End-To-End Scaling: The Response Time Pipe

Dr. Tim R. Norton
Simalytic Solutions, LLC
CMG 2001, Session 3208, December 4, 2001

The notion of scaling used to mean "How big do we have to make the server?" Unfortunately, the new e-business approach to applications has spread the work across many different components owned by many different organizations. Now, when we talk about scaling, we have to address networks, storage, security, application architecture, and even other entire applications. This paper proposes an approach to modeling business transactions to determine what is impeding the business process, what to do about it, and the effect of taking that action.

1. Introduction

When we talk about 'scaling' an application we're interested in how much bigger it can become. If it processes 1,000 orders a day now, will it be able to process 100,000 orders a day six months from now? In other words, will it scale by a factor of 100? In the past, that type of scaling meant "How big do we have to make the server?" Times are different with the new e-business approach to applications because work is not only spread across many different components, but often some of those components are owned by different organizations within the company or even by different companies. Now, when we talk about scaling, we have to address many different components of different types, such as networks, web servers, database servers, and storage devices. The relationships between the components are complicated by security issues, application architectures and services provided by other companies. Such services, such as B2B (Business-To-Business) solutions for functions such as credit card processing and distributed document management, are really entire applications but they appear to be a single component. This paper proposes an approach to modeling business transactions to determine what are the major obstacles impeding the business process, what are the actions needed to minimize the effects of the obstacles, and because there are often mutually exclusive alternatives to dealing with obstacles, what are the effects of taking different actions.

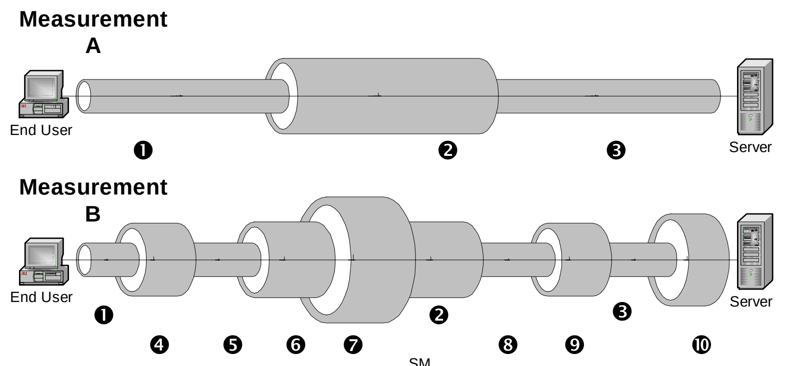

Figure 1: The Response Time Pipe^{SM}

❶ end-user connection to local LAN
❷ hosting site ISP (Internet Service Provider)
❸ hosting site application server
❹ end-user's LAN environment
❺ end-user's corporate firewall
❻ end-user's corporate ISP
❼ Internet backbone
❽ hosting site firewall
❾ hosting site web server
❿ hosting site database server

Of course, before there can be any view of the future, there must be an understanding of the present. Measuring the overall, or end-to-end, response time of web applications provides the understanding of what is happening; and a model of the response time provides the view of what will happen. By changing component behaviors in the model, different actions can be explored and different views of different futures exposed. Most of the time, the behaviors we want to change are those related to bottlenecks somewhere in the path the transaction follows.

The ability to identify bottlenecks in the path of a web transaction is directly related to the granularity of the measurements along that path. To visualize this process, think of the path the transaction takes as a pipe that connects the end-user with the web server as shown in Figure 1. If only one measurement is available, then the section of the pipe that it represents will appear smooth and straight. If several measurements are available then the major bottleneck can be identified as one of the sections with reduced capacity. If there is a single small pipe section, then we cannot tell the difference between a pipe that has uniformly poor performance and a pipe that performs very well except for a single constriction or bottleneck. Unfortunately, the granularity must be relatively fine before corrective action can be determined. It is just not acceptable to tell end-users that the server is performing fine so the problem must be ‘somewhere’ in the Internet. Figure 1 shows two examples of a Response Time Pipe^{SM} for an application. *Measurement A* shows three measurement sections and *Measurement B* shows the same connection with ten measurement sections. The identifiers on each of the sections are only for illustration as to what *might* be measured, not to identify the actual components. There are many more components involved in a connection such as this, but from the standpoint of transaction response times, many components will be hidden because they are not directly measured. Ideally, each component should be measured and treated as a section of the RTP^{SM} (Response Time Pipe), but that is seldom practical. What is measured for any given section of the pipe often determined simply by what is already being measured or what can be measured easily and quickly.

