Quality of service management issues in electronic commerce applications

Gregor v. Bochmann (University of Ottawa)
Brigitte Kerhervé (University of Québec at Montreal)
Mohamed-Vall O. Mohamed-Salem (University of Montreal)

1. Introduction

Quality of service issues have first been discussed in relation with the performance of communication networks. However, for understanding end-to-end performance of a distributed system, it is important to also consider the performance at the application level, of the different components of the distributed application, as well as the quality of service view of the user. Most work on quality of service management at the application level has been done for applications involving access to distributed multimedia databases, such as video-on-demand. The same QoS issues apply also to electronic commerce when users access a catalog which may include multimedia information in addition to the hard-core data elements, such as name and price of the product. In this paper, we identify various issues related to quality of service management for electronic commerce applications, and concentrate on one of these, namely the selection of a server from a pool of servers in order to optimize the overall cost-performance of the system and the user's satisfaction.

The paper is organized as follows. Section 2 reviews QoS specification, its categories and dimensions in distributed systems. In section 3 we look at the QoS issues (QoS specification and provision) in the specific context of electronic commerce. Section 4 describes a scalable architecture for QoS management for the electronic commerce applications.

2. QoS Specification

Quality of Service management on an end-to-end basis means that the user expresses his wishes such as the quality he wants or the cost he is willing to pay. QoS specification deals with the definition of the requested QoS level, provided by the user or by the application programmers. It can be offered through a specification language or an adequate interface allowing the user to express how he perceives the quality of the provided service. QoS specification is made according to different dimensions belonging to QoS categories such as *performance* or *cost*. A dimension can be defined as a qualitative or quantitative attribute of a category [Frolund, 1998]. An example of a dimension in the *performance* category is *response-time*. In most existing QoS approaches, categories and dimensions are performance-oriented. They were generally proposed in the framework of distributed multimedia systems to support audio and video transfer with synchronization constraints.

Emerging applications, such as electronic commerce, health-care applications, digital publishing or data mining, integrate large amounts of new types of data which are large-size, heterogeneous and/or time-dependent. High volumes of data are located on several sites interconnected through different communication networks and potentially accessed using mobile computing equipment. Information discovery becomes a hard task, often frustrating while getting poor and irrelevant data with unacceptable response time. That leads to a revision of the traditional assumptions of QoS specification, to a broader definition of QoS and to new issues for QoS provision. The traditional categories and dimensions of QoS, such as performance, time or cost, should then be revised and extended to integrate the concepts of data quality, quality of data source as well as the quality of provided services such as search strategies, data transfer or data presentation.

Traditional categories and dimensions for QoS specification have been defined to characterize non-functional properties of services provided for the delivery of multimedia streams in distributed systems. They generally include the *performance*, *reliability* and *security* categories. In different systems and prototypes we can find specific dimensions to define each category. The *performance* category may include the following dimensions: *response-time*, *delay*, *latency*, *throughput*, number of transactions per second (TPS). The reliability category may include the following dimensions: *time-to-failure, time-to-repair,* or number of failures. We refer the reader to [Koistinen, 1997] for an interesting literature survey on dimensions for reliability. The *security* category can include the following dimensions: *anonymity*, *encryption, and authentication.*

Category	Dimensions
Performance	response time delay latency throughput transactions_per_second
Reliability	time_to_failure time_to_repair number_of_failures
Security	Anonymity Encryption Authentication

Table 1: Traditional categories and dimensions for QoS specification

In the framework of the Total Data Quality Management (TDQM) research program, Wang and al. [Wang, 1996] propose a methodology to deliver high-quality information products to information consumers. They investigate data quality definitions, modeling and control in order to integrate quality in the different phases of the data life cycle. They have identified four categories and fifteen dimensions for data quality (DQ). *Intrinsic* DQ defines if data are in conformance with actual values and includes the following dimensions: *accuracy, accessibility, believability* and *reputation*. *Accessibility* DQ defines if data are available or obtainable and includes the access and security dimensions. *Contextual* DQ defines if data are applicable and pertinent to the task of the user and includes the *relevancy, value-added, timeliness, completeness* and *amount of data*. *Representational* DQ is related to the format and the meaning of data and includes the following dimensions: *interpretability, ease of understanding, concise representation* and *consistent representation*. The categories and dimensions defined in the context of this research project are very pertinent to extend the concept of Quality of Service in large scale distributed systems. These definitions can be seen as core categories and dimensions to specify the data quality part of QoS specification.

