Principled Technologies

White Paper
January 2005

Effects of Hyper-Threading Technology on Response Time in Basic Collaboration Scenarios

For Intel Corporation

Executive summary

Intel® Corporation (Intel) commissioned Principled Technologies^{SM} (PT) to investigate the effects of Hyper-Threading (HT) Technology on response time—the speed with which applications respond to user commands—while using basic collaboration tools, such as email and Microsoft SharePoint sites, and enabling technologies, such as XML, to work as part of a team on a project. Because HT Technology can run two instruction streams simultaneously, it can improve the response time of systems when executing more than one application at a time, including collaboration applications. We looked at how well HT Technology fulfilled that possibility for teams collaborating online by running a multi-part workflow scenario much like the type of work members of enterprise teams commonly perform.

We used a variety of common enterprise business applications in our testing and the resulting workflow test scenario. Enterprise applications we used in this study include the following:

- SharePoint Portal Server 2003
- Windows Messenger 5.0
- Internet Explorer 6.0
- Microsoft Access 2003
- Microsoft Excel 2003
- Microsoft Word 2003
- Symantec Norton AntiVirus 2004

We used XML as a data-exchange medium, because XML is a growing standard for the online aggregation and dissemination of information.

We found that HT Technology frequently improved the response time of PCs in the collaboration workflow we examined.

Key findings

- Hyper-Threading Technology improved user response time in all four of the sample basic collaboration multitasking scenarios we present here.
- The response-time improvement from HT Technology occurred with little or no penalty to the completion time of the background tasks.
- With one of our test background tasks, HT Technology increased overall system throughput.
- The response-time improvement from HT Technology in these multitasking situations was generally much greater than the throughput improvement.

Before beginning our testing, we discussed collaboration issues and application usage with IT managers and looked at other sources of information on those topics. We then examined common multitasking collaboration actions. We created a workflow scenario in which the user was performing a series of foreground tasks at the same time some background tasks were executing. All of the tasks, both foreground and background, were related to the overall collaboration functions the user was trying to accomplish.

We generally found that HT Technology significantly improved the response time of the foreground applications. This response-time improvement usually occurred with little increase in the time for the background tasks to complete. In one of our two scripted background tests the background task performance even improved. Finally, we found that HT Technology consequently generally improved overall system throughput, the amount of work the system can accomplish in a given amount of time.

To ensure that our tests did not rely on characteristics specific to one vendor or system, we tested on PCs from three different leading computer vendors. We wanted to conduct our testing on current system technologies enterprise IT managers were evaluating, so we specified the following configuration:

- Intel® Pentium® 4 Processor 550 Supporting HT Technology
  - 1MB L2 cache, 3.40 GHz, 800 MHz FSB
- Intel® 915G Express chipset with Intel® GMA 900 graphics
- 512MB of DDR-400 RAM
- 80GB Serial ATA (SATA) hard disk
- Windows XP Professional with SP2

We purchased all three test systems from the vendor Web sites using their enterprise class system portals. Because different vendors usually configure similar systems slightly differently, our test systems have similar configurations but are not identical. In light of those differences, and because our goal is to focus on HT Technology, not to compare system performance, we refer to these PCs as System A, System B, and System C. Appendix A provides configuration details for each of these systems.

We found that HT Technology improved response time on all systems on multiple tests of our collaboration workflow scenario employing multiple applications.

We tested a collaboration workflow included tasks and applications that our research and experience indicated were akin to those real enterprise users might encounter. After we selected the workflow and its foreground and background tasks that we felt were representative and appropriate for this paper, we manually tested and timed them by launching combinations of foreground and background application tasks. Specifically, we timed four foreground tasks and two background tasks; the first three foreground tasks were running concurrently with the first background task, while the user executed the fourth foreground task at the same time as the second background task.

To test our workflow scenario repeatedly and accurately, we automated the hand-timed application functions using scripts. Of course, such scripts do add a small amount of overhead to the application functions and are not exact duplications of the hand-timed scenarios. They provide, however, reasonable and repeatable mechanisms for accurately running the workflow multiple times on multiple systems. We created the scripts in IBM's Rational Visual Test 6.5 (see below for more on the scripts). We use the times from our tests using these scripts in our analysis below.

Figure 1 shows that HT Technology delivered significant improvements for all four of the foreground collaboration tasks on all three test systems. The average response-time improvement was almost 70 percent on all three systems for all four foreground collaboration tasks. (We look more closely at each foreground task, as well as the two background tasks, in the next section.)

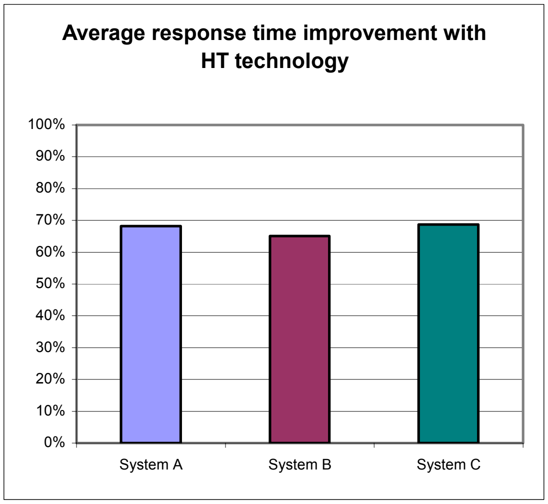


Figure 1: Average response time improvement over all four foreground tasks with HT Technology. Improvement is the percentage faster the test applications responded to user commands with HT Technology enabled than without it.

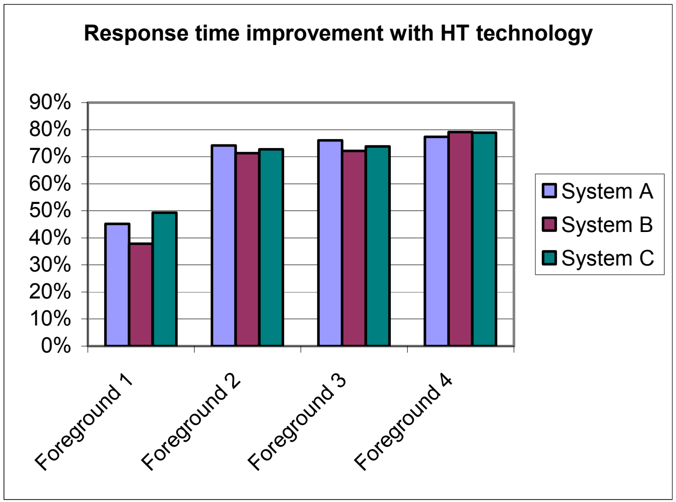


Figure 2: Response time improvement for each of the four foreground tasks with HT Technology. Improvement is the percentage faster the test applications responded to user commands with HT Technology enabled than without it.