When looking at transaction based applications, it is easy to focus on the methods of measurement that are well known and comfortable. However, web applications are seldom measured in these terms. (Some planners even question if they are *measurable* in these terms!). Instead, they are measured using metrics like ‘hits per second’ (the number of web page retrieves per second) and ‘stickiness’ (how long a user stayed at the site and clicked on different links) that don’t translate well to the transaction modeling techniques that we have used for so many years. Transactions vary in how many web pages are required to complete the business functions, and web pages vary in how many artifacts (images, frames, ads, text, etc.) are retrieved. It is very difficult to relate such low level traffic to the overall business. The argument has been made that business transactions will meet the overall service objective if each of the components meets its service objective. However, variability within a component often requires that its service objective have a broad range to avoid false problem alerts. Normally the responsiveness of all the components average out across the transaction path, but sometimes several components are on the high end of the acceptable range at the same time, which causes the overall transaction to fail to meet its objective. Also, it is easy to see from Figure 1 that there are many places in the Response Time Pipe where a component of higher capability may have to wait for a component of lower capability. These narrow sections of the RTP can account for traffic delays due to congestion and queuing, as a transaction moves from one section to another. Of course, the political process for measuring each component wants these delays accounted for in the responsiveness of the other components to avoid missing performance objectives that effect bonuses and promotions.

Techniques to measure individual systems have been well understood for some time. Techniques to measure applications have gained sophistication and popularity over the last several years. The challenge now is to provide a business focus that gets the best return on application measurement while reducing the cost of data collection and analysis. The Response Time Pipe facilitates this objective by measuring each section at the highest level and collecting detail measurements from the problem sections but not from the sections that are performing well.

1.1 Background

Today’s computer environments must be viewed with the objective of understanding how the systems meet the end user’s requirements. By addressing the business needs, we avoid viewing the computing environment as an end in itself, and we can then relate the benefits of an application to the business that depends on it. Unfortunately, there are many pressures that try to shift the focus to advances in technology without regard for what the computing environment provides to the business it was intended to support. Advances in technology lead to a very rapid change, and it is often difficult to relate the value of that change to the overall business in an objective manner. (Business in this context means more than a for-profit company. It can include any type of company, institution, agency, or organization with an overall objective, be it revenue, service or regulatory.)

Measurement has traditionally focused on resources (CPU utilization, I/O rate, etc.) and workflows (job throughput, internal transaction response times, etc.) to determine if a given system is “good enough” to service a workload (which, in theory, translates to an application). Today, application measurement in large computer installations with multiple systems requires an understanding of not only operating systems, platforms, clients, servers, networks, transaction systems, etc., but also the relationships between them and the business objectives (such as staffing levels and “widgets” sold). This relationship allows business managers to understand the impact of application responsiveness on the overall business objectives. Instead of analyzing individual systems, the responsiveness of the application needs to be understood across the entire enterprise to insure that the computing environment addresses the requirements of the business objectives and goals. But this understanding requires not only measurement of both the application (end-to-end response time) and the individual components (internal response time), but also an understanding of the relationship between the two. The focus of the Response Time Pipe is to provide the understanding of the relationship between application responsiveness and individual component responsiveness.

2. Response Time Measurement

The idea of response time measurement has been around for a long time. Early mainframe transaction systems, such as CICS and IMS, quickly developed robust measurement facilities. Often referred to as *internal* response time, these measurements reflect the time from when the host system receives the transaction until a response is sent back to the user. The application was considered to be performing well if the internal response time was within prescribed limits. Network time was generally considered a separate (external) problem to be dealt with by the network support organization.