Category	Dimensions
Intrinsic	accuracy accessibility believability reputation
Accessibility	access security
Contextual	relevancy value-added timeliness completeness amount of data
Representational	interpretability ease of understanding concise representation consistent representation

Table 2: Categories and dimensions for data quality in TDQM

The quality of data sources refers to characteristics of service providers such as database servers, continuous-media file servers, brokers or catalog managers. Defining the quality of a data source can be derived from both the quality of the data provided by this data source and from the non-functional properties of the data source such as performance, reliability or security. As we can see while using the Internet, we generally rate web servers or search engines according to the believability, accessibility or completeness of the data they provide as well as according to the performance or security provided by the system.

The quality of provided services refer to non-functional properties of services provided by computers. Categories and dimensions related to provided services strongly depend on the service type. Nevertheless, we can attempt to identify them for essential functions such as search, transfer or presentation. Quality of a search service is determined by the quality of the results of the search and the quality of query formulation framework [Dreilinger, 1997]. We can then evaluate and quantify search strategies according to the following dimensions: *query formulation, language, ranking algorithms and duplicates removing.*

The quality of transfer can be evaluated according to *reliability*, *security* and *performance* categories previously presented. Quality of presentation depends on the type of data to be displayed. In distributed multimedia systems, interesting work has been conducted to support high quality presentation of multimedia documents respecting the spatial and temporal constraints defined for multimedia documents [Vogel 1995, Bochmann 1997]. These presentation constraints induce some *performance* constraints such as the *throughput* for the communication network. The quality of the presentation also obviously depends on the quality of the data and more specifically related to the *representational* DQ category and the *concise representation* dimension. We believe that it would be important to refine this dimension in order to precisely define dimensions such as *format*, *resolution*, or *frame-rate* for multimedia data.

Category	Dimensions
Search	query formulation language ranking algorithms duplicates removing
Presentation	spatial constraints temporal constraints format resolution frame-rate

Table 3: Categories and dimensions for quality of provided services.

This set of QoS categories and dimensions should be the basis for the design and the development of a user-oriented QoS specification tool [Hafid 1996]. This tool should allow users to set up QoS profiles and contracts defined along specific dimensions and categories. User QoS profiles and contract should integrate the relevant categories and dimensions, the domain definitions for dimensions, as well as the ordering relations on domains, dimensions and categories. QoS profiles and contracts strongly depend on the type of target applications and predefined QoS profiles should be defined by the application developers or system administrators. These predefined QoS profiles should then be specialized to the specific needs of the user.

3. Quality of Service in the Context of Electronic Commerce application

For Electronic Commerce, we identify the following pertinent categories and dimensions: (i) performance, reliability and security for both data sources and provided services; (ii) intrinsic DQ and more specifically the accuracy and reputation dimensions; (iii) representational DQ and more specifically concise representation, and consistent representation dimensions; (iv) accessibility DQ and more specifically the security dimension; and (v) multimedia presentation quality. In what follow, we describe the typical electronic shopping model and the related QoS issues.

3.1 Typical electronic commerce shopping Scenario

The diagram in Figure 3.1 shows the different steps and possible paths in a typical electronic commerce shopping session. Each individual request in a session corresponds to a TCP/IP connection establishment with a merchant server, followed by the client think time, another request and so on. When a client connects to a merchant server, he/she first receives the merchant front-end page. After that, the client can do a search or browse a particular product information. While browsing the client can use the shopcart to keep track of selected products. The application gives also the possibility to prepare an order and to execute a payment. The client session may go along for a long time extending over several TCP/IP connections. Servers keep track of the clients previous requests by retrieving the so-called web cookies stored on clients' machines.