Figure 2 shows the response time improvement percentage for the three systems on the four different foreground collaboration tasks. Although the systems exhibited different amounts of improvement for each of these tasks, HT Technology produced large response-time benefits for all four foreground tasks on all three systems. For three of the four tasks, that improvement was over 70 percent.

In the following sections we look more closely at this application workflow scenario (Application workflow scenarios) and discuss some useful basics about HT Technology (HT Technology background). We then examine the results of our tests with this workflow (Test results and analysis) and how we actually tested (Test methodology). In the appendices we provide details about the configurations of the test systems, explain how to manually execute the application functions in our workflow scenario, and discuss some issues in the development of the scripts.

Application workflow scenario

We began the process of constructing a test application workflow scenario by speaking with IT managers about basic online collaboration applications and approaches. We also did some additional informal research on enterprise team collaboration applications. We looked at the types of functions that members of teams collaborating online would commonly perform. We settled on basic applications, such as the Microsoft Office 2003 products, and collaboration tools, such as Microsoft’s SharePoint. We typically first explored multitasking application scenarios with HT Technology disabled, running them by hand as typical users would. As we performed the application functions, we monitored the processor usage of both the background and foreground tasks with the CPU Usage feature of the Windows Task Manager and the Windows Perfmon utility. We then tried those same multitasking scenarios with HT Technology enabled.

After looking at a variety of common collaboration functions, we settled on the following workflow scenario as representative of common online collaboration activities.

An enterprise has decided to gather into a single intranet resource for use by its sales and marketing teams all of the sales data it has compiled on its various competitors. To create this resource, the company creates a task force comprising one member from each of the marketing teams of each of its major product groups. Each task-force member is to gather his/her group's data, which is currently stored in many different formats, and store it in a single format, XML. XML is both ideal for online aggregation and distribution of the information and a great choice for collaboration when multiple parts of an organization (or multiple organizations) have data that they want to share but that they created originally in other formats. As one task-force member starts her day and prepares for an online meeting of the group, she has Microsoft's internal IM tool (the enterprise Windows Messenger) open on her system to communicate with her colleagues.   The company has given her the job of sharing her product groups' historical competitive data, which currently is in Access. She instructs Access to export reports summarizing that data. She exports the reports to her local drive, where she plans to review them before sharing them with the rest of the task force on the SharePoint site. This export takes time, so to be efficient she works in some other applications while waiting for her reports. She opens Internet Explorer (IE) on the SharePoint home page for the team's project so she'll be ready to add her reports to the homepage when she's reviewed them and so she can review reports other members of the task force have posted. She then opens Outlook to create a message to other task-force members. When the Access reports finish, she closes Access. She wants to study the data other members of the team have created, so she uses Excel to open one of the XML files on the SharePoint site. While the file opens, a process that takes time, she types the message to her Workgroup and sends it.

In this multitasking workflow, there are two time-consuming background tasks:

- the XML report generation in Access
- the XML file open in Excel

We timed both of these background tasks. Both our test files for these background tasks contain sales data.

The workflow contains four foreground tasks that we timed:

- the opening of the SharePoint home page in Internet Explorer
- the opening of Outlook
- the opening of a mail message in Outlook
- the saving of the mail message

For more details on how we executed this scenario and measured its sections, our specific test functions, and the files the scenarios involved, see Appendix B.

HT Technology background

Before discussing and analyzing the results of our tests, a brief overview of HT Technology is useful. User computing situations that benefit the most from HT Technology are those in which a background task appears, on a system without HT Technology or one with HT Technology disabled, to utilize all or nearly all of the processing capacity of the system. We use the term “appears” because even though performance-monitoring tools will correctly show the processor as being fully or nearly fully subscribed, in most cases multiple execution units within the processor will be idle. (Modern processors, such as the Intel Pentium 4, contain multiple execution units, the parts of the processor that actually execute program instructions. Many of the execution units in a processor are typically inactive at any given time.) By making a single processor appear to the operating system as two logical processors, HT Technology lets the operating system schedule additional work on the processor by using what it sees as the second processor. That extra work often ends up taking advantage of additional—and otherwise idle—execution units.

The effect on system throughput—the amount of work the system performs in a given unit of time—of being able to keep more execution units busy can be significant. The effect on system response time—how quickly the system responds to user commands in applications the user is currently running—in multitasking scenarios, however, has the potential to be much greater than the throughput improvement.

To see why, consider the following hypothetical example, first on a system without HT Technology. A user launches a background task, such as a report generation function, that if running alone on the system would take 50 seconds to complete. The task appears to use 100 percent of the system’s processing power for most of those 50 seconds. As users frequently do, rather than wait for the background task to complete, the user wants to keep working and so, after a wait of a few seconds, launches another application. The launch of that application if running alone on the system would take five seconds to complete, during which time it would normally use a significant amount of the system’s processing power. Without HT Technology, in such cases the operating system frequently either does not schedule the second task until the first completes or at best allocates the second task such small amounts of processor time that the second task can accomplish very little. Thus, the user effectively cannot begin to work in the second (foreground) application until the first (background) application has completed its work and the second has finally received its chance to launch. The foreground task’s response time—the time it takes before it responds to the user’s command to start—is thus over 50 seconds. We saw delays of this type in our investigation of collaboration workflow scenarios.

Consider the same example on a system with HT Technology. The user launches the same background application. This time, however, that application appears to the operating system to be using all of only one of the two (logical) processors of the system; the other processor appears to the operating system to be available. So, when, a few seconds later, the user launches the second (foreground) application, the operating system can schedule it on the second (logical) processor. The foreground application then launches almost as quickly as it would if running alone on the system. In this case, the foreground task might complete in six seconds, just a little more time than it would take if running alone. From the user’s perspective the response time of that application is six seconds—a vast improvement over the more than 50-second response time on a system without HT Technology.

Because HT Technology in this case is letting the system use otherwise idle processor execution units, it also typically would have little to no effect on the time the background task would take to complete. So, the background task would still complete in roughly the same 50 seconds it would have required if running alone.