The proliferation of multiple-tier client/server applications has made response time measurement much more complex. There is no longer a single place to collect the measurement information needed to determine how an application is performing. Network time is now interwoven with the response times of other components of the application and cannot be easily deferred to another organization. Furthermore, even if all the transaction data is collected, it often doesn't contain enough information for a planner to understand the impact of the application's performance on the overall business.

2.1 Measurement Techniques

The Response Time Pipe doesn't provide any new techniques for measuring transactions, but instead, it provides a new way of using the measurements from existing techniques. The details of various techniques for measuring response time are available in the current literature (Knight and Haworth 1996; Lipovich 1997; McBride 1997; Ramanathan and Perry 1999; Smead 1998; Smith and Williams 1998; Thompson, Muñoz, and DeBruhl 1997; Tsykin and Langshaw 1999). Some of these concepts are briefly described below. The interested reader is encouraged to refer to these, and other, papers on various aspects of the general topic of application response time measurement for specific implementation details.

Tsykin and Langshaw (1999) provide a good overview of the general techniques. They list four broad techniques for application measurement:

- Application instrumentation: modifying the application at the source code level to collect performance data.
- Client instrumentation: inserting hooks into the client environment to collect data on activities such as operating system interrupts and/or messages (as with Microsoft Windows).
- Wire Sniffing: monitoring, decoding and analyzing either raw network (sniffer) traffic or server network packets (i.e., TCP/IP).
- Benchmarking: application scripts periodically executed and measured.

Although there are variations of these techniques in both the cited references and other sources, the general concepts fall into the four broad categories above. There are many factors involved in any decision to measure an application. Because each of these authors has focused on a somewhat different combination of these factors, their conclusions and suggested solutions are more focused toward one of the four categories than the other three.

Tsykin and Langshaw (1999) are concerned about the volume of data and processing required to correlate individual units of work across a complex enterprise, so they advocate a technique based on client instrumentation that characterizes user work patterns, instead of collecting detailed transaction data.

McBride (1997) focuses on the need to identify and measure the business transaction and advocates the use of application instrumentation with ARM (Application Response Measurement).

Smead (1998) provides an in-depth analysis of different application instrumentation techniques, but is focused on the low-level IT transaction, rather than the high level business transaction.

Lipovich (1997) takes an application level, rather than resource usage, view but looks at the components of the application response time from a server-centric perspective.

Smith and Williams (1998) provide an interesting approach to understanding application design for modeling with the use of Message Sequence Charts to describe application behavior. These charts provide a very clear representation of how messages and functions flow between servers or components in an application. Their use of these charts to represent three types of CORBA-based synchronization (synchronous, deferred synchronous and asynchronous) highlights some of the pitfalls in selecting measurement points and attempting to correlate resource usage to end-to-end response time.

2.2 Measurement Points

The concept of measurement points was introduced by the author to promote a top down approach to measuring transactions. The Response Time Pipe is an extension of that idea. A full explanation of measurement points is available in (Norton 1999). By starting with the business needs and functions, the measurement effort can focus on actions which support business improvement to gain quicker acceptance by management and other sponsors. The Response Time Pipe allows the data collected to be easily identified as either required or unnecessary.

The reader is reminded that this is not intended to be a definitive, detailed description of transaction measurement techniques. Each real client/server application will differ. There are many measurement tools to chose from, some of which may implement other techniques. Omission of a tool or technique only indicates the author is not familiar with it, not that there is any reason not to use it. The purpose of this short discussion is to encourage critical thinking about what is being measured and where the measurements are taken (measurement points). It is expected that readers will identify measurement points supported by tools with which they are experienced. This is not intended to imply any negative connotation to these, or any other, techniques, but to encourage readers to investigate what is being measured, how it is being measured and the relationship between the measurements, the transactions and the resulting business value of performance trade-offs to a given application.

