Journal of Digital Imaging

Development and Evaluation of Different Methods to Assess Download and Display Time of Image Web Systems

M. Pietsch,[1] A. Schlaefke,[2] T. J. Vogl,[3] and B. Bergh[1]

Objectives: **The aim of this study was to develop and verify different methods of measuring time-to-display (TTD) for radiological images with image web systems (IWS). The process should be automatable in order to repeatedly perform a large number of measurements without human interaction.** ***Materials and Methods:*** **Three methods were defined and compared with respect to usability, stability, and quality of results. Method 1 was based on Windows 2000 Performance Monitor, whereas method 2 employed phototransistors taped to the screen and connected to a separate PC. A software tool developed for method 3, which used Windows application programming interface (API) function, calls to read the color code assigned to specific pixels on the screen.** ***Results:*** **Method 3 proved to be the most reliable and easy to automate. The accuracy is practically equivalent to method 2, but it proved to be far more automatable. Method 1 produced the largest mean error, was easily disturbed, but was also easy to set up and provided additional insights into the system's architecture especially if combined with method 3.** ***Conclusions:*** **To measure the performance of image distribution systems, any of these methods can be used, but method 3 proved to be superior.**

KEY WORDS: Image distribution, web-based, internet technology, WWW (world wide web), PACS (picture archiving and communication systems), image compression, performance measurement

INTRODUCTION

The increasing importance of hospitalwide image distribution is a prerequisite in order to achieve cost savings and campuswide optimization in picture archiving and communication system (PACS) projects.[1–5] There is an increasing demand for accurate data on the performance and configuration of image web systems (IWS). Predominant questions are the time-to-display (TTD) of images on the desktop client and the time span between submission of an image to the IWS and its availability on the web. A few studies on this topic have been published, but a standardized approach to measure TTDs does not exist. Clark et al.[6] describe an intricate system to monitor image flow between modalities and PACS, which generates performance data as well. Most authors,[7,8] however, rely on manual measurements using stopwatches to determine the TTD of images. Although this approach is very simple to implement, it has substantial disadvantages. Because TTDs of modern systems are usually in the range of a few seconds, the reaction time of individuals has a large impact. To achieve reliable results, a measurement has to be repeated very often, but the number of achievable repetitions is limited by the endurance of the human observer.

[1]*From the Department of Information Technology and Biomedical Engineering, Universitätsklinikum Heidelberg, Tiergartenstrasse 15, 69121, Heidelberg, Germany.*

[2]*From the Department of Information and Communication Technology, Klinikum der Johann Wolfgang Goethe Universität, Frankfurt am Main, Germany.*

[3]*From the Institute for Diagnostic and Interventional Radiology, Klinikum der Johann Wolfgang Goethe Universität, Frankfurt am Main, Germany.*

Correspondence to: B. Bergh Department of Information Technology and Biomedical Engineering, Universitätsklinikum Heidelberg, Tiergartenstrasse 15, 69121, Heidelberg, Germany; tel: +49-6221-56-2000; fax: +49-6221-56-2004; e-mail: bjoern.bergh@med.uni-heidelberg.de

Copyright © 2006 by SCAR (Society for Computer Applications in Radiology)

Online publication 14 September 2006
doi: 10.1007/s10278-006-0856-5

Different methods for automatic measurements are conceivable. The ideal method would be simple, automatable, and would deliver reliable and reproducible results. For the investigation of this study three different methods were selected. All methods simulated a human user by mouse and keyboard interactions, but leveraged different methods to detect image arrival. The objectives were to develop, evaluate, and compare the three methods with respect to their usability, stability, and accuracy.

MATERIALS AND METHODS

Image Web System

As the underlying image web system (IWS) the commercially available software Exhibit (release 3.1) developed by Mitra (today part of Agfa, Mortsel, Belgium) was chosen. This package was installed on a dedicated server. A detailed description of the set-up can be found in Bergh et al.[9]

Test Objects

Three groups of image types—computed radiography (CR), computer tomography (CT), and magnetic resonance (MR) imaging—were defined, each consisting of five different examinations. The CR group comprised five chest x-ray images, all displayed in a 1×1 setting (Fig. 1a). For CT and MR, 16 images were displayed in a 4×4 setting (Fig. 1b). A total of 15 test objects, divided into three groups and containing 165, images were evaluated.