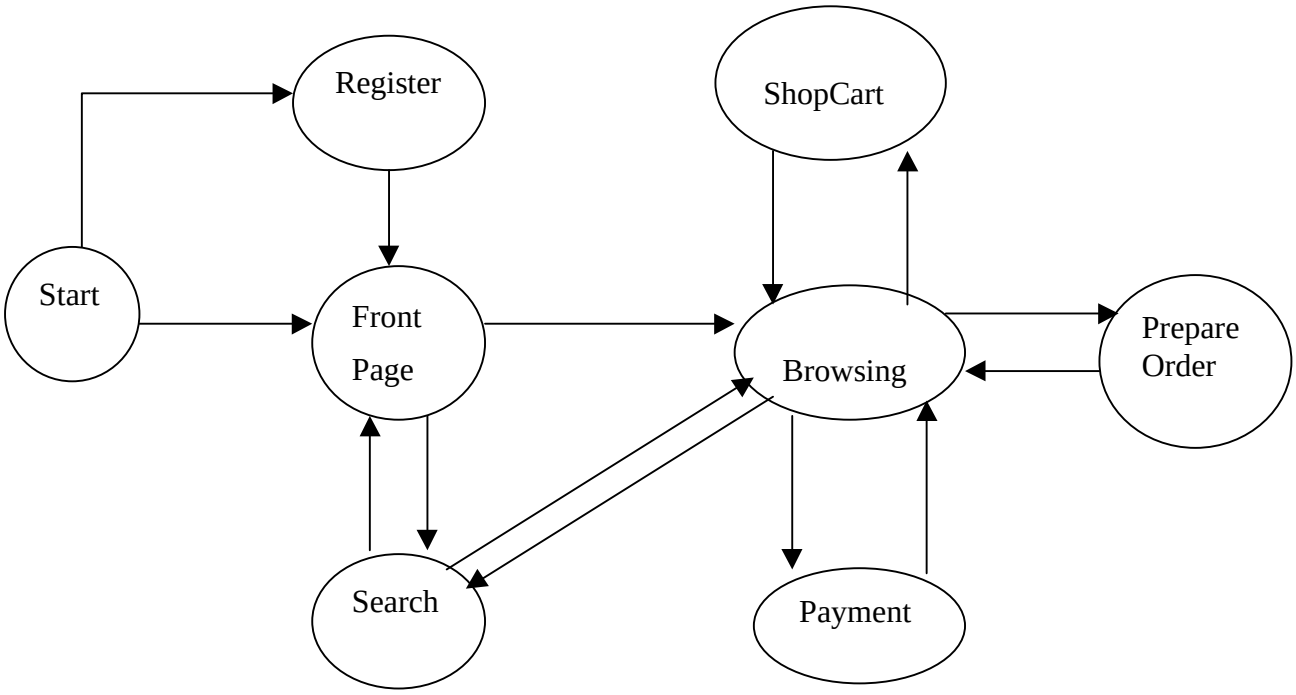

Figure 3.1 Client shopping session

3.2 Quality of Service Issues

Information retrieval through browsing and searching represents the main part of an electronic shopping session. The performance with which this operation is carried out has a direct impact on the user's satisfaction and the reputation of merchant sites. The response time is defined as the time the user spends waiting for a request to be completed. This response time depends on factors like the server capacity, the available network bandwidth between the client and the server, the size of the requested information, and the complexity of the request. While some requests just require the transmission of an HTML page, other requests, like for instance a payment transaction, necessitate a lot of processing (ciphering/deciphering) and external interactions (payment gateways and banks).

While the response time expresses the performance level with which the information is delivered to the user, the quality of the delivered information is also very important. When a user for example issues a search request for some information in a digital library system, the quality of the results will depend on the quality of the search (depth and filtering), the type and quality of the information (audio/video/text/etc.), and the quality of information sources.

Most of the existing electronic commerce applications enable merchants to keep some information related to their clients (session information, identification, preferences, etc.). An important aspect of the quality of service will then be the level of security offered to users. Users have to be able to specify the level of discretion they want for their personal information from unauthorized listeners and from undesired usage by merchants. Different users will have variable concern about the security issue. In general, users need to have a clear understanding of the available security categories to choose from. This may include the level of security offered for personal information, the legal guarantees on possible usage of their personal information, and the degree of confidence in the authentication process.