In addition, the system's total throughput for those 50 seconds is greater with HT Technology than without it, because with HT Technology the system both finished the background task and launched the foreground application during the same time period.

During our investigation we encountered many cases in which HT Technology led to improvements on this scale. As you can see in Figure 2 above, all four of the foreground tasks we timed show very large response-time improvements.

Finally, we must, of course, note that HT Technology does not always improve response time. When each of the application tasks running concurrently does not fully or nearly fully subscribe the processor, HT Technology may not improve response time.

Test results and analysis

As we noted earlier, we automated our multi-step workflow scenario with Visual Test 6.5 scripts that both perform and time the background and foreground application functions. In this section we examine the results of tests with the scripts of each of the foreground tasks and discuss the effects of HT Technology on the response time and other relevant aspects of the foreground and background tasks.

Foreground Task 1

Figure 3 shows the results of our tests of our first foreground task. We show with it the times for the first background task. (For ease of reference, we will repeat this background's times in the charts with the times for each of the other two foreground tasks that were running at the same time as this background.) Each of the times we present in this table and in the corresponding tables for the other foreground tasks is the median of five runs, in seconds. We show three decimal places of timing results because the timers in our Visual Test scripts report in milliseconds. We're measuring response times, so smaller times are better.

Test PC	**Foreground (Open SharePoint Home Page in IE)**		**Background (Generate XML Report in Access)**	
	HT On	**HT Off**	**HT On**	**HT Off**
System A	9.687	17.69	84.940	88.250
System B	5.922	9.532	85.610	81.770
System C	8.734	17.220	84.700	81.530

Figure 3: Median test times for scripts of the first foreground task and the first background task on all three test systems. Smaller is better.

As these results (and Figure 2 above) show, HT Technology provides a significant improvement in foreground application response time on both test systems. HT technology's effect on the background task varied across the systems. On System A, the background task completed over three seconds faster with HT Technology than without, while on the other two systems the background task was about the same amount faster without HT technology.

To see why HT Technology improves response time in this case, examine Figures 4 and 5 below. Each of these figures shows the processor utilization of one of the two tasks running alone on a system without HT Technology enabled. (For these processor utilization measurements and for those in the rest of this paper, we used System A. The task times in these processor utilization charts will not match the times in the test results for two reasons: we captured the processor utilization of each task running alone with HT Technology disabled, and we did not stop gathering processor utilization data at exactly the same points as the scripts considered the tasks complete.)

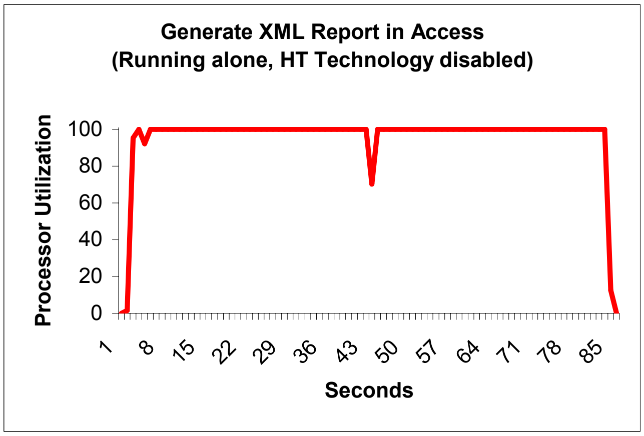


Figure 4: System processor utilization, without HT Technology, of the first background task (generating an XML report in Access) running alone.

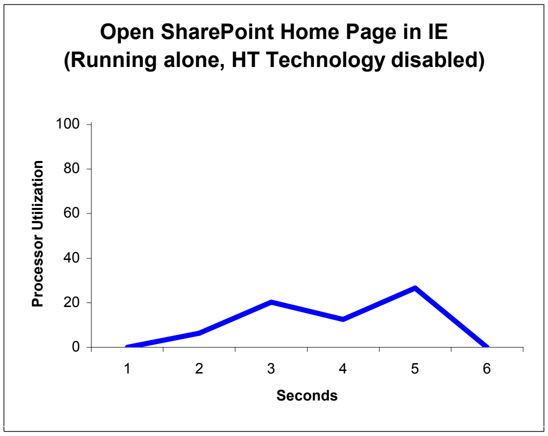


Figure 5: System processor utilization, without HT Technology, of the first foreground task (opening a SharePoint home page in IE) running alone.

As Figure 4 demonstrates, the background task—generating the competitive data XML report in Access—is so computationally intense that it uses 100 percent of the available processing power for the vast majority of the time it is running. Consequently, without HT Technology when the background task is executing and the operating system wants to schedule the foreground task, little processor time is available. Though the foreground task—opening the SharePoint home page in Internet Explorer—is not particularly demanding of the CPU, it nonetheless needs some processing resources, resources it does not easily get on a system without HT Technology. With HT Technology, by contrast, the operating system appears to have a second (logical) processor available to run the foreground task. By running the foreground task on this second logical processor, the system is able to take advantage of the idle execution units in the processor and run the two tasks concurrently. As the results in Figure 3 show, HT Technology delivers over eight seconds of response-time improvement on two of the test systems, and almost four seconds on the third.

Foreground Task 2

The second foreground task, as Figure 6 shows, is another case in which HT Technology a response-time improvement, in this case a very large one. The response-time win for this launch of Outlook averages about 14 seconds on Systems A and B, and is 13 seconds on System C—a very large improvement that any a user would greatly appreciate.

Test PC	**Foreground (Open Outlook to the Inbox)**		**Background (Generate XML Report in Access)**	
	HT On	**HT Off**	**HT On**	**HT Off**
System A	4.938	19.090	84.940	88.250
System B	5.797	20.170	85.610	81.770
System C	4.860	17.830	84.700	81.530

Figure 6: Median test times for scripts of the second foreground task and the first background task on all three test systems. Smaller is better.

The key to understanding why HT Technology was able to so greatly improve response time again stems from the processor utilization characteristics of the background and foreground application tasks. Figure 7 shows the percentage of processor utilization of this second foreground task; refer to Figure 4 for the background task.

