Transactional Characterization of Front-End E-commerce Traffic

Krishna Kant and M. Venkatachalam
Intel Corporation, Oregon, USA
{krishna.kant | muthaiah.venkatachalam}@intel.com

Abstract— **In this paper, we provide a generic model to characterize transactional behavior in an e-commerce environment which can be applied to Business-to-Business (B2B) e-commerce sites, Business-to-Consumer (B2C) e-commerce sites, and other (e.g., simple browsing) type of web-sites. The ideas presented in this paper have been derived based upon detailed examination of the E-commerce server architecture and traffic from a couple of sites. The paper also studies the characterization of embedded requests. We also discuss about how the transactional and embedded request models can be used in generation of artificial front-end e-commerce traffic in the labororatory for resource engineering of servers.**

Keywords: e-commerce server, transactional characterization, Markov chain, Embedded requests, traffic generation.

I. Introduction

With E-commerce taking a central place in today's economy, the traffic in the e-commerce environment has become an important issue. It has been shown that Internet traffic is temporally quite complex and shows both long-range dependence and nonstationary. Reference [2] shows how to separate long-range dependence and non-stationarity under certain simplifying assumptions. This paper, however, concentrates on the transactional characterization of the e-commerce traffic which can also be quite complex. The main goal here is to enable realistic, synthetic traffic generation in a scalable way in a lab environment.

At the transactional level, e-commerce traffic is significantly more complex than a simple web-browsing traffic because of a variety of activities other than browsing (e.g., hotlist viewing, product search, shopping cart, payment, etc.). The user behavior in terms of moving to different stages in e-shopping can be conveniently described via a Markov chain, however, there are a number of questions that have not been addressed in the literature including (a) order and representation of the Markov chain, (b) representation of the embedded requests, and (c) characterizaton of secure and non-secure requests. We provide a generic methodology for addressing these questions and validate it based on data from a small set of business to consumer (B2C) and business to business (B2B) environments. As usual, individual e-commerce sites vary considerably in their characteristics; however, we believe that the qualitative observations are quite representative of these environments.

A recent hierarchical study of e-commerce traffic can be found in [5]. This work talks about a multi-layer characterization of ecommerce traffic and comes up with interesting statistics on the user behavior in e-commerce sites. Some of the findings include session length distribution, model for capturing individual user behavior, confirmation of the presence of long-range dependence in e-commerce traffic. In this paper, we discuss a somewhat more detailed characterization that includes a generic embedded Markov chain model for transactions and behavior of embedded requests, both with an view of generating the traffic artificially. Section II deals with the overall traffic characteristics in the E-commerce environment. Section III provides a generic transactional model. Section IV applies the model to B2B and B2C example web sites. Section V deals with modeling embedded requests. Section VI talks about artificial traffic generation and we conclude in section VII.

II. E-commerce Traffic Environments

In this section, we consider three types of traffic – B2B, B2C and web-browsing (WWW). Major characteristics of these are indicated. Some of these characteristics have been noted elsewhere [6], while others are based on the data that we have examined.

B2B: This refers to large sites that handle automated business transactions between a business and its partners and other downstream businesses. A close study of these sites indicates a number of unique characteristics, many of which are believed to be typical of B2B sites. In particular, all transactions (rather than just the sensitive ones) are secure (i.e., use HTTPS). Also, the traffic shows a typical day-time busy period pattern (reminiscent of busy-period pattern in traditional telephony systems), although this behavior tends to be washed out a bit because of accesses from a variety of time-zones. An analysis of the transaction types shows a lot of serious business being conducted (negotiation, ordering, order checking, payment, etc.) instead of being dominated by idle "window shopping" or simple browsing. The web-pages are found to be rather simple with little in the way of fancy advertising (meaning, not too much dynamic content beyond what comes from the back-end server). Also, B2B requests come exclusively from business customers who typically have high-speed Internet connections. These characteristics imply that there are fewer errors, retries, etc. B2B systems are typically designed for very high robustness, with a lot of background "healthcheck" transactions between the systems.