The nature of electronic commerce application as described above is very different from distributed multimedia applications. A shopping session in fact can be characterized by short and frequent requests, variable think times, and an undefined number of requests and hence an undefined session length. The above considerations make the quality of service provision in the electronic commerce slightly different from traditional QoS provision mechanisms. From a practical point of view, it is not reasonable to have a QoS negotiation before every request. Another important issue is how and where to store the information on the user profile and server performance. The client QoS profile can either be stored during registration time at the server, or sent on request from the client side. We think that it is necessary to introduce QoS-aware intermediate nodes (QoS-brokers) between clients and servers to handle the QoS related issues. The QoS negotiation process should then be carried out implicitly while the client is locating a server. Before connecting to any server, the client has to go first through a QoS broker to specify his QoS requirements (for example, a maximum tolerable response time). Based on the required QoS and the implemented server selection policy, the broker chooses a server and sends back its address to the client, or directly forwards the request to the selected server.

4. Scaling electronic commerce applications

An Electronic Commerce (EC) application is a set of cooperating pieces of software that may or may not be on the same machine. A typical application will be composed of a WEB server that talks HTTP with client's browsers, a merchant database, and one or more EC servers. The role of the EC server is to implement the logic of the supported shopping model. An EC server then parses and checks client requests, generates and executes appropriate queries to the database, formats and forwards the results to the client through the WEB server. EC servers can be organized so that they can handle all types of requests (registration, browsing, payment, etc.), or specialized in a particular type of requests. Each application then could support as many concurrent requests as the number of running EC servers, but such level of saturation may not necessarily be acceptable if the system wants to operate in acceptable QoS limits. When the service becomes popular, it has to be replicated to enhance its availability, performance and reliability. In a single server architecture, clients know the server address and contact it directly. In replicated independent architectures, each replica has its own real address, and users choose which server to use often from a given list of servers depending on the geographical proximity or individual affiliation. In inter-dependent architectures, clients connect through a virtual server address to an intermediate node that forwards their requests to the real servers based on a particular server selection policy. We call such intermediate nodes QoS-Brokers when they consider QoS issues in their selection policy. A QoS-Broker is then a public front-end server, that uses up to date information on the state of real servers, to maximize the capacity of the overall system while meeting QoS requirements. The broker has three main functions, interactions with clients, server monitoring, and server selection. The broker's interactions with clients represent the negotiation protocol in a broker-oriented architecture. It defines how clients can specify and communicate their QoS requirements (user profile). The user profile information has to be stored in a place where it will be available for the negotiation process. Also users may be grouped into different categories that has each its own QoS requirements.

4.1 Server Monitoring

The broker needs static and dynamic information on each supported server. Initial static information could be supplied during registration to the broker. This information may include the server's real address, its capacity, the type of requests supported, and the desired operating thresholds. Once a server and a client are connected, the session length will depend on the load of the server, the size of the requested files, and the status of the network connection between the client and the server. To keep the broker up to date, either the servers have to push periodically their performance to the broker or the broker has to request that information. It will be very helpful then if the next generation of electronic commerce servers could support a standard performance reporting capability.

4.2 Server Selection and Load Balancing

Once a service is replicated, the main concern becomes how the clients can locate the best server to use. A broker-based architecture offers transparency to the clients, the latter needs to know

only the address of the broker, while the real servers may come and go transparently. The interaction with the broker can take one of several forms:

1. Clients can choose to go through the broker for every request and the broker forwards the request directly to a server. In this case, the broker has complete control over where the client requests are serviced, and then he could make sure that the load is evenly shared between the different servers. The main drawback of this scheme is that the broker itself could become a serious bottleneck. Also when a user session is split between different servers, scaling techniques that anticipate user requests, like for instance caching and prefetching, may not perform well. This scheme is the one actually used in commercial product like the IBM eNetwork Dispatcher and the CISCO LocalDirector. One of the reasons for this is probably that its implementation does not require any changes to the current HTTP protocol between clients and servers

2. Clients ask for a server to be used for a complete session. This schema reduces the burden on the broker since each client interacts with him only once for each session. However, when the session time becomes variable and long, the broker may loose track of the number of clients assigned to servers, and hence the expected load on them.