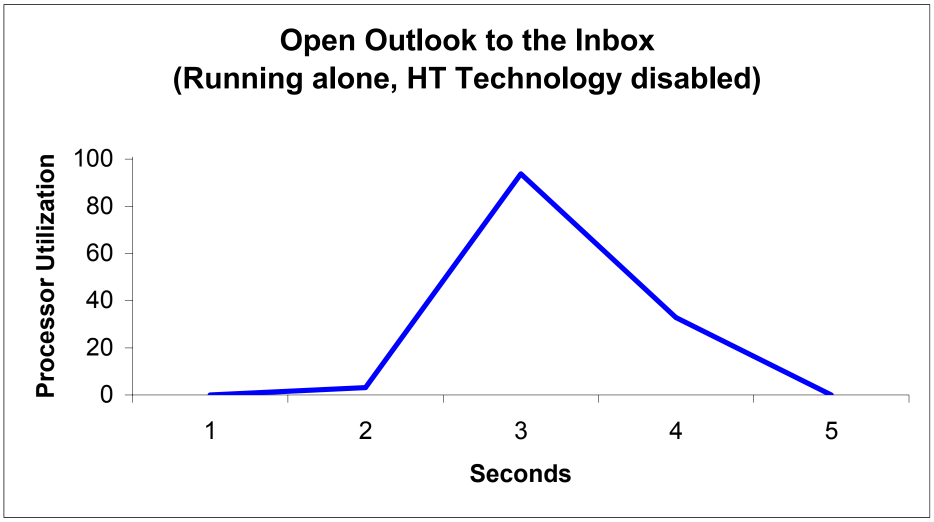


Figure 7: System processor utilization, without HT Technology, of the second foreground task (opening Outlook to the Inbox) running alone.

As you can see, though in the beginning this foreground task does not consume much of the processor, there is a period of a few seconds during which its processor utilization grows rapidly, to a momentary spike at 100 percent. Without HT Technology, the background task (Figure 4) is so demanding that the foreground task ends up getting very little processor time. With HT Technology, however, the system is able to find some idle execution units to service the foreground task during the relatively short time it needs significant processor resources. The result was, as our tests show, dramatically better response time

Foreground Task 3

The third foreground task was one that seems simple and easy: Creating a new, empty email message. Figure 8 shows the median results of our Visual Test scripts of that task (and, again, of the first background task that was running at the same time). As you can see, with HT Technology this simple task occurs quickly, in about 1.5 seconds on Systems A and B, and in just under two seconds on System C. Without HT Technology, however, this task takes an average of over six seconds, long enough to be annoying to many users.

Test PC	Foreground (Open a New Message in Outlook)		Background (Generate XML Report in Access)	
	HT On	HT Off	HT On	HT On
System A	1.578	6.594	84.940	88.250
System B	1.516	5.453	85.610	81.770
System C	1.906	7.266	84.700	81.530

Figure 8: Median test times for scripts of the third foreground task and the first background task on all three test systems. Smaller is better.

Figure 9 shows the processor utilization of the first four seconds of this task running alone. As you can see, it does not consume significant processor resources except for a momentary spike to 40 percent.

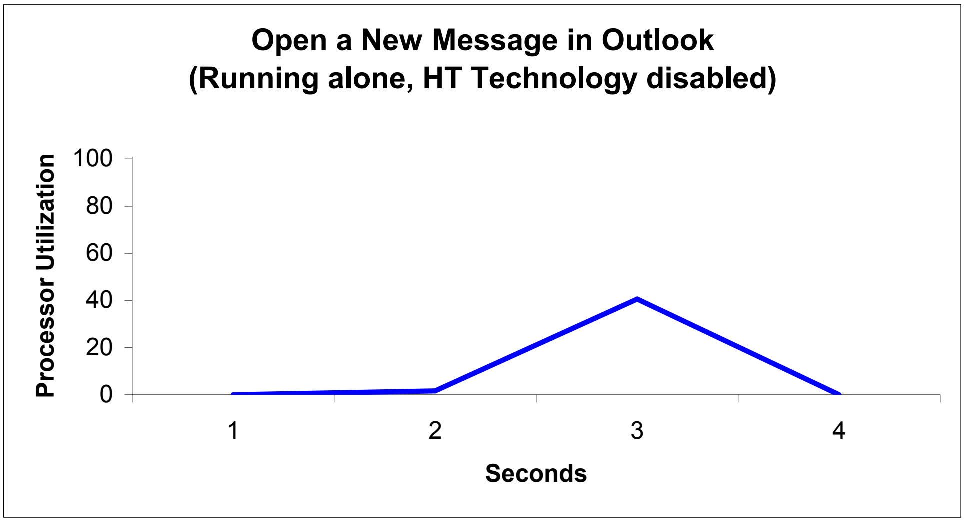


Figure 9: System processor utilization, without HT Technology, of the third foreground task (opening a new message in Outlook) running alone.

Despite this relatively low processor demand, this task is difficult for the systems without HT Technology to run in the face of the ongoing, very demanding background task. With HT Technology, however, this foreground task quickly gets the processor resources it needs and the message opens in under two seconds.

Foreground Task 4

Figure 10 shows the results of the tests of our fourth foreground task (saving an email message in Outlook), which was running at the same time as our second background task (opening an XML file in Excel). As you can see, this task took under two seconds to complete with HT Technology on each of the three test systems, while without HT Technology it required over eight seconds. In addition, the background task finished about a half a second faster with HT Technology, so both response time and overall throughput improved with HT Technology.

Test PC	Foreground (Save an Outlook Message)		Background (Open XML File in Excel)	
	HT On	HT Off	HT On	HT Off
System A	1.891	8.328	59.360	59.770
System B	1.703	8.172	58.770	59.330
System C	1.735	8.235	59.670	60.090

Figure 10: Median test times for scripts of the fourth foreground task and the second background task on all three test systems. Smaller is better.

Figures 11 and 12 below show the processor utilization of these background and foreground tasks, respectively. As you can see in Figure 12, the foreground task is extremely undemanding, with processor utilization never touching 10 percent. The background task, however, is so demanding, with an almost constant processor utilization of 100%, that without HT Technology it leaves little of the processor available for the foreground task.