3. The Response Time Pipe

When building a Response Time Pipe, the overall objective is to develop a set of transformation formulae that are then combined to provide the total transaction response time. The formula for each section of the RTP is based on that section's contribution to the overall response time. (Note: All of the formulae presented here are for illustration only. Each application requires the creation of appropriate formulae for each section of the Response Time Pipe.) To create the set of formulae for a section of an RTP, start with the lowest level measurement data available and create a formula for the response time of whatever it measures, such as network packets or web server hits. Each successive formula within the formulae set for that section builds on the prior formula, until a response time number is derived for use in the overall response time formula (see Equation (5) on the next page). Different measurement units and scales can be used for each section because the final response time value derived is for the impact of that section on a single business transaction. The final scale for the response time values for each of the RTP sections will be the same because it must match the scale used for the business transaction objective (i.e., if the business transaction objective is expressed in seconds per transaction, then the response time values for each section of the RTP must also be in seconds). The remainder of this section of the paper shows the creation of an RTP using a network section as an example. The creation of the formulae shown in Equations (1) through (4) make up a formulae set for one section of the RTP and would be repeated for each RTP section (the number of formulae required for a formula set will vary depending on the measurement data available and the type of RTP section).

The objective of the first formula is to have some low level measurements for each section of the RTP. In fact, you can think of each RTP section as the part of the transaction path that a particular measurement covers. For example, in Figure 1B, measurement data for section 4 (end-user's LAN environment) might be from sniffer data for the backbone section of the end-user's corporate LAN. The number of network packets and the response times between each packet sent and its acknowledgment are what can be measured. An example formula might be something like:

$$r_4 = \text{SUM}\ (\ (k_4^1 - j_4^1),\ \ldots,\ (k_4^n - j_4^n)\ )\ /\ n \qquad (1)$$

where r_4 is the average response time of packets, j_4 is the timestamp of a packet (j_4^1 for the first packet measured through j_4^n for the nth packet measured), k_4 is the timestamp of the packet's acknowledgment, and n is the number of packets, all for section 4 of the RTP. Care must be taken to use the actual packet acknowledgment and not the response to the request in the packet. For example, if the packet contains an HTTP GET request, the acknowledgment is the TCP/IP ACK, not the web page requested. Some network protocols do not acknowledge every packet or block acknowledge every nth packet. Such behavior must be considered when creating this formula.

The objective of the second formula is to understand the relationship between the section metric and the business transaction. This requires some form of correlation between the two, either by identifying which of the low level items being measured (network packets in the example) belong to each of the transactions or by running the transactions in a standalone environment and just measuring all of the low level items. An example formula might be something like:

$$p_4 = q_4\ /\ t * i \qquad (2)$$

where p_4 is the average number of packets per transaction, q_4 is the count of the number of packets associated with the transaction for section 4 of the RTP and t is the measured number of transactions per second and i is the number of seconds over which the packets were counted.

The objective of the third formula to understand the load on the measured section of the RTP. It is very important to understand the difference between the current load from the transactions of interest and the load from other traffic. The predictive value of the Response Time Pipe depends on also understanding the future load of both. In the above example, assume the transaction we're interested in contributes $t*137$ packets per second to the section load, where t is the measured number of transactions per second. Load from other workloads must be measured or estimated independently. It is also extremely important to understand the variability of the other workloads. If they are uniform over the measurement interval then they will affect all transactions in the measured workload equally. However, as their variability increases, their impact will become more erratic, thus reducing the accuracy of this formula. The measurement interval may need to be reduced to decrease workload variability. The future load is based on the business forecast. Therefore, the load in a section might be expressed something like:

$$u_4 = ((t * p_4 * s_4 * f) + (x_4 * y_4 * z_4)) / c_4 \quad (3)$$

where u_4 is the utilization of section, t is the measured number of transactions per second, p_4 is the number of packets in section 4 for each transaction, s_4 is the average size of the packets for the transactions, f is the forecast multiplier (1 for the current load, 1.3 for projection of 30% more load, etc.), x_4 is the measured packets per second for all other workloads, y_4 is the average size of the other packets, z_4 is the forecast multiplier for other work in the section, and c_4 is the total capacity, all for section 4 of the RTP. The load in this case is expressed as utilization because that's what is needed in Equation (4) below.

