpha/\nu$ $\Pi_0 = \left[\sum_{i=0}^{Nw} \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i\right]^{-1} \qquad \Pi_i = \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i \Pi_o$
Redundant Architecture (imperfect coverage) $\mathcal{A}(Web\ service) = 1 - \left[\sum_{i=1}^{Nw} \Pi_i p_K(i) + \sum_{i=1}^{Nw-2} \Pi_{y_i} + \Pi_o\right]$ $p_K(i) = \frac{\rho^K}{i^{K\text{-}i} i!}\left[\sum_{j=0}^{i\text{-}1} \frac{\rho^j}{j!} + \sum_{j=i}^{K} \frac{\rho^j}{i^{j\text{-}i} i!}\right]^{-1} \qquad \rho = \alpha/\nu$ $\Pi_0 = \left[\sum_{i=0}^{Nw} \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i + \sum_{i=0}^{Nw\text{-}2} \frac{\mu(1-c)}{\beta(Nw\text{-}i\text{-}1)!}\left(\frac{\mu}{\lambda}\right)^{Nw\text{-}i\text{-}1}\right]^{-1}$ $\Pi_i = \frac{1}{i!}\left(\frac{\mu}{\lambda}\right)^i \Pi_o \qquad \Pi_{y_i} = \frac{\mu(1-c)}{\beta(i\text{-}1)!}\left(\frac{\mu}{\lambda}\right)^{i\text{-}1} \Pi_o$

Table 5. Web service availability

4.2. Function level availability

The availability evaluation of each function is based on the availabilities of the services involved in its accomplishment and — when various function execution scenarios are possible — on the activation probability of each scenario. Table 6 gives the availability for the *Home*, *Browse*, *Search*, *Book* and *Pay* functions.

$\mathcal{A}(WS)$, $\mathcal{A}(AS)$, $\mathcal{A}(DS)$ correspond to $\mathcal{A}$*(Web service)*, $\mathcal{A}$ *(Application service)* and $\mathcal{A}$ *(Database service)* given in Tables 4 and 5. $\mathcal{A}$ *(PS)* corresponds to $\mathcal{A}$*(Payment service)* given in Table 4. $\mathcal{A}$ (Flight), $\mathcal{A}$ *(Hotel)* and $\mathcal{A}$ *(Car)* are given in Table 4. The parameters q_{ij} involved in the availability of the *Browse* function are associated to the three execution scenarios of this function given in Section 3.2 (Fig. 3).

Note that all function equations include the product $A_{net}A_{LAN}$, meaning that if the TA connectivity to the Internet or the internal communication among the servers is not available, none of the TA functions can be invoked by the users. Also, the *Book* function has the same availability equation as the Search function. This is due to the assumption that the former uses a subset of the resources used by the latter. Indeed, in our example the Book function is achieved only if the Search function has succeeded. This led us to assume that if the Search function succeeds, automatically the *Book* function succeeds. Of course, other situations can be modeled.

$\mathcal{A}\,(Home) = \mathcal{A}_{net}\,\mathcal{A}_{LAN}\ \mathcal{A}(WS)$
$\mathcal{A}\,(Browse) = \mathcal{A}_{net}\,\mathcal{A}_{LAN}\,\mathcal{A}(WS)\ [q_{23}\ +\mathcal{A}(AS)(q_{24}.q_{45} + q_{24}.q_{47}.\mathcal{A}(DS))]$
$\mathcal{A}\,(Search) = \mathcal{A}_{net}\mathcal{A}_{LAN}\ \mathcal{A}(WS)\ \mathcal{A}(AS)\ \mathcal{A}(DS)\ \mathcal{A}(Flight)\ \mathcal{A}(Hotel)\ \mathcal{A}(Car)$
$\mathcal{A}\,(Book) = \mathcal{A}_{net}\mathcal{A}_{LAN}\ \mathcal{A}(WS)\ \mathcal{A}(AS)\ \mathcal{A}(DS)\ \mathcal{A}(Flight)\ \mathcal{A}(Hotel)\ \mathcal{A}(Car)$
$\mathcal{A}\,(Pay) = \mathcal{A}_{net}\ \mathcal{A}_{LAN}\ \mathcal{A}(WS)\ \mathcal{A}(AS)\ \mathcal{A}(DS)\ \mathcal{A}(PS)$