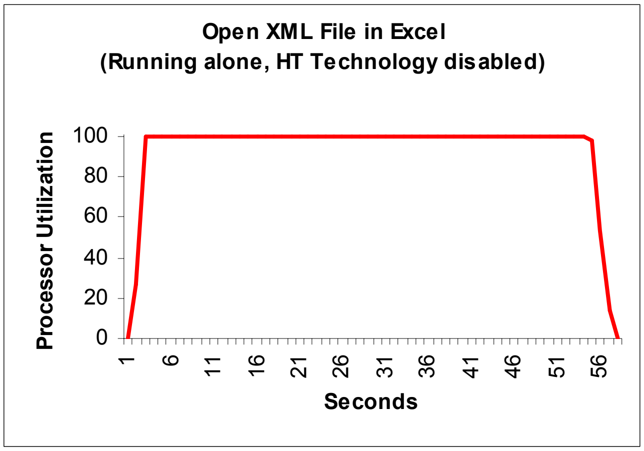


Figure 11: System processor utilization, without HT Technology, of the second background task (opening an XML file in Excel) running alone.

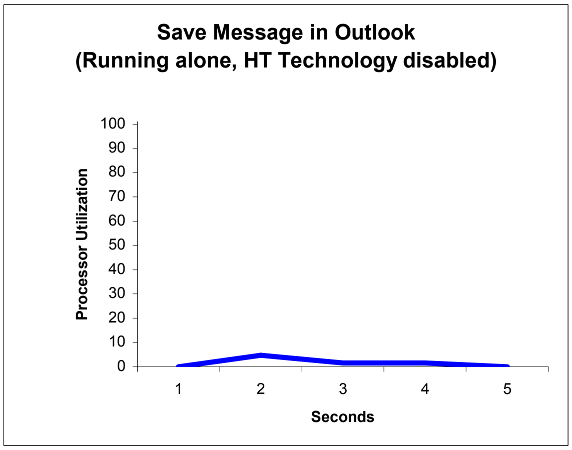


Figure 12: System processor utilization, without HT Technology, of the fourth foreground task (saving an Outlook mail message) running alone.

With HT Technology, however, the system is able to find idle processor execution units and quickly perform the foreground task. The result is not only a dramatic response-time improvement but also better overall throughput.

An obvious corollary to our results and analysis is that HT Technology is unlikely to deliver response-time improvements in multitasking situations in which neither the foreground or background tasks place heavy demands on the processor. Though some multitasking scenarios obviously are not computationally intensive, others do involve such application tasks. Users working with collaboration technologies that involve computationally intensive tasks are likely, our investigation and results suggest, to reap significant response-time improvements from HT Technology.

Test methodology

We measured the effects of HT Technology on the foreground and background tasks in our workflow scenario (see “Application workflow scenario”) both by hand and with automated test scripts. Appendix B details the steps we followed when we hand-timed the scenarios. We developed the test scripts using IBM’s Visual Test 6.5. In this paper, we concentrate most of our discussions on the results of the automated test scripts, because those results are generally more repeatable than hand timings. We collected results for five runs of each script with HT Technology enabled and five with HT Technology disabled on each of the test systems.

We set up the test systems as follows:

- We completed the manufacturer’s Windows XP setup and initialization process.
- We installed a purchased copy of Office 2003.
- Using the standard Windows Update and Office Update Web sites, we applied all current Windows XP and Office 2003 updates, including Windows XP SP2, except those updates, such as a new version of Windows Moviemaker and Windows Messenger 4.7, that were totally unrelated to the goal of this paper.
- To reflect the emphasis of our focus on enterprise IT users, we then acted as an enterprise IT department might and removed from both test systems all the applications that were unique to a particular system.
- Both to reflect this enterprise focus and to speed system boot times for testing, we removed all non-essential startup applications using the standard Windows MSCONFIG tool.
- To ensure as consistent a starting point as possible for our performance measurements, we defragmented the disk of each system.
- Using Symantec’s Ghost utility, we made an image of each system’s hard disk so we could return to that clean image whenever necessary in our testing.

To obtain each test result we cite here, we followed the same basic process:

1. Reboot the system.
2. Run the test script (or hand-timed application functions, as appropriate).
3. Record the result.
4. Repeat this process five times. (If any test or script failed, we discarded that test’s results and ran the test again).
5. Take the median result of the five iterations as the representative of the group.

We consequently discuss here only the median results of five runs of any system state (i.e., with or without HT Technology enabled).

We also computed the median performance improvement HT Technology delivered. We computed this by comparing the median time for a script (or hand-timed application function) with HT Technology enabled to the median result with HT Technology disabled. To reflect the results of all three systems, we sometimes note the average HT Technology benefit; we computed this by taking the arithmetic average of the median HT Technology benefits on each of the systems.

Appendix A: System configurations

In this appendix, we list the important configuration information for the three test systems. We purchased these systems in these configurations from the enterprise sections of the Web sites of their respective vendors.

Our goal was to have a set of test systems that were as similar as their manufacturers' standard configurations allowed. We chose the following specifications to represent commonly available and modern technology that enterprise IT managers are evaluating:

- Intel® Pentium® 4 Processor with HT Technology 3.40 GHz
- Intel® 915G Express chipset with Intel® GMA 900 graphics
- 512MB of RAM
- 80GB Serial ATA (SATA) hard disk
- Windows XP Professional with SP2

Each vendor offered the options of having Norton AntiVirus and Microsoft Office 2003 Professional Edition pre-installed on the enterprise systems, and we chose those options. Because IT organizations often employ standard configurations and to make the systems as comparable as reasonably possible, we manually uninstalled from each system any vendor-supplied applications we did not need for our testing. We used Microsoft's MSCONFIG utility to remove all unnecessary startup tasks and to make sure both systems had the same startup tasks. We applied all Windows updates, including SP2, available as of November 1, 2004, except those unrelated to enterprise systems, such as Movie Maker 2. We then defragmented the disk of each system and saved the resulting disk image. We performed all tests on these "cleaned" disk images.

We used the BIOS of each system to turn HT Technology on and off as needed for the testing.

Every vendor typically configures similar systems slightly differently. Consequently, our test systems have similar but not identical configurations. Because of those differences and because our goal is to focus on HT Technology, not to compare individual system performance, we refer throughout this paper to the systems as System A, System B, and System C.