The objective of the fourth formula is to develop a transformation function from the measured units to response time units. Following with the same example, assume we observe that there are 137 network packets (with accompanying acknowledgments) for each transaction and that the packet response time increases by a factor of three times the section utilization above 40%. (Please note that this is a completely hypothetical example with no basis in real measurements. These formulae, relationships and measurement values are for illustrative purposes only and should not be used without modification and verification.) We can now express the impact of this section of the RTP with a formula like:

$$R_4 = IF(u_4<0.4,\ r_4 * p_4,\ (r_4 * (u_4 * 3)) * p_4) \quad (4)$$

where R_4 is the transaction response time impact, r_4 is the average response time of packets (from Equation (1) above), p_4 is the average number of packets per transaction (from Equation (2), 137 in this example) and u_4 is the utilization from Equation (3), all for section 4 of the RTP. Because each section of the RTP has its own transformation formula that is based on a measurement of that section, the type of metric used must measure only that section, and not the remaining sections, of the RTP. If a metric also measures an adjacent section of the RTP, which is also included in the final combining formula, then the final transaction response time value will be incorrect (effectively double counting a section of the RTP). The complexity of the transform function is dependent on the available measurements and the desired accuracy.

The objective of the fifth formula is to combine all of the transformation functions into a single formula that predicts the overall, or end-to-end, response time by combining the results from each section of the RTP. In it's simplest form this combination will be a summation, such as:

$$R = R_1 + R_2 + \ldots + R_n \quad (5)$$

where R is the total transaction response time and R_1 through R_n are the calculated response times for sections 1 through n. There is no restriction on the complexity of this formula except what is practical to collect data for and to solve. In fact, it is quite reasonable to use the results or metrics from one section in the part of the combination formula for a different section. For example, high loads in one section might cause queues to build up in another to the point where packets are rejected and performance suffers due to increased retries.

4. Transaction Measurement

For any given application, understanding the measurement points (where measurement data should be collected in the application) is a function of both what the application does and the objective of the measurement. What constitutes a transaction? How do we count them? What is the business impact if there are more (or fewer) transactions than expected or if they take more (or less) time to complete? Measurements for Service Level Management (Service Level Objectives and Service Level Agreements) may satisfy a political need, but they will be frustratingly useless if they do not provide enough information to determine what is causing the application response time to fail to meet the service objective. On the other hand, large amounts of resource-centric information (i.e., CPU utilization, network segment utilization, I/O rates, etc.) doesn't help the decision makers understand the impact to the application at the business level. The Response Time Pipe provides a way to take all of the measurements, from many different measurement points, and combine them into a single response time value that can be verified using existing end-to-end measurement tools. The measurements don't have to be in the same units or even of the same type. The only requirement for building a Response Time Pipe is that there must be a way to connect the measurements for a given section of the RTP with the transactions being modeled. Verification then becomes very important but it also can be done pragmatically. Measurement data from different sample times can be loaded into the model and verified against the end-to-end measurements to insure that the results are consistent.

Once an overall Response Time Pipe model is built, it should be verified that its result is reasonably close to measured response times. Then the overall response time value can be used to determine the impact of changes to different sections of the RTP. The objective is not to predict the exact response time of a transaction under all conditions, but rather to see how major changes effect the overall response time, so that their business value can be determined.

End-to-end response time measurements must be related back to the business impact. Response time is a valid metric only if there is some valid business reason to use it. An example of this relationship is shown by relating the responsiveness of an application supporting an order entry call center to the number of calls an operator can handle in an hour. The fewer calls the operator handles means the more operators that are required for a given call volume. That relationship provides the information necessary to make the business decision of upgrading the server or hiring more operators (Norton 1998).

Understanding the overall end-to-end transaction response time only addresses part of the problem. The end-to-end transaction response time is directly related to bottlenecks or constrictions in the RTP. A single poorly performing component will cause increased response time, regardless of the speed and capacity of all of the others components. The Response Time Pipe doesn't provide a detailed analysis of any of the sections. Instead, it provides a way to determine if improving the responsiveness in one section makes a significant enough difference to the overall response time to improve the business. Continuing with the same example from above, the overall effect on response time can be examined for different alternatives, such as reducing either s_4, the average size of the packets for the transactions in Equation (3), or p_4, the average number of packets per transaction in Equation (4). These are two very different actions and require different implementation. Either change will have an effect on the results of Equation (5). Then the question becomes which has the greater desired effect and what is the cost to implement that change. The Response Time Pipe helps understand which alternative provides the greatest return in the form of improved response time. How to implement either one or the trade-offs in terms of costs, such as implementation and operations issues, are best addressed by other, more traditional, techniques, but those decisions are much easier when the effect on the overall business is known. The value of the Response Time Pipe is gaining the understanding as to where to apply those techniques without having to try them everywhere to determine that only a few have a return to the business.