Table 6. Function level availabilities

4.3. User level availability

For a given user operational profile, the user perceived availability can be obtained by evaluating for each user execution scenario derived from the operational profile, the expression specifying that all functions invoked in the corresponding scenario are available. When several functions are invoked in a given scenario, a careful analysis of the dependencies that might exist among the functions due to shared services or resources is needed at this stage to evaluate the availability measure associated to the scenario from the availability of the corresponding functions.

Based on the activation probabilities of all user scenarios i, π_i, (presented in Table 1) and the availability of the functions involved in each scenario, the user availability is given by equation (10).

$$\begin{aligned}\mathcal{A}(user) = {} & \mathcal{A}_{net}\,\mathcal{A}_{LAN}\,\mathcal{A}(WS)[\pi_1 + \\ & (\pi_2 + \pi_3)\ \{\ q_{23} + \mathcal{A}(AS)\ (q_{24}\,q_{45} + q_{24}\,q_{47}\ \mathcal{A}(DS)\} \\ & +\mathcal{A}(AS)\mathcal{A}(DS)\mathcal{A}(Flight)\mathcal{A}(Hotel)\mathcal{A}(Car) \\ & \{(\pi_4 + \pi_5 + \pi_6 + \pi_7 + \pi_8 + \pi_9) + (\pi_{10} + \pi_{11} + \pi_{12})\mathcal{A}(PS)\}] \qquad (10)\end{aligned}$$

It can be seen that the availabilities of the LAN, the net and the web service are the most influential ones (i.e., their impact is of the first order, while the others are at least at the second order). This is due to the fact that all requests (i.e., all user scenarios) use these three services.

5. Evaluation results

We will first show the impact of the number of web servers as well as their failure rates on the web service availability, according to the request arrival rates. Then, based on the various equations derived in the previous section, we will evaluate the user availability as perceived by user classes A and B.

5.1. Web service availability results

Figures 11 and 12 give the web service availability for perfect and imperfect fault coverage, with the number of web servers NW varying from 1 to 10. When only one web server is used (NW = 1), the results correspond to the basic architecture. The parameters used to obtain these curves are indicated on the figures. Sensitivity analyses are done considering different values of web server failure rates (10^{-2}, 10^{-3} and 10^{-4} per hour) and request arrival rates (50, 100 and 150 requests per second). It is assumed that each web server has a processing rate ν equal to 100 per second and a repair rate μ equal to 1 per hour. The mean reconfiguration rate of the web server architecture (β) is 12 per hour (i.e., 1/β = 5 min) and the buffer size K is assumed to be 10.

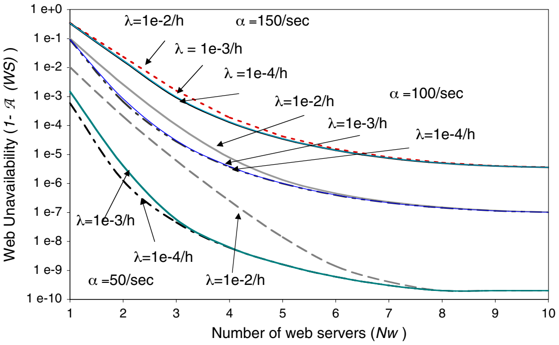

Figure 11. Web service unavailability (perfect coverage)

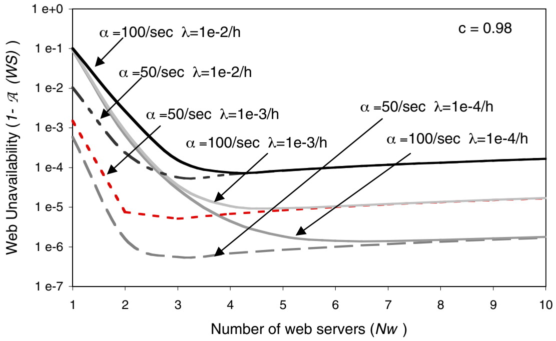

Figure 12. Web service unavailability (imperfect coverage)