B2C: This refers to the typical e-tailer sites. A close study of these sites also indicates a number of unique characteristics, many of which are again believed to be typical of B2C sites. In particular, the busy-period behavior here is a bit hard to identify except for the substantial traffic at nights and weekends as well. As with B2B traffic, the temporal characteristics are somewhat muted by the fact that orders come in from a variety of time-zones. An analysis of transaction types shows not much serious business being conducted, instead, most users do extensive product browsing, often even going to the step of putting things in the shopping cart only to dump them and exit out at the end. With respect to the use of HTTPS transactions, for certain sites (e.g., online banking, stock trading, etc.) the situation is identical to those for B2B sites in that the entire user session runs in secure model. In a e-tailing environment, however, HTTPS transactions are used only for the sensitive parts (e.g., order checkout and payment), and really don't play much of a role. The amount of image and other dynamic information is also directly correlated with the HTTPS percentage. Generally, because of very high cost of SSL processing [1], HTTPS pages tend to be much simpler than non-HTTPS ones. In either case, the higher bandwidth/processing power requirements combined with primarily low-speed client connections usually results in a lot of abort and retry traffic.

WWW: This refers to the web-browsing sites such as corporate or departmental intranets where the servers not only provide access to the content created by the organization itself but also to all external access from within the organization. Thus WWW provides an example of traffic that is not limited to a specific native site but rather a mixture of traffic going to a large number popular and not-so-popular sites. Because of its nature, this traffic is not tied to any particular back-end database or, for that matter, any sort of commercial service per se. Typically the pages are all static in nature and there tends to be little HTTPS traffic and dynamic pages.

We examined the HTTP logs for several consecutive weeks for these traffic types. Fig. 1 shows the arrival process for 24 hour periods for 5 successive days (dotted lines separate day boundaries) in the B2B environment. It is clear that the traffic varies significantly from week to week, and day to day within a week. However, we see the existence of a "busy period" during each day, which is typically between 7:00 AM to 6:00 PM but with some sharp drops during lunch hour. In particular, consistent with typical behavior in a business environment, one could identify a "morning busy period" during 7:30–11:30 AM and an afternoon busy period 1:30–5:30 PM. However, there appears to be lack of stability in the start/end points of the busy period and the behavior during the busy period.

Fig. 2 plots the time series for five days worth of data in the B2C environment for a bucket size of 30s, which is from a representative e-commerce site.[1] The non repeatability of daily traffic profile is again quite obvious. The traffic does not show pronounced morning and afternoon traffic; instead, the traffic shows lot more variability from day to day. In fact, it is hard to identify busy-period per se; all one can observe is low traffic during wee hours. Part of the reason for this may be accesses from different time zones, especially for well-known e-tailor sites. In such cases, the analysis must consider the entire 24 hour period.

III. A Generic Transactional Model

A comprehensive characterization of transactions in terms of their resource requirements is often difficult because the details of the applications running on the site are generally unavailable. Moreover, e-commerce servers often run multiple applications simultaneously on the same platform without adequate information to separate the impact of various applications. Consequently, we consider them as a single canonical application. When available, the server-side scripts used in each application can be useful in getting some insight into the nature of the workload. The sites we examined used Microsoft's active server pages (ASPs) as the scripting engine. Since the ASPs invoked are recorded in the HTTP log, it is possible to do a characterization based on those. We note here that a real e-commerce site might use thousands of ASP scripts, but most of them are simple variants of a base ASP. Although the number of base ASPs may run into 100s, top 10-20 ASPs account for more than 99% of the work. If the actual ASP code is not available, one has to depend on indirect ways of finding their functionality (including site browsing and looking at the HTML code).

In identifying the important transaction types, we determined the frequency of usage of all the ASP scripts invoked over several week's worth of HTTP logs. These were arranged in decreasing order of frequency, and all ASP's that collectively amounted to less than 0.1% of the total ASP samples were simply ignored, whereas many others were grouped together because they performed very similar functions. This still leaves more scripts than what may be really necessary for an effective characterization of the traffic. Thus, the next step is to identify and group the ASPs further driven by their traffic impact. To this end, we propose a generic model in the next subsection and then show how the data observed from a B2B and a B2C site fits this model.