System A	
Vendor	Dell Computer Corp.
Model	Optiplex GX280
Processor	Intel® Pentium® 4 Processor with HT Technology 3.40 GHz
L2 cache	1MB
Front bus speed	800-MHz
Motherboard	Dell 0Y5638
Chipset	Intel® 82915G
BIOS	A02
RAM	512MB of PC3200 (400-MHz) RAM in slots 1 and 2
Hard disk	Maxtor 6Y080M0 (7,200 rpm) 80,000MB
Hard disk buffer	8MB
Disk controller	Intel® 82801 FB Ultra ATA Storage Controller
Network adapter	Broadcom NetXtreme 57xx Gigabit Ethernet Controller
Graphics chipset	Intel® 82915G
Graphics RAM	128MB
Graphics driver	6.14.10.3889
Test graphics resolution	1024 X 768
Test color depth	32-bit
Sound chipset	SoundMax Integrated Digital Audio
CD-ROM drive	Samsung SC-148A
Diskette drive	NEC FD1231M
Operating system	Microsoft Windows XP Professional Build 2600 Service Pack 2

System B	
Vendor	HP
Model	Dc7100 CMT (DX438AV)
Processor	Intel® Pentium® 4 Processor with HT Technology 3.40 GHz
L2 cache	1MB
Front bus speed	800-MHz
Motherboard	HP 0968h
Chipset	Intel® 82915G
BIOS	786C1 v.01.05
RAM	512MB of PC3200 (400-MHz) RAM in slots 1 and 2
Hard disk	Seagate ST380013AS (7,200 rpm) 80,000MB
Hard disk buffer	8MB
Disk controller	Intel® 82801 FB Ultra ATA Storage Controller
Network adapter	Broadcom NetXtreme Gigabit Ethernet Controller
Graphics chipset	Intel® 82915G
Graphics RAM	128MB
Graphics driver	6.14.10.3889
Test graphics resolution	1024 X 768
Test color depth	32-bit
Sound chipset	SoundMax Integrated Digital Audio
CD-ROM drive	MSI MS-8148C1
Diskette drive	Mitsumi D353M3D-5055
Operating system	Microsoft Windows XP Professional Build 2600 Service Pack 2

System C	
Vendor	IBM
Model	ThinkCentre 842243U (A51P)
Processor	Intel® Pentium® 4 Processor with HT Technology 3.40 GHz
L2 cache	1MB
Front bus speed	800-MHz
Motherboard	IBM 915G
Chipset	Intel® 82915G
BIOS	IBM 2BKT31AUS
RAM	512MB of PC3200 (400-MHz) RAM in slots 1 and 2
Hard disk	Seagate ST340014AS (7,200 rpm) 40,000MB
Hard disk buffer	8MB
Disk controller	Intel® 82801 FB Ultra ATA Storage Controller
Network adapter	Broadcom NetXtreme Gigabit Ethernet Controller
Graphics chipset	Intel® 82915G
Graphics RAM	128MB
Graphics driver	6.14.10.3889
Test graphics resolution	1024 X 768
Test color depth	32-bit
Sound chipset	SoundMax Integrated Digital Audio
CD-ROM drive	LG GCR-8482B
Diskette drive	Sony MPF920
Operating system	Microsoft Windows XP Professional Build 2600 Service Pack 2

Appendix B: Instructions for running the application scenarios

This appendix explains how we manually tested the workflow application scenario and how we manually measured the response times of its foreground applications. Though the vast majority of our discussions here focus on the results of the automated tests, we felt it was also important to verify that manually performing the same functions showed the same type of results as those the scripts produced.

As the instructions below reflect, to get the most consistent possible timings and to make our hand-timed actions more like the ones the automated scripts perform, we sometimes chose to follow procedures for launching applications that were different from those typical users would follow. (See Appendix C for additional information on scripting issues.) For example, we typically launch foreground applications from the Run prompt rather than by double-clicking on a desktop icon. When we made such choices we also independently verified that the typical user procedures would still show similar results with HT technology enabled and disabled.

We consequently are confident that the benefits HT technology demonstrated in the response time of these foreground tasks are ones users would realize in real work situations and are not artifacts of the measurement or scripting technology.

We ran all tests five times with HT enabled and five times with HT disabled.

Our workflow scenario uses five files:

- CompetitiveData.mdb, a 24.5MB Access database the first background task uses. This file is a sales database that exports reports to XML when you run the ExportXML macro.
- Nov.xml, Nov.htm, Nov.xsd, and Nov.xsl. These four files comprise a 4.32MB XML file (which we originally created from data in the Competitivedata.mdb database) and the accompanying schema and presentation files.

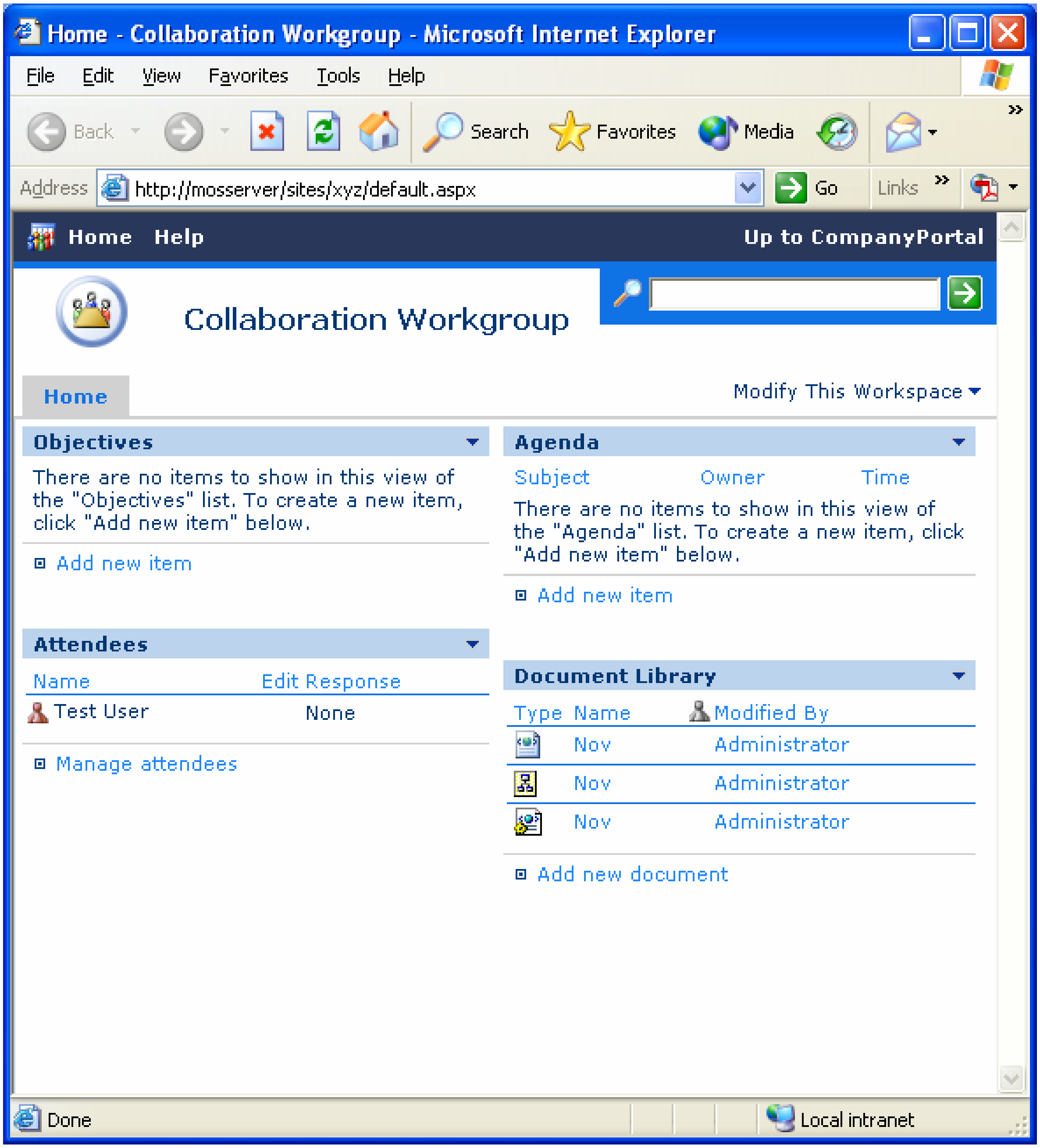
In the course of performing the actions in this workflow, we use seven applications:

- Internet Explorer 6.0
- Microsoft Access 2003
- Microsoft Excel 2003
- Microsoft Word 2003
- SharePoint Portal Server 2003
- Symantec Norton AntiVirus 2004
- Windows Messenger 5.0

We used the following process in our manual tests of the workflow scenario.

First, we set up the test systems to be able to run the workflow by doing the following steps, which are necessary only once:

- Set up the SharePoint site (name it XYZ) and copy the files in C:\Whitepaper3\SC1\Content\Document Library\1 to the site.
- Pull down the Modify This Workspace option on the site's home page and choose Site Settings. Select 'Change Site Title and Description' and give the site the title 'Collaboration Workgroup'. Go back to site settings and choose 'Apply theme to site'. Select the 'Ice' theme. The resulting site should look like the image below.


Home - Collaboration Workgroup - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://mosserver/sites/xyz/default.aspx
Go
Links
Home Help
Up to CompanyPortal
Collaboration Workgroup
Home
Modify This Workspace
Done
Local intranet

Objectives

There are no items to show in this view of the "Objectives" list. To create a new item, click "Add new item" below.

Add new item

Agenda

Subject	Owner	Time

There are no items to show in this view of the "Agenda" list. To create a new item, click "Add new item" below.

Add new item

Attendees

Name	Edit Response
Test User	None

Manage attendees

Document Library

Type	Name	Modified By
	Nov	Administrator
	Nov	Administrator
	Nov	Administrator

Add new document

- Install the Visual Test runtime.exe in Whitepaper 2 scripts\runtime.
- Make sure that Microsoft Office and Norton Antivirus 2004 are installed. (Both were present, as we discussed earlier, on our test systems.)
- Open each of the following applications, put it in windowed (not full-screen) mode, and exit it.
    - Internet Explorer
    - Access
    - Excel
    - Word
- As a real user would typically do, leave Norton AntiVirus autoprotect enabled.
- In Norton AntiVirus, set malicious script detection off. (This is necessary to let all elements of our test scripts run.)

- In Windows Explorer, map drive X: to //mosserver/sites/xyz/document library/1. Then, enter a password and user name, and check remember password
- Open Internet Explorer. Go to tools/Internet options and select Use Blank for your Home Page.
- Open Outlook and do all of the following:
    - Go to the Tools/Options menu. Go to the Other tab in the Options dialog. Click on the Advanced Options button. Check the 'Startup in this folder' setting. If Inbox is not the startup folder, click on the browse button and select 'Inbox' from the list of folders, then press OK to close the 'Select Folder' dialog and again to close the 'Advanced Options' dialog.
    - While still in Outlook's Tools/Options dialog, go to the Preferences tab. Click on E-Mail Options. In the E-mail options dialog, click on Advanced E-mail options. Uncheck Automatic name checking.
    - Turn off AutoArchive in Outlook: On the Outlook tools menu, click Options, and then click the Other tab. Click AutoArchive. Clear the Run AutoArchive every n days check box.
    - Go to Tools->Customize and select always show full menus.
    - Go to Tools->Options->Mail setup and Uncheck send immediately. Select HTML for the Mail format. Check Use Microsoft Office 2003 to edit messages.
    - Create a test-only, non-working POP3 email account in Outlook with the following parameters (the exact parameters should not matter, because Outlook should never actually try to connect):
        - Your name: Test
        - Email address: test20@mos.com
        - Incoming: pop.mos.com
        - Outgoing: smtp.mos.com
        - Username: test20@mos.com
        - Password: intel.
    - Set Outlook's send/receive settings to disable scheduled send/receive.
    - Clear all items from the email inbox.
    - Close Outlook.
- Open Norton AntiVirus. Go to Options. Click on Miscellaneous on the left side of the dialog. Uncheck "Alert me if my virus protection is out of date."
- Create a shortcut to the SharePoint homepage you created.
- Create a desktop shortcut to the CompetitiveData.mdb Microsoft Access file.
- Use Windows Explorer to make Excel own the file association for XML.

To run the workflow and time the four foreground tasks and two background tasks, we then did the following:

1. Reboot the system.
2. Wait 10 seconds.
3. To start the first background task, click on the shortcut to CompetitiveData.mdd. When Access opens the file, start the ExportXML macro.
4. Wait seven seconds, an amount a user might wait before deciding to move to the next task. Start the timer for the first foreground task and click on the shortcut to the SharePoint homepage you created. Stop the timer when Internet Explorer has fully rendered that page.
5. Wait three seconds to mimic the time the user might take to decide what to do next. Start the timer for the second background task and click on the shortcut to your email Inbox. Stop the timer when the inbox displays and shows 0 items in the status box.
6. Again wait three seconds for the user to decide what to do next. Start the timer for the third background task and click on New to create a new email message. Stop the timer when the message displays and you can type in it.
7. At this point the Access background task should finish. Access is done when it displays the Export Complete status bar message.
8. Close Access.
9. The workflow test script edits the email message as part of the timed foreground task. To allow for varied human typing speeds, edit that message now. Address the message to 'test@test.com'.