Desktop PC Client

In all tests, a desktop PC client manufactured by Fujitsu Siemens Computers with a 500 MHz Pentium III processor, 128 MB of RAM, and 10 GB hard disk drive was used. On this desktop PC, the graphics adapter (INTEL 82815, 4 MB) and the network interface card (INTEL Pro/100 VM) were delivered onboard. The selected network setting was 100 Mbit/s in full duplex mode. The PC was connected to the same Catalyst WS-C 6509 (Cisco®) backbone switch as the image distribution server. For the operating system, Windows 2000 Professional and MS Internet Explorer (MSIE) with Java Virtual Machine (both version 5.0; Microsoft Corporation, Redmond, WA, USA) were used.

Method 1: Windows Performance Counter-Based

Method 1 was based on performance counters provided by the Performance Monitor, which was delivered as part of the Windows 2000 operating system. Figure 2 shows a screenshot of this tool during an image download process. A variety of tools are available, which can log performance counter values to comma-separated values (CSV) files. For this purpose, a tool was developed in-house to achieve a high sampling rate while limiting system stress by logging only one counter. The selected counter represented the processor time used by the MSIE process. Because the Java applet displaying images runs within this process, this counter shows all CPU usage of the applet, which is high during image decompression, plus all activities of MSIE.

An automated process analyzed the generated CSV files, detected thresholds of defined counters within the CSV file, and used these to calculate the duration of specific operations. Because other factors may also cause high CPU usage, results had to be verified according to rule sets and manually confirmed.

The freeware tool Automouse (Kakuya Yamamoto, Japan) was used to automatically and repetitively log into the IWS, load the list of patients, select a study, and download the CR image or set of CT or MR images. To avoid any repercussion and to clear local caches, the IWS client applet and MSIE were restarted after each set of images. This ensured a clear separation of the measurements. A detailed list of the performed action steps is provided in Table 1.

Method 1b: Windows Performance Counter-Based, External Logging

The same approach as described above was used, but the tool creating the CSV files was executed on another computer. Performance counters can easily be accessed from a different PC over the network. This relieved the original desktop client of hard disk IO traffic, but created a network demand of about 400 kb/second.

Method 2: Phototransistors Taped to the Screen

In method 2, three phototransistors were taped on the screen of the client computer. These phototransistors measured the brightness of preselected areas of the screen (Fig. 3) to detect the arrival of images. They were connected to analog–digital converters (ADC), which themselves were connected to the parallel port of a separate PC (Fig. 4). This separate PC was executing MS DOS 6.22 and an application written in QuickBasic 4.5 (Microsoft Corporation) that exported the measured values to a file. Additionally, a relay was installed and controlled by the separate PC. The purpose of this relay was to short-circuit the left button of the client PC's mouse to simulate the mouse click starting the download process. To prepare a download by selecting a patient and an image, the tool Automouse was used as described in method 1.

The phototransistors were placed on locations of the screen, which considerably change brightness when the image is downloaded and shown. One was placed outside the actual image to detect the applet's switch to display mode. The second one was located in the center of the image, and the third was placed inside a small icon, which is part of the IWS client applet and indicates that the download process is still ongoing. This icon disappears after the image has been loaded