Since the primary purpose of transactional classification is to gain a somewhat detailed understanding of server resource requirements for various transactions, we arrive at the classification by grouping the server side scripts by their abstract functionality. Furthermore, we first do this at a rather high level to get "macro states"; if needed, each macro-state can be further refined into micro-states. For instance, at the highest level, all scripts that involve the use of the backend database server may be classified into one macro-state, all the requests to search server into another macro-state, and so on. In particular, we propose the following macro-states (see figure 3):

- **Static:** This class refers to static web-pages (excluding the embedded requests), some of which might be wrapped within Active Server Page (ASP) scripts. In traditional architectures, these accesses involve streaming of data from the memory to the network interface (NIC), usually with a

[1] All data sources had to be omitted because of privacy concerns.

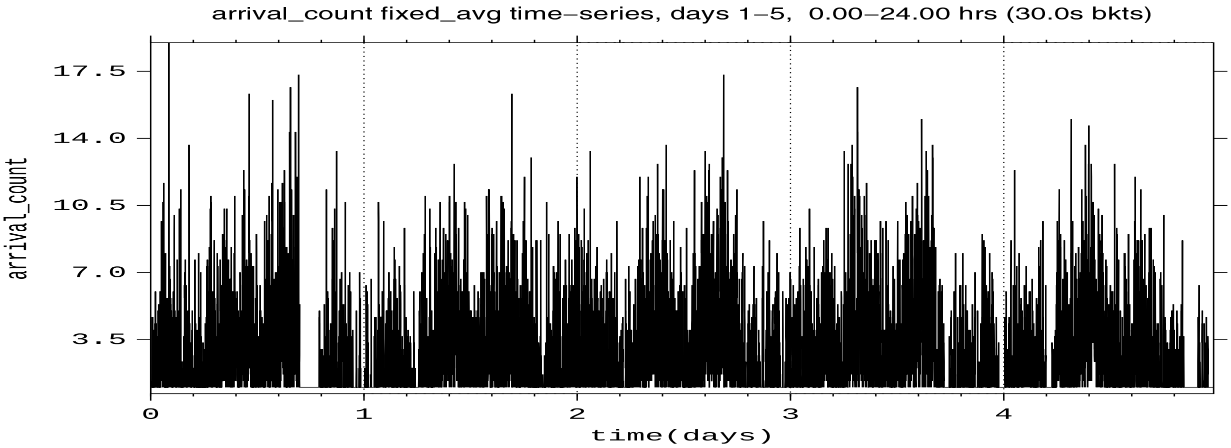

Fig. 1. Five days of B2B traffic, bucket-size = 30s

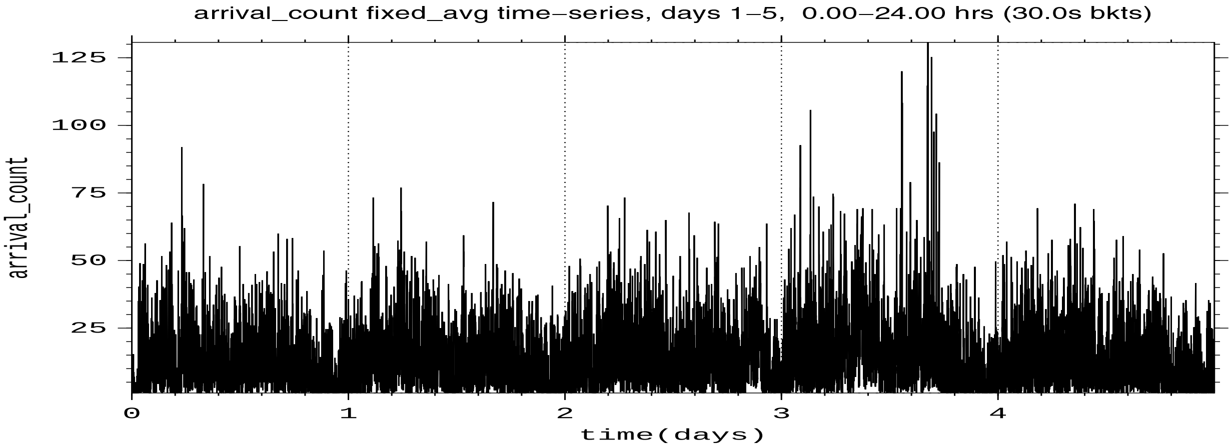

Fig. 2. Five days of B2C traffic, bucket-size = 30s

lot of CPU intervention. Emerging architectures may also do a direct streaming of large pages from the disk to the NIC.

