A Hierarchical Availability Analysis Methodology for Web Applications

JIJUN LU and SWAPNA S. GOKHALE*
Department of Computer Science and Engineering
University of Connecticut, Storrs, CT 06269, USA

(Received on May 8, 2007)

Abstract – We propose a hierarchical availability analysis methodology for multi-tiered Web applications. The methodology partitions the analysis into three levels, namely, server, request and session, and considers only the relevant factors at each level. The levels are connected using a hierarchical approach; the results obtained from one level are propagated for use in the analysis at the next one. The methodology thus decouples the different factors that influence availability and yet provides an integrated framework to consider them simultaneously.

Keywords: multi-tiered Web applications, availability analysis

1. Introduction
Multi-tiered Web applications have become a dominant source of information and services. The widespread use of these applications in many domains where service unavailability can lead to severe financial and other losses places a high premium on their availability.

The availability of a multi-tiered application, as perceived by a user, is influenced by various factors and characteristics. Examples of these factors include failure modes and repair parameters of a server, the tiers visited by a request when it is served by the application, and the sequence of requests presented by the user to accomplish a specific task. Incorporating all these factors into a single model would make the model extremely complex, lead to state space explosion, and make it difficult to determine the impact of each factor on the application availability. To avoid these pitfalls, we propose a hierarchical, three-level methodology for the availability analysis of these applications. By partitioning the analysis into three levels, we can consider factors that are most relevant at each level. The methodology thus provides a way to decouple the factors that need not be considered together. The analysis at each step thus remains tractable and provides a clearer picture of the influence exerted by each factor. Finally, the three levels are coupled by propagating the results obtained from each level for use as a parameter in the next level of analysis. The rest of the paper is devoted to the discussion of the methodology and comparing it with prevalent work.

2. Availability Analysis Methodology
The three levels in the methodology and the rationale behind them are as follows. The server-level considers the availability of a single server, since it is a basic unit which lies at the heart of a multi-tiered Web application. The request-level considers the availability of a request,
which is a fundamental unit of interaction between the user and the application. The session-level considers the availability of a session, since a user’s perception of the application’s availability will depend on whether it can fulfill the set of related requests presented during a session to accomplish a specific task. A brief description of each level is as follows:

**Server-level analysis:** The availability of a single server, considering different failure modes [1], namely, transient and permanent hardware, operating system and software server failures is analyzed at this level. Based on how these failures are resolved, we classify them into two categories, internal and external. Internal failures (transient hardware, operating system, and software server) are resolved by the support and maintenance staff employed internally. Permanent hardware failures are resolved externally by the manufacturer according to a maintenance contract. The terms of such a contract may include a fixed maintenance cost per server in exchange for a guaranteed mean time to repair. A Markov model, which represents the different failure and repair types is constructed and is solved to obtain server availability.

**Request-level analysis:** The availability of a request as it traverses through the different tiers is analyzed at this level. To fulfill a request, all the tiers through which it passes (a request may or may not pass through all tiers) need to be available. To provide superior performance through load balancing and to improve availability through redundancy, multiple servers may be deployed in each tier. Ideally, if at least one of the servers is available, then the tier may be considered available. However, to ensure acceptable performance, the number of operational servers must exceed a certain threshold. Thus, the availability of each tier is determined using the expression for a \( k\)-of-\( m \) good system [2]. The request availability is then determined as the product of the availabilities of those tiers through which the request traverses.

**Session-level analysis:** The availability of a session, which consists of a sequence of related requests, is analyzed at this level. The sequence of requests is determined by the navigational pattern of the group to which the user belongs. Further, an application may have multiple user groups, each with a distinct pattern. Thus, the analysis is conducted in two steps, first for a group and then across groups. For each group, we assume that the navigational pattern is represented by a Customer Behavior Model Graph (CBMG) [3], which is mapped to a discrete time Markov chain (DTMC) [2], to provide the mean and the variance of the number of requests of each type, which are then combined with the request availabilities obtained from the request level to provide session availability. The session availabilities for different groups are composed based on the distribution of groups to obtain the overall session availability.

Thus, in summary, the first level of analysis involves a single server and a single request, the second level involves multiple servers and a single request, and the third level involves multiple servers and multiple requests. We explain these levels using an online bookstore [2]. In this case, a single request can perform a task in the process of buying a book. For example, a SEARCH request may return a list of books which match a keyword, a SELECT request will put the selected book(s) into the shopping cart, and an INSTR request can offer instructions on how to buy/sell books. On the other hand, a session represents a sequence of requests issued to obtain a specific service. A session starts with a LOGIN request, and ends with an EXIT request after possibly multiple INSTR, BROWSE, SEARCH, SELECT, and PAY requests.
Single-server availability from the first level will be used at the request level to compute the availabilities of BROWSE, SELECT, SEARCH, SELECT, PAY, INSTR, LOGIN, and EXIT requests. It is very likely that the INSTR request does not need support from the database tier, while requests such as SEARCH and BROWSE do. These traversal patterns of requests will be considered while computing their availabilities. The availabilities of these requests will then be used in session-level analysis to compute the availability of a session.

3. Related Research
The research efforts which consider availability of Web applications and their limitations with respect to our approach are presented in this section. Hecht [4] uses Markov chains and reliability block diagrams to analyze Web site availability. Merzbacher et al. [5] measure the availability of selected sites and propose a new metric for availability characterization. Martinello et al. [6] analyze service availability of Web cluster architectures, considering error recovery strategies and traffic models.

The methodology described in this paper is closest to function and service-level availability considered by Kaaniche et al [7]. However, their approach relies on enumerating paths which provides only an approximate availability estimate when the graph has loops. On the other hand, our approach can consider the impact of loops analytically. Second, their approach is purely computational, which makes sensitivity and predictive analysis cumbersome. We produce analytical availability expressions at all levels, and hence, sensitivity and predictive analysis and bottleneck identification is easily facilitated.

4. Conclusions and Future Research
In this paper we presented a hierarchical availability analysis methodology for multi-tiered Web applications. Our future research consists of illustrating the potential of the different types of analyses facilitated by the methodology using examples.

References