Both figures show that increasing the number of web servers NW from 1 to 2, 3 or 4 (depending on the failure and request arrival rates) reduces the web service unavailability. However, the trend is reversed when the coverage is imperfect for NW values higher than 4 (Figure 12). This is due to the fact that when the coverage is imperfect, increasing the number of servers also increases the probability for the system being in states y_i (of Figure 10) where the web service is unavailable and a manual reconfiguration action is required. Actually, the probability of a request being rejected because the buffer is full plays a significant role until a certain value of NW. When the number of servers is higher than the threshold value, the total service rate and the buffer capacity are sufficient to handle the flow of arrivals without rejecting requests. In this case, the unavailability of the web service mainly results from hardware and software failures leading the web server architecture to a down state. Compared to the imperfect coverage model, it can be noticed that the model with perfect coverage is more sensitive to the variation of NW. Indeed the unavailability decreases exponentially when NW increases and the trend is not reversed for values higher than 4. Also, the web servers failure rate has a significant impact on availability only when the system load (α/ν) is lower than 1.

Design decisions can be made based on the results presented on these figures. In particular, we can determine the number of servers needed to achieve a given availability requirement, or evaluate the maximum availability that can be obtained when the number of servers is set to a given value. For instance, considering the model with imperfect coverage, the number of servers needed to satisfy an unavailability lower than 5 min/year (unavailability < 10^{-5}), with a failure rate equal to 10^{-3} per hour will be at least *NW*=2 if the request arrival rate is 50 per second and *NW*=4 if the request arrival rate is 100 per second. We obtain the same result with a failure rate 10^{-4}per hour, however such a requirement cannot be satisfied with a failure rate of 10^{-2}per hour.

Similar sensitivity analyses can be done to study the level of availability that can be achieved when the number of web servers is set to a given value. For example, if we decide to employ three servers to support the web service, we would have an unavailability lower than 1 hour per year, when the failure rate varies from 10^{-2} to 10^{-4} and the system load (α/ν) is less than 1.

5.2. User level availability results

Considering equation (10), we will evaluate the availability as perceived by user classes A and B. The values of the parameters involved in this equation are given in Table 7. The probabilities characterizing user execution scenarios for classes A and B profiles have been presented in Table 1. It is assumed that the web service is implemented on four servers, with imperfect coverage (NW=4, c=0.98, α=100/sec, $\lambda=10^{-4}$/hour).

$$\mathcal{A}_{net} = \mathcal{A}_{LAN} = 0.9966 \quad \mathcal{A}(C_{AS}) = \mathcal{A}(C_{DS}) = 0.996 \quad \mathcal{A}(Disk) = 0.9$$

$$\mathcal{A}_{PS} = \mathcal{A}_{Fi} = \mathcal{A}_{Hi} = \mathcal{A}_{Ci} = 0.9 \quad \mathcal{A}(WS) = 0.999995587$$

$$q_{23} = 0.2 \quad q_{24} = 0.8 \quad q_{45} = 0.4 \quad q_{47} = 0.6$$

Table 7. Model parameters

Table 8 presents the user perceived availability for user classes A and B, considering different values for the number of flight, car and hotel reservation systems (N_F, N_H, N_C) interacting with the travel agency system. The same number is assumed for N_F, N_H and N_C.

The results show that for a given user class, the user perceived availability increases significantly when the number of reservation systems increases from 1 to 4, and then stabilizes. The availability variation rate is directly related to the availability assigned to each reservation system. Comparison of the results obtained for class A and B users show that different operational profiles might lead to significant differences in the availability perceived by the users. For instance, considering the case $N_F = N_H = N_C \geq 5$, the user perceived unavailability is about 173 hours per year for class A users and 190 hours for class B users. Such unavailability takes into account all the scenarios that might be invoked by the users.