- **Dynamic:** This state represents all the requests for dynamic pages that do not involve accessing the search server or the backend. A typical example would be looking at hotlists or simply constructing the web-page from logos, images, advertisements, text, audio, etc. This process is typically very CPU intensive.
- **Search:** This state represents all the requests that result in a significant search activity (e.g., search for web-pages with given keywords, or search for a product in a backend database). Since such searches are typically run on a separate server (search server or backend database), the impact on the front-end server is typically in terms of handling a large amount of data and communicating with the backend server.
- **Backend:** This state represents all the requests to the backend database (other than the searches). When a significant non-search database activity is involved, this state can be further decomposed into distinct states such as shopping_cart, product_order, order_status, etc.
- **Home:** This is a hypothetical state from which all the users start and end their sessions.

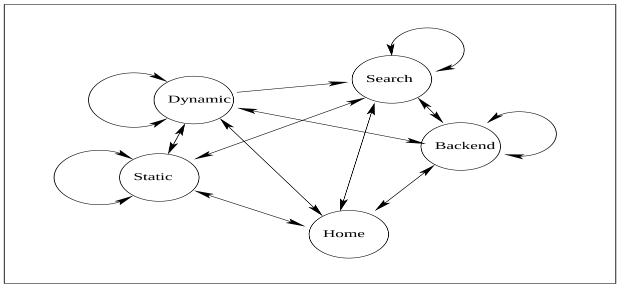

Fig. 3. Generic transactional model. This can be made more specific to the case by expanding relevant states

Although the above classification is intuitive, it runs into trouble if applied literally. Web servers running Microsoft's Internet Information Server(IIS) often tend to encapsulate every web page inside an ASP. ASPs can contain plain HTML along with clearly marked visual basic scripts that generate HTML. The ASP interpreter executes the script part to generate the HTML code, but leaves the existing HTML alone. Consequently, it is possible to simply rename an HTML file as an ASP file, and it would work just fine (it will simply be scanned for scripting commands by the ASP interpreter, but no other effort will be expended on it.) Literally speaking, such encapsulated HTML would be classified as "dynamic", but it is really entirely static. An even more confusing, but frequent situation occurs in product description ASP scripts. Such a script takes one parameter, which is simply the unique id for the desired product. Using the id, the script retrieves the static page describing the product. In this situation, it appears that the ASP script dynamically generates a different page each time, but in reality it may do little additional work than retrieving the static page directly. We have classified such transactions as "static" in the data shown in this paper.

Often, the product description pages are stored as records in the product catalog database. In this case, the script may actually have to go to the backend to obtain the page. However, backend access would be needed only if the page is not already cached in the front-end due to an earlier access. This is one instance where classifying transactions by merely the names of the scripts breaks down. The correct approach is to classify the transaction as backend if the backend access is actually involved, and as static/dynamic otherwise. If such a classification is not possible, one could perhaps take the following approach: (a) assume that all product pages come from the cache and thus the transactions can be described as static or dynamic, and (b) determine the intensity of backend transactions based on overall access characteristics to the backend. In this approach, a one to one correspondence between front-end transactions and back-end accesses is lost.

One important issue that we need to address in this classification is the handling of secure HTTP transactions since they have a very substantial impact on server performance. In situations where all transactions are secure (i.e., B2B sites, and banking/trading related B2C sites), no further characterization is needed. In other cases, there are two ways of handling secure transactions:

1. Split each relevant macro-state into secure and unsecure sub-states,
2. Treat security merely as an attribute applied to each transaction probabilistically.

The main virtue of approach (a) is that would correctly model the secure phase of a user session (i.e., purchase related interaction in an e-tailor environment); however, to be able to represent this clumping, one would have to typically split almost all the states into secure and non-secure sub-states. This not only increases the number of states but more importantly the number of parameters needed to calibrate the model. The approach (b) merely needs the percentage of secure transactions but cannot retain any sequencing information. If HTTPS transactions generally represent only a small portion of user session (usually true in e-tailer environment as mentioned earlier), the simpler approach (b) is quite adequate.