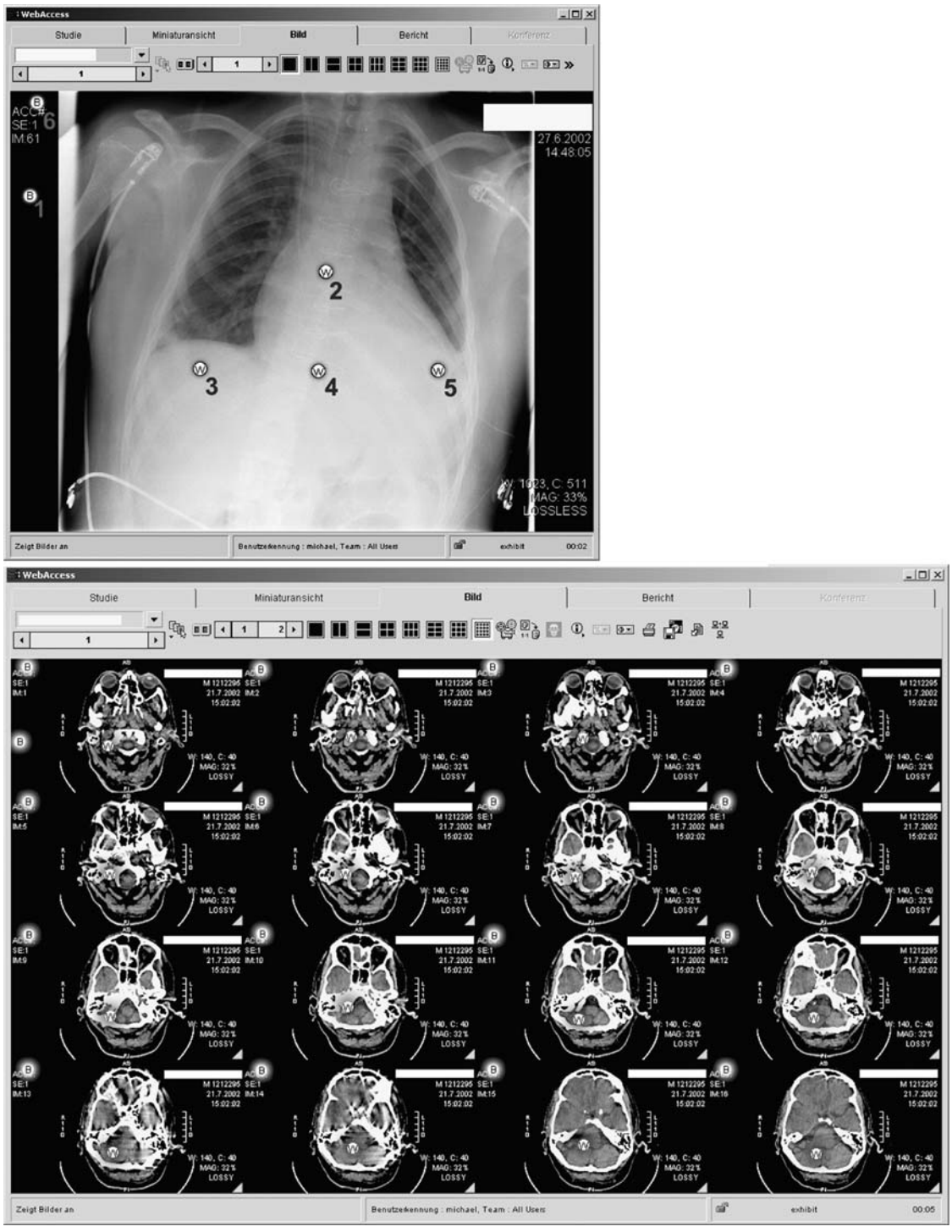

Fig 1. (a) CR images were displayed in a 1 × 1 setting. Locations of screen pixels monitored by using method 3 are indicated by dots. The target RGB value assigned to the dots marked B was "black" and to the ones marked W "brighter than dark gray." (b) For CT and MR, always 16 images were displayed in a 4 × 4 setting. The locations of screen pixels monitored by using method 3 are indicated by dots. The target RGB value assigned to the dots marked B was "black" and to the ones marked W "brighter than dark gray."

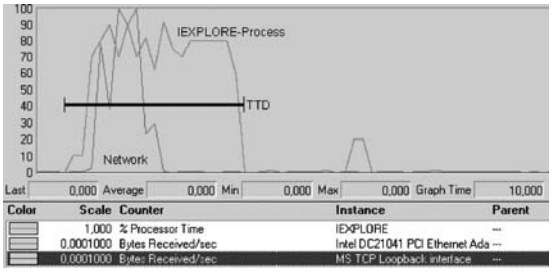

Fig 2. Method 1: curves of different performance counters during the reception of an image. The arrival of the image is represented by the network activity. During the reception and for some time afterwards, CPU usage of the IEXPLORE process remains high, indicating postprocessing and display of the image.

completely. A description of each action step can be found in Table 2.