3. Clients ask for a server to use for a certain period of time, i.e. the name to address resolution has a time-to-live value (TTL). In this scheme the broker can predict the load on each server in the near future through its control of the TTL value while avoiding to some extent to be a bottleneck. The optimal TTL estimation may be very difficult to define, but a good estimate can be derived from the actual observed values of server performance and client access patterns.

The broker main function, and probably most difficult one, is the server selection. Server selection can be done based on static variables, geographic proximity, or dynamically based on the status of the servers and the changes in user access patterns. The following approaches could be proposed:

- Random selection
- Least number of connections (requests)
- Number of connections per second
- Number of active connections
- Round-Robin
- Weighted percentage between servers with different capacities
- Maximum number of connection (pre-measured capacity limit)
- Geographical proximity between client and server (client IP address)
- Time of day (to anticipate peak hours, and also to profit from idle servers on a far site in the globe)
- Dynamic measured server response time
- The status of the link between the server and the client

5. Conclusion

The successes of electronic commerce will depend on the overall QoS offered to its clients. It is necessary to identify first of all what types of QoS need to be considered, and how could it be

provided in such context. The overall QoS includes the quality of information, the quality of information sources, and the performance by which it is delivered to users. We discussed in details in this paper the different QoS categories and dimensions and how we think they could be considered in the context of electronic commerce. QoS management involves cooperation between the server side and the client side. The objective is to provide a framework in which client's specific QoS requirements are considered and servers' architectures are scalable. We proposed a broker-oriented architecture that can handle user requirements and server scaling transparently from each other. An electronic commerce shopping session can be characterized by short and frequent requests, variable think times, and an undefined number of requests and hence an undefined session length. The broker interactions with users and servers have to be designed carefully with regards to the nature of the electronic commerce shopping.

Acknowledgement: This research was supported by a grant from the Canadian Institute for Telecommunications Research under the NCE program of the Government of Canada.

6. References

[Bochmann 1997] Bochmann, G. v. and Hafid, A., Some principles for quality of service management, Distributed Systems Engineering Journal 4 (1997), pp.16-27.

[Frolund 1998] Frolund, S., & Koistinen, J. (1998). Quality-of-Service Specification in Distributed Object Systems. IOP/BCS Distributed Systems Engineering Journal (December 1998), to appear.

[Dreilinger 1997] Dreilinger, D., & Howe, A. (1997). Experiences with Selecting Search Engines Using Meta-Search. ACM Transactions on Information Systems, 15 No 3 (July 1997), 195-222.

[Hafid 1996] Hafid, A., Bochmann, G. v., Kerhervé B., A quality of service negotiation procedure for distributed multimedia presentational applications, in the proceedings of the Fifth International Symposium of High Performance of Distributed Processing (HPDC-5), Syracuse, New York, 1996, pp.330-339.

[Koistinen 1996] Koistinen, J. (1997). Dimensions for Reliability Contracts in Distributed Object Systems (Technical Report No. HPL-97-119). HP Labs.

[Sairamesh 1997] Sairamesh, J., Nikolaou, C., Ferguson, D., Yemini, Y. (1996). Economic Framework for Pricing and Charging in Digital Libraries, D-Lib Magazine, february 1996.

[Vogel 1995] A. Vogel, B. Kerhervé, G. v. Bochmann and J. Gecsei, Distributed multimedia applications and quality of service: A survey, IEEE Multimedia, Vol. 2, No. 2 (ISSN 1070-986X), Summer 1995, p.10-19. A reduced version was published in Proceeding of CASCON (IBM Toronto), Oct. 1994.

[Wang 1996] Wang, R., Strong, D., & Guarascio L (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. Journal of Information Systems, 12 No 4 (Spring 1996), 5-34.