Given the states, a user session can be represented by transitions between these states. Each user starts and ends the session in the home state. During the time that a user is in a given state, it may issue one or more transactions relevant to that state. The transitions between states could, in general, depend on more history than just the current state. In fact, in some cases, we found significant second order effects (i.e., a transition's dependence on last two states visited). Thus, in general, user sessions can be described by a kth order semi-Markov chain (with an appropriately chosen k).

IV. Two example applications of the Generic Model

In this section, we apply the generic model to the B2B and e-tailer B2C sites discussed earlier. In particular, we decompose the macro-states into micro-states as appropriate and show some numerical results obtained from the analysis of the HTTP logs and server side scripts from these sites.

A. E-tailer B2C site

A typical characteristic of the e-tailer B2C environment that we analyzed is that most of the traffic concerns "window shopping" (searching products, retrieving production information, comparing prices, etc.) and very little had to do with actual product purchase. In fact, most of the entries made to the shopping cart are done merely as a "bookmark" and are often discarded eventually. Thus a single "backend" state is quite adequate, with the payment transaction part of it considered as secure. The "dynmaic" state could be decomposed into "image" and other types of content since the images were served by a separate server. However, for simplicity, we don't do this in the shown example.

Assuming a first order embedded Markov chain model, Table I shows the transition probabilities. The first column in Table II shows the probabilities of various states using the embedded first-order Markov chain model. It is seen that although 29% of the visits are spent in the backend state, nearly all of the accesses concern accessing product description pages, rather than going through the purchasing steps. The other two columns in Table II show the mean and standard deviation of the residence time in milliseconds in each state. It is seen that residence times are highly variable in most cases (a standard deviation of 6-8 times that of the mean). Note that since embedded requests are ignored in this analysis, the residence time in a state includes the time to request and load embedded pages as well.

In order to verify the assumption of a first order Markov chain (MC) model, we also carried out an extensive analysis of the HTTP logs to identify higher order transitional dependencies. It was found that in most instances, the depencence on prior states was quite weak, however, there were some significant exceptions. In fact, in a second order MC model, the following sequences had associated probabilities that were about an order of magnitude higher than all others:

1. Static → dynamic → static,
2. Dynamic → static → dynamic, and
3. Dynamic → search → dynamic.

Given the close relationship between dynamic and static pages, a substantial chance of alternating sequences of them is not surprising. In fact, one could even argue from here that the "static" and "dynamic" states should perhaps

state	static	dynamic	search	backend	home
static	0.341	0.062	0.017	0.196	0.384
dynamic	0.089	0.184	0.016	0.470	0.242
search	0.044	0.024	0.444	0.192	0.297
backend	0.099	0.073	0.042	0.384	0.403
home	0.139	0.413	0.071	0.377	0.000

TABLE I
State transition probabilities in the B2C scenario

name	no	state-prob	mean	stddev
static	1	0.120	102.3	309.7
dynamic	2	0.191	43.8	220.6
search	3	0.051	89.1	459.2
backend	4	0.293	96.5	348.9
home	5	0.345	1.0	0.0

TABLE II
Markov chain state properties for the B2C environment

be merged into one. The main reason to keep them separate is significant differences in processing requirements of the two. The third sequence also appears reasonable in view of the fact that search is often preceded by a POST operation and followed by retrieval of the listed product information.

The existence of only a few long dependency sequences suggests that a direct use of k-th order MC (with $k > 1$) is quite wasteful and not very robust. Instead, model can allow for longer sequences explicitly while keeping the rest of the MC as first order.

B. B2B site

Since the B2B sites typically involve a lot of transactions that require backend access (business negotiations or purchasing), it is generally useful to split the backend macro-state into a number of micro-states representing frequently invoked operations. Since the examined B2B site was product purchase oriented, the following micro-operations were selected based on the frequency of occurrence:

1. **price_avail:** Checks the price and availability of the products.
2. **order_prod:** Handles placement of an order.
3. **order_status:** Checks the status of an order.
4. **change_req:** Makes changes to an existing order.