Thirty-three phototransistors would have been necessary to perform measurements with 4 × 4 tableaus of CT and MR images (16 images + 16 icons indicating download in process + 1 indicating switch to image display mode). In this study, the measurements were therefore limited to the same five CR images used for methods 1 and 3. To clear the local cache of the IWS client, the client applet and MSIE were both restarted after each set of image downloads.

Table 1. Action steps performed by automouse during runs of method 1

0.	Start MSIE, navigate to IWS homepage, log in, request list of CRs
1.	Select first patient then wait 30 seconds
2.	Click on "get image" button then wait 30 seconds
3.	Return to list mode, select next patient, wait 30 seconds, proceed with 2
4.	After fifth image, exit MSIE
5.	Start MSIE, navigate to IWS homepage, log in, request list of CTs
6.	Select patient, switch to miniature mode, select first series then wait 30 seconds
7.	Click on "get image" button then wait 30 seconds
8.	Return to miniature view, unselect previous and select next study, proceed with 7
9.	After second tableau, exit MSIE
10.	Start MSIE, navigate to IWS homepage, log in, request list of MRs
11.	Select patient, switch to miniature mode, select first series then wait 30 seconds
12.	Click on "get image" button then wait 30 seconds
13.	Return to miniature view, unselect previous and select next study, proceed with 12
14.	After second tableau, exit MSIE
15.	Proceed with 0

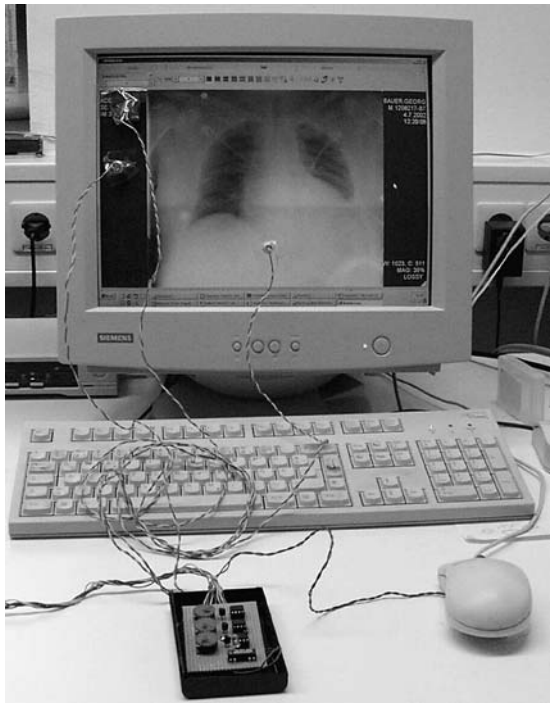

Fig 3. Method 2: position of phototransistors on the screen. The phototransistor on the left bottom detects black if the IWS client applet is in image display mode, the one in the center detects the image itself, whereas the top-left phototransistor is located above the small icon that indicates an ongoing download. In the foreground, the box containing the ADCs can be seen.

Method 3: Reading the Color Code of Screen Pixels

An application, able to identify the color code of defined screen pixels and to simulate mouse operations (esp. movement and clicks) and keyboard entries, was developed to investigate the download operations in more detail. Similar to the use of the phototransistors in method 2, the pixels were selected so that the progress of a download operation could be deducted from the change of the color code.