$N_F = N_H = N_C$	$\mathcal{A}$(A users)	$\mathcal{A}$(B users)
1	0.84235	0.76875
2	0.96509	0.95529
3	0.97867	0.97593
4	0.98004	0.97802
5	0.98018	0.97822
10	0.98020	0.97825

Table 8. Class A and B user availabilities *wrt* N_F, N_H, N_C

The user perceived availability can be analyzed from another perspective by grouping user scenarios into four categories, denoted as SC1, SC2, SC3 and SC4, and evaluating the contribution of each category to the perceived availability:

- SC1 gathers all scenarios that lead to the execution of functions "Home" or "Browse" without invoking the other functions (scenarios 1-3 of Table 1).
- SC2 gathers all scenarios that include the invocation of the "Search" function, without going through the "Book" or "Pay" functions (scenarios 4-6 of Table 1).
- SC3 gathers all scenarios that include the "Book" function (scenarios 7-9 of Table 1).
- SC4 gathers all scenarios that reach the "Pay" function (i.e., scenarios 10-12 of Table 1).

This is illustrated on figure 13 considering class A and class B users, respectively, and assuming that the web service is implemented on four servers with imperfect coverage.

UA (A users) (respectively *UA (B users)*) denotes the unavailability perceived by Class A users, and *UA(SCi)* denotes the contribution of scenarios *SCi* to the user perceived unavailability.

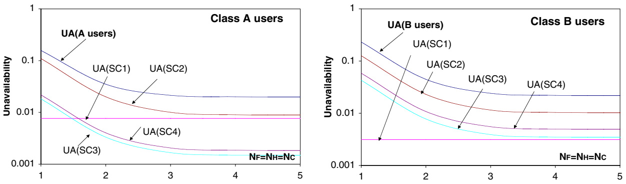

Figure 13. User perceived unavailability and UA(SCi)

It can be seen that the unavailability caused by scenarios SC4 that end up with a trip payment is higher for class B users compared to class A users (43 hours downtime per year for class B users compared to 16 hours for class A users. when considering the steady values). Therefore, the impact in terms of loss of revenue for the TA provider will be higher. Indeed. if the users transaction rate is 100 per second, the total number of transactions ending up with a payment that are lost is 5.7 million for class A users and 15.5 million for class B users. Assuming that the average revenue generated by each transaction is 100$. Then the loss of revenue amounts to 570 million dollar and 1.55 billion dollar, respectively. This result clearly shows that it is important to have a faithful estimation of the user operational profile to obtain realistic predictions of the impact of failures from the economic and business viewpoints.

6. Conclusion

In this paper, we have illustrated the main concepts that we defined within our hierarchical modeling framework proposed in [12] for the dependability evaluation of internet based applications on a travel agency example. Our objectives were: 1) to show how to apply our framework considering the decomposition of the target system according to four levels: user, function, service, and resource levels, and 2) to present typical dependability analysis and evaluation results that could be obtained from the modeling to help the e-business providers in making objective design decisions.

For the sake of illustration, we have deliberately considered simplified (yet realistic) assumptions, concerning the users operational profile and the TA architecture models, and analyzed their impact on the user perceived availability. The availability measure considered takes into account the impact of performance related failures as well as traditional software and hardware failures. The sensitivity analyses presented in this paper clearly show the appropriateness of this measure. We have showed that the proposed hierarchical framework provides a systematic and pragmatic modeling approach, that is necessary to be able to evaluate the dependability characteristics of the target application at different levels of abstractions.

Future work will be focused on the extension of the framework to handle more complex assumptions and models. For example, besides taking into account performance failures related to the loss of user transactions due to servers input buffers being full, we can also extend the measure to include failures that occur when the response time exceeds an acceptable threshold.

References

[1] Bakos Y., "The Emerging Role of Electronic Marketplaces on the Internet", *Communications of the ACM*, 41 (8), pp.35-42, 1998.