In this example, the dynamic macro-state primarily consisted of just one script - hot_list, which examines the current "hot-list" and selects products from them. The static pages consisted of most of the other ASP scripts (which, as stated earlier, simply retrieved static pages based on one or two parameters). Also, all user interaction with the server occurred via HTTPS where a secure connection is established at the beginning of the user session. Table III shows the transition probabilities for this environment. One immediate conclusion from this matrix is that all states are "sticky", i.e., the Markov chain stays in the same state with a very high probability. The data extracted for each session shows that the client issues a large number of page views for each core operation before moving on to the next step. Table IV shows the state probabilities and statistics of residence times. It is clear that purchasing forms a very substantial part of B2B activity and thus a finer grain characterization of backend activities is essential. As with B2C, the state residence times have a high variability and thus cannot be reasonably modeled as exponential.

state_name	no	residence	Residence time	
		prob	mean	stddev
change_req	1	0.057	590.0	1810.7
dynamic_page	2	0.050	482.3	1753.6
order_prod	3	0.124	448.1	1524.1
order_status	4	0.230	750.0	2525.7
price_avail	5	0.127	814.4	2735.4
search_prod	6	0.020	195.3	841.4
static_page	7	0.116	239.9	1266.3
home _state	8	0.275	110.2	1145.6

TABLE IV
Markov chain state properties for the B2B environment

We also looked at second and higher order dependencies in this environment and surprisingly we couldn't find any that were significantly higher than the rest. It is not clear if this is just accidental, or says something about the nature of B2B site traffic.

In addition to the regular client (or user) transactions, our B2B environment had a number of "system" transactions that are routinely sent to the server by the transactor for health-check. We found that the system transactions amount to about 21% of the total transactions, and thus have a substantial influence on performance. However, since system transactions are very much site specific, we ignore them in this paper.

V. Modeling Embedded requests

The HTML code for a web page often contains references to other objects (e.g., images, javascript for animation, etc.) which are not an integral part of the "base page" and are instead loaded via explicit requests from client's browser when it interprets HTML code. These are known as *embedded requests* and they naturally follow shortly after the base web page is received. The number of embedded requests per base page has been steadily increasing and could range from a few to a few tens. Embedded requests are not important from a transactional analysis point of view, since they can be considered as an integral part of the base transaction. However, we still need to be able to generate embedded requests when generating e-commerce/web traffic artificially. In this section, we look at how to model embedded requests in the e-commerce environment and thereby use it in synthetic traffic generation.

Figures 4 and 5 show the probability mass function of embedded image requests in the B2C and the B2B environments respectively. Apart from image embeddings, the other embedding that we came across was HTML embedding. But, we do not present the results of HTML embedding as they were few and far apart.

Table V shows the statistics for the embedded images in the B2C and B2B scenarios. Firstly, we observe that

current state	change request	dynamic page	order product	order status	product avail	search prod	static page	home state
change_req	0.9215	0.0047	0.0101	0.0276	0.0085	0.0003	0.0123	0.0150
dynamic_page	0.0077	0.8816	0.0152	0.0388	0.0123	0.0108	0.0130	0.0206
order_prod	0.0054	0.0044	0.9151	0.0313	0.0107	0.0047	0.0119	0.0166
order_status	0.0080	0.0063	0.0186	0.9180	0.0163	0.0018	0.0116	0.0194
price_avail	0.0053	0.0045	0.0141	0.0332	0.8964	0.0008	0.0139	0.0318
search_prod	0.0032	0.0363	0.0548	0.0430	0.0098	0.8256	0.0159	0.0114
static_page	0.0126	0.0083	0.0241	0.0551	0.0283	0.0019	0.8568	0.0129
home_state	0.0033	0.0031	0.0071	0.0276	0.0146	0.0001	0.0154	0.9287