In the final analysis, the measurement objective has the greatest influence over the type of transactions to measure. Business related questions will focus attention on business transaction measurement and resource related questions will focus measurement on IT transaction measurement. While it is certainly possible, and even desirable, to collect both measurements, any correlation between the two must be done very carefully. The Response Time Pipe provides the formal structure to develop and define that relationship.

5. Simple Example

Now let's look at a *very* simple example of a Response Time Pipe that only has two sections, one network and one web server. Assume section 1, the network section, is as described above in section 3, The Response Time Pipe. Because this example is just to illustrate the RTP, we will use some extreme simplifying assumptions, such as the effect of load on network utilization in Equation (4) and the effect of server load below. These are not realistic assumptions but adequate for the purposes of illustration.

For RTP section 1, assume:

$r_1 = 0.3$ seconds from (1)

$p_1 = 180000/10*900=20$ (2)

$u_1 =((10*20*300*f)+(2500*700*z))/5000000$ (3)

$u_1 = 36\%$ if f=1, z=1 (3)

$u_1' = 47\%$ if f=10, z=1 (3)

$u_1'' = 71\%$ if f=1, z=2 (3)

$R_1 = 0.3 *20 = 0.6$ for $u_1 < .4$ (4)

$R_1' =0.3*(0.47*3)*20=8.46$ for $u_1 > .4$ (4)

$R_1'' =0.3*(0.71*3)*20=12.78$ for $u_1 > .4$ (4)

For RTP section 2, assume that we measured web server hits at the server with the following values (please remember that this is an extreme example, greatly over simplified, for illustrative purposes only):

Hits per transaction = 15

Average response time per hit

= 0.7 seconds if hits/sec < 300

= 1.9 seconds if hits/sec > 300

Average response time per transaction

= 15*0.7 = 10.5 seconds if hits/sec < 300

= 15*1.9 = 28.5 seconds if hits/sec > 300

$R_2 = 15 *0.7 = 10.5$ for $h_1 < 300$

$R_2' = 15 *1.9 = 28.5$ for $h_1 > 300$

For the overall analysis we can look at several alternatives:

Current response time:

$R = R_1 + R_2 = 0.6 + 10.5 = 11.1$

Increased network load response time (f=10):

$R = R_1' + R_2 = 8.46 + 10.5 = 18.96$

Increased network load response time (z=2):

$R = R_1'' + R_2 = 12.78 + 10.5 = 23.28$

Increased server load response time (h>300):

$R = R_1 + R_2' = 0.6 + 28.5 = 29.1$

The way these response time values are assessed depends on the business transaction response time objective, so assume that 20 seconds is acceptable for this business, based on what the transaction does and the nature of the business. In that case, we can see, at a high level, that a ten fold increase in transaction network volume will still result in acceptable response times. However, either doubling the other network traffic or increasing server, load will cause the response time to be unacceptable. If this is the only transaction using the server then it will generate 150 hits per second (10 transactions per second ∗ 15 hits per transaction). Therefore, doubling the transaction volume will not cause a response time problem from the network perspective but will exceed what the server can absorb without increased response times.

This example shows how some very quick and simple calculations can provide a great deal of insight about transaction response times, especially when coupled with the business objectives and expectations. It also shows how easy it is to jump to wrong conclusions when the interrelationships between RTP sections aren't considered, as in the case above of increasing the transaction arrivals in the network section but neglecting to propagate that increase to the server section. Such errors can be avoided by implementing the RTP with some calculation tool, such as a spreadsheet program.