[2] Menascé D. A. and Almeida V. A. F., *Capacity Planning for Web Services: Metrics, Models, and Methods,* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2002.

[3] Shim S. S. Y., Pendyala V. S., Sundaram M. and Gao J. Z., "Business-to-Business E-Commerce Frameworks", *Computer* (October), pp.40-47, 2000.

[4] Purba S., *Architectures for E-Business Systems: Building the Foundation for Tomorrow's Success,* Best Practices Series, AUERBACH Publications - CRC Press LLC, Boca Raton, FL, USA, 2002.

[5] Goodyear M., *Enterprise System Architectures: Building Client/Server and Web-based Systems,* AUERBACH Publications - CRC Press LLC, Boca Raton, FL, USA, 2000.

[6] Long D., Muir A. and Golding R., "A Longitudinal Survey of Internet Host Reliability", in Proc. *14th Symposium on Reliable Distributed Systems (SRDS-95),* pp.2-9, Bad Neuenahr, Germany, September 1995.

[7] Kalyanakrishnam M., Iyer R. K. and Patel J. U., "Reliability of Internet Hosts: a Case Study from the End User's Perspective", *Computer Networks*, 31, pp.47-57, 1999.

[8] Machiraju V., Dekhil M., Griss M. and Wurster K., *E-services Management Requirements*, HP Laboratories Palo Alto, CA, USA, N°HPL-2000-60, May 2000.

[9] Paxson V., Mahdavi J., Adams A. and Mathis M., "An Architecture for Large-Scale Internet Measurement", *IEEE Communications Magazine* (August), pp.48-54, 1998.

[10] Xie W., Sun H., Cao Y. and Trivedi K. S., "Modeling of Online Service Availability Perceived by Web Users", in *IEEE Global Telecommunications Conference (GLOBECOM 2002),* IEEE Computer Society, Taipei, Taiwan, November 2002.

[11] Kaâniche K., Kanoun K. and Rabah M., *A Preliminary Framework for SoS Dependability Modelling and Evaluation*, DSoS Project, IST-1999-11585, LAAS Report N° 01157, April 2001.

[12] Kaâniche K., Kanoun K. and Rabah M., "A Framework for modeling the Availability of e-Business Systems", in *10th International Conference on Computer Communications and Networks,* pp.40-45, IEEE CS, Scottsdale, AZ, USA, 15-17 October 2001.

[13] Menascé D. A. and Almeida V. A. F., *Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning,* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.

[14] van Moorsel A., "Metrics for the Internet Age: Quality of Experience and Quality of Business", in *Fifth International Workshop on Performability Modeling of Computer and Communication Systems,* pp.26-31, Universität Erlangen-Nürnberg, Institut für Informatik, Germany, September 2001.

[15] Menascé D. A., Almeida V. A. F., Fonseca R. C. and Mendes M. A., "Business-oriented Resource Management Policies for E-commerce Servers", *Performance Evaluation*, 42 (2-3), pp.223-239, 2000.

[16] Hariri S. and Mutlu H. B., "A Hierarchical Modeling of Availability in Distributed Systems", in *11th International Conference on Distributed Computing Systems,* pp.190-197, IEEE Computer Society, Arlington, TX, USA, 1991.

[17] Kanoun K. and Powell D., "Dependability evaluation of bus and ring communication topologies for the Delta-4 distributed fault-tolerant architecture", in *10th IEEE Symposium on Reliable Distributed Systems (SRDS-10),* pp.130-141, IEEE Computer Society, Pisa, Italy, 1991.

[18] Meyer J. F., "On Evaluating the Performability of Degradable Computer Systems", *IEEE Transactions on Computers*, C-29 (8), pp.720-731, 1980.

[19] Meyer J. F., "Closed-form Solutions of Performability", *IEEE Transactions on Computers*, C-31 (7), pp.648-657, 1982.

[20] Allen A. O., *Probability, Statistics, and Queuing Theory — With Computer Science Applications,* Computer Science and Applied Mathematics, Academic Press, 1978.