TABLE III

Transition matrix for the first order Markov chain in the B2B environment

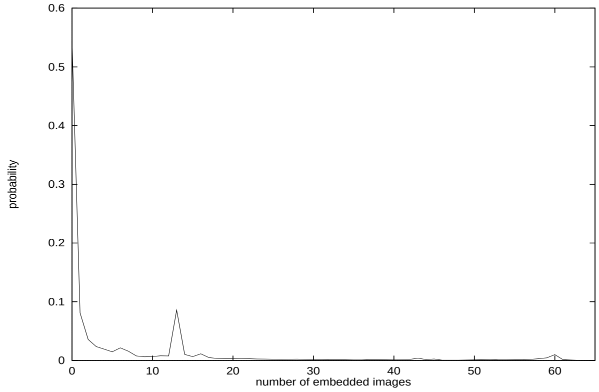

Fig. 4. Embedded image probability distribution for B2C scenario

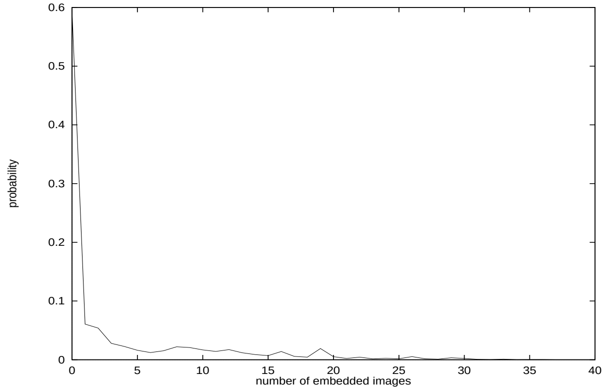

Fig. 5. Embedded image distribution function for B2B scenario

Environment	95 percentile	mean	stddev
B2C	37	6.22	12.71
B2B	19	3.58	6.58

TABLE V

Statistics for the embedded images

the number of embedded images per base request is fairly variable as the standard deviation is around twice the mean value. Secondly, we see that the B2C environment has more embedded images than B2B environment, which is compatible with our earlier comments.

VI. Synthetic Traffic generation

As mentioned earlier, one of the main goals of this characterization is to enable synthetic traffic generation for server performance analysis. Reference [3] discusses a traffic generator called Geist for this purpose. Basically, Geist first generates aggregate traffic coming at the server in order to better emulate properties such as self-similarity, non-stationarity, etc. This aggregate traffic is then divided up into multiple streams, each one corresponding to a virtual user. For each such user, an embedded Markov chain controls the transactional behavior as discussed here.

With each transaction, a random number of embedded requests are generated. This random number is controlled by the probability mass function, described in the previous section. For all the details, the reader is referred to [3].

VII. Conclusions and Future Work

In this paper, we looked at the characterization of user sessions in the e-commerce environment. For this purpose, we devised a generic embedded Markov chain model for capturing the individual user behavior. We also addressed the issue of modeling embedded requests. Along the way, we pointed out some of the significant differences between the B2B and B2C environments and showed that despite the differences they can still be characterized together in a coherent manner due to inherent commonalities.

Our work so far has not carefully examined the back-end transactional characteristics and their relationship to front-end transactions, primarily due to lack of available data. A detailed study in this regard is essential for a good classification methodology, and plan to pursue this further. We also do not have adequate data to characterize user abandonments and retries and its impact on the aggregate traffic.

References

[1] K. Kant, R. Iyer and P. Mohapatra, "Architectural Impact of Secure Socket Layer on Internet Servers", Proc. of ICCD, Sept 2000.

[2] K. Kant and M. Venkatachelam, "Modeling traffic non-stationarity in e-commerce servers", available at `kkant.ccwebhost.com/download.htm`.

[3] K. Kant, V. Tewari and R. Iyer, "Geist: A generator of e-commerce and internet server traffic", Proc. of ISPASS, Tucson, AZ, Nov 2001, pp 49-56.

[4] K. Kant, "On Aggregate Traffic Generation with Multifractal Properties", proceedings of GLOBECOM'99, Rio de Janeiro, Brazil, pp 1179-1183.

[5] D. Menasce, V. Almeida, et al, "In Search of Invariants for E-Business Workloads", in the Proc. of 2nd ACM conf. on electronic commerce, Oct. 17-20, 2000.

[6] W. Rajput, *E-commerce systems architecture and applications*, Artech House, Aug 2000.