In addition, the RTP results for different alternatives can be connected with an overall business process model to see the business level impact of the changes. Assume the transaction supports some type of call center that receives 3,000 calls during the peak hour. If it takes an operator an average of 15 transactions plus 3 additional minutes per call, then 289 operators are required, as shown by:

$$289 = 3000 / (60 / (((15 *11.1)/60)+3))$$

For the increased network load alternative, 387 operators are required because the transaction response time goes up to 18.96 seconds, as shown by:

$$387 = 3000 / (60 / (((15 *18.96)/60)+3))$$

For the increased server load alternative, 514 operators are required because the transaction response time goes up to 29.1 seconds, as shown by:

$$514 = 3000 / (60 / (((15 *29.1)/60)+3))$$

The difference between 289 and 387 operators, or between 289 and 514 operators, to support the business over the peak hour is substantial and just the type of information the decision maker is looking for to assess the ROI (Return On Investment) for a network or server upgrade. This, admittedly exaggerated, example shows how the Response Time Pipe provides estimated response time information in a way that can be used effectively at the business decision level.

6. Conclusion

The traditional view of application measurement is evolving because of the desire to understand the impact that application response time has on the overall business. Applications designed to exploit a client/server architecture greatly increase the complexity of both the computer system configurations and the applications themselves. Measurement of these applications is very complex and must focus on the business result to avoid massive data collection and analysis problems.

The Response Time Pipe is a series of response time measurements, represented as sections of a pipe, connected together to provide an overall business analysis. Each section is related to the application by the way its measurements affect the overall response time. Because unique formulae are developed for each section, there are no requirements that the same measurement technique be used for every section. Each section can use different metrics and techniques as long as the results for each section describe its impact on the business transaction in the same units as the business transaction objective.

How measurements are connected is more important than where the application is measured, which is more important than how the measurement is implemented. This top down approach starts with the business need to determine what type of information is required and then implements the measurement technique that both provides that type of information and fits with the application. If the business need is to understand the number of call center operators to hire, then there must be a section of the Response Time Pipe for each of the required activities. If the business need is to understand the server capacity required to meet the planned call volume, then each of the server types used must be shown as an RTP section while all of the network can be a single section. In each case, once the business need has been identified, it becomes much easier to determine where, and then how, to measure the application and to create a Response Time Pipe showing the relationships between those measurements.

7. References

Knight, Alan and Jonathon Haworth. 1996. Self Instrumenation - A Discussion of Requirements and Approaches. In UKCMG, Proceedings: Computer Measurement Group.

Lipovich, G. Jay. 1997. Fixing Capacity Planning's Achilles Heel: An Approach to Managing Forecast Inaccuracy. In Computer Measurement Group, Proceedings.

McBride, Doug. 1997. Performance Management of the Desktop Client Fact or Fantasy? In Computer Measurement Group, Proceedings.

Norton, Tim R. 1998. Don't Predict Applications When You Should Model the Business. In Computer Measurement Group, Proceedings:922-933. Anaheim, CA: CMG, Inc.

Norton, Tim R. 1999. End-To-End Response Time: Where to Measure? In Computer Measurement Group, Proceedings:322-330. Reno, NV: CMG, Inc.

Ramanathan, Srinivas and Edward H. Perry. 1999. The Value of a Systematic Approach to Measurement and Analysis: An ISP Case Study. In SIGMETRICS, Proceedings V24 N1:232-233. Atlanta, Georgia: ACM.

Smead, Steven. 1998. Service Level Instrumentation 101 - An in depth look at how to instrument end user transactions. In Computer Measurement Group, Proceedings.

Smith, Connie U. and Lloyd G. Williams. 1998. Performance Engineering Models of CORBA-based Distributed-Object Systems. In Computer Measurement Group, Proceedings.

Thompson, George I., Javier Muñoz, and James K. DeBruhl. 1997. The Availability & Quality of SAP R/3 Workload Data For Performance / Capacity Management Process Requirements. In Computer Measurement Group, Proceedings.

Tsykin, Mike and Christopher D. Langshaw. 1999. End-to-End Response Time And Beyond: Direct Measurement of Service Levels. Computer Measurement Group Transactions (95): 41-48.