Give it a Subject of “Reminder to team” and text of “Remember to copy all your competitive data files to the SharePoint site before the next meeting.”

10. Start the second (Excel) background task by clicking on the Nov.xml link on the SharePoint homepage you opened in Step 4. When a dialog appears asking you about proceeding, click OK.
11. Wait five seconds, again to mimic the time a user might wait to see how if the background task will take long enough that she should move to another task. Start the timer for the fourth (and last) foreground task. Click on the email message and press Send. Stop the timer when the message window disappears.
12. The Excel background task is done when the file displays in Excel.

Appendix C: Issues in script development

To the best of our knowledge, IBM's Visual Test 6.5 is the tool most widely used today for constructing application-based benchmarks and performance tests for PCs running various versions of Microsoft Windows. We have used this product (and previous versions of it) for many years to build performance tests. The tool does, however, have some stated limitations that unavoidably affect the way you develop performance tests with it.

First, the product's own documentation notes that its primary goal is to be a tool for automating application testing, not a benchmark development system. Consequently, the granularity of some of its functions and the way some of its functions behave are not ideal for benchmark development.

IBM also does not officially support Visual Test 6.5 for the Windows XP operating system. Because Windows XP is the leading and most current desktop version of Windows today, we nonetheless felt it was essential to use that operating system in our tests.

The presence of any scripting tool has the potential to affect the performance of a system. The tool unavoidably must, for example, occupy some memory and consume some processing power. Consequently, developing a performance-measurement script with such a tool involves maintaining a delicate balance between using the tool to automate typical real user behavior and minimizing the effects of the tool on system performance.

The combination of these limitations means that in some cases functions that are simple for a human to perform are either not possible or not appropriate for a Visual Test script whose goal is to accurately measure performance. For example, though a user can easily launch any of the background applications we used in the test scenarios and then double-click on a desktop icon of a data file (which a foreground task would then launch), a Visual Test script might not be able to do the same, especially on a system without HT Technology, until the background task terminated. The Visual Test script would instead have to wait until the operating system was able to allocate processor resources to it—a wait that would essentially serialize functions real users would perform in parallel.

To avoid these limitations and problems we sometimes had to use scripting techniques that would achieve the same results as typical user behavior but not exactly mirror that behavior. Such techniques include inserting delays to mimic user think time and launching applications with a click on the OK button of a pre-filled Run command line. The hand timing instructions we provide in Appendix B reflect those techniques, so following those instructions will yield results similar to those the scripts produce. We should note here, as we did in Appendix B, that whenever we had to use one of these alternative techniques we manually verified that doing so did not materially alter the way the system behaved and that real users performing the same actions in more typical ways would see the type of HT Technology benefits we describe.

The timings the scripts produce also inevitably contain some variability. This variability is a result of the combination of the tool's limitations and the generally asynchronous nature of the many functions Windows XP and other modern operating systems have running at any given time.

One of our goals was to produce scripts that Intel could own and distribute. To make such distribution as legally simple as possible, Intel needed to own all rights to the data files in the scripts. At the same time, we needed to be able to do all our research and analysis using data files we provided. To address both these requirements, we did all initial research and scripting for this White Paper using data files we provided. When we had settled on the workflow scenario for this paper, we gave Intel the specifications—such as size, type of content, and graphical and computational intensity—for the files we used. Intel then provided files that met those specifications. We verified that all the files Intel provided did indeed meet our specifications and satisfied ourselves that the files were appropriate for our use. We then used those files in our final test scripts and hand timings. In the course of those hand timings we further verified that the HT Technology benefits we found with these files were basically the same as those we found with our original files.

Finally, though one of the goals of this effort was to produce distributable scripts, we were not trying to build bulletproof benchmarks for wide distribution and use. We developed the scripts to mimic user behavior on our specific test systems; on significantly faster or slower systems the scripts might show different levels of HT Technology benefit or even possibly fail to work. So, though the scripts are as reliable, self-contained, and free of system dependencies as we could reasonably achieve within the project's timeframe, they do sometimes fail or encounter problems. Should a problem occur, rebooting the system and running the script again will generally yield a good result.

Principled Technologies

Principled Technologies, Inc.
4813 Emperor Blvd., Suite 100
Durham, NC 27703
info@principledtechnologies.com

Disclaimer of Warranties; Limitation of Liability:

PRINCIPLED TECHNOLOGIES, INC. HAS MADE REASONABLE EFFORTS TO ENSURE THE ACCURACY AND VALIDITY OF ITS TESTING, HOWEVER, PRINCIPLED TECHNOLOGIES, INC. SPECIFICALLY DISCLAIMS ANY WARRANTY, EXPRESSED OR IMPLIED, RELATING TO THE TEST RESULTS AND ANALYSIS, THEIR ACCURACY, COMPLETENESS OR QUALITY, INCLUDING ANY IMPLIED WARRANTY OF FITNESS FOR ANY PARTICULAR PURPOSE. ALL PERSONS OR ENTITIES RELYING ON THE RESULTS OF ANY TESTING DO SO AT THEIR OWN RISK, AND AGREE THAT PRINCIPLED TECHNOLOGIES, INC., ITS EMPLOYEES AND ITS SUBCONTRACTORS SHALL HAVE NO LIABILITY WHATSOEVER FROM ANY CLAIM OF LOSS OR DAMAGE ON ACCOUNT OF ANY ALLEGED ERROR OR DEFECT IN ANY TESTING PROCEDURE OR RESULT.

IN NO EVENT SHALL PRINCIPLED TECHNOLOGIES, INC. BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH ITS TESTING, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL PRINCIPLED TECHNOLOGIES, INC.'S LIABILITY, INCLUDING FOR DIRECT DAMAGES, EXCEED THE AMOUNTS PAID IN CONNECTION WITH PRINCIPLED TECHNOLOGIES, INC.'S TESTING. CUSTOMER'S SOLE AND EXCLUSIVE REMEDIES ARE AS SET FORTH HEREIN.