To automate the measurements, scripts for mouse and keyboard operations were created and stored in text files. The script creation included a choice of locations on the screen for simulated mouse clicks as well as the careful selection of to-be-monitored pixels. For all image types, only the minimal number of pixels necessary to clearly identify completion of download was defined. For CR, these were six pixels, and for CT and MR 33 pixels were necessary. For all image types, 1 pixel was located outside the image area to ensure that the Exhibit client actually switched into image display mode. In CRs, 3 more pixels were located below the diaphragm, one within the mediastinum and one within a small icon which is part of the Exhibit client and indicates that the download is not yet complete. In 4 × 4 scenarios of MR and CT tableaus, pixel location had to be chosen very carefully, because

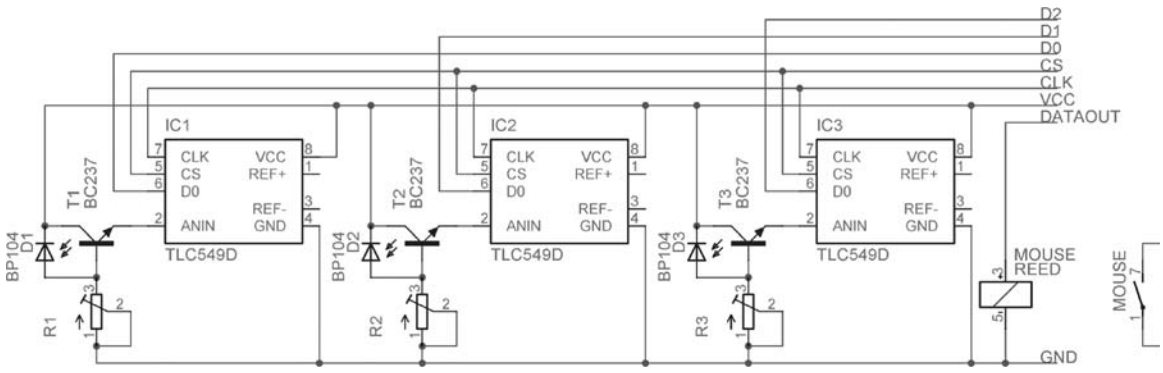

Fig 4. Method 2: wiring diagram and set-up.

cranial CTs and some MRs contain numerous black or very dark gray areas. If the monitored pixel was located in one of these areas, the arrival of the image could not be detected. One pixel was monitored within each of the 16 images, because the images did not always arrive in order. Each image contains its own icon indicating that the download is incomplete; therefore, another 16 pixels within these icons had to be monitored.

Our application used standard Windows application programming interface (API) calls to gdi32.dll to read the color of screen pixels and to simulate mouse operations and keyboard entries, including movement and clicks.

Early versions of the application used a local text file containing simple commands for automation. These commands included leftX100Y200 to simulate a left mouse click on a certain screen location and moveX500Y123 to move the mouse to a location. The scripts also contained information on the location and expected color of screen pixels to monitor, such as pixcX527Y450R150G150B150 to monitor a certain pixel and wait for it to reach a certain shade of gray. It was possible to define ranges of color pixcX527Y450R > 150G > 150B > 150. Simple repetitions (loops) were possible, and branching was not implemented. Subroutines were implemented to allow code reuse where numerous tasks had to be repeated. The most important of these cases was the definition of locations and expected colors of pixels for CT and MR.

Including some minor adjustments, all scripts were based on the same principle. Through the set of simulated mouse clicks and keystrokes, the MSIE was started and navigated to the IWS homepage. Next, the IWS client applet was loaded and a login was performed, before selecting the patient and concrete study. Afterwards, all selected screen pixels to be monitored as well as the color codes were transmitted to the client and locally stored. The local storage was done to avoid additional network traffic during the actual image download. The client application then simulated a mouse click on the "get image" icon of the IWS client applet, started the timer, and immediately began monitoring the first pixel with an interval of 1/100 second. The first selected pixel was positioned in such a way that it would turn black if the IWS client applet changed to the display mode. After having reached its expected color code, the next pixel was checked without latency. When all pixels had reached their expected color code, the timer was stopped. After completion, the next patient or study was selected. To clear the local cache of the IWS client, the client applet and MSIE were restarted after each set of image downloads, but not between different types of images.

During the tests with single desktop clients[10] the measuring application read the script files directly. For tests with multiple desktop clients,[9,11] this application was divided into a client and a server component, communicating through transmission control protocol/Internet protocol (TCP/IP). The server, located on a separate computer, read the script file and simultaneously transmitted the same commands to any number of connected clients across the network. The clients in turn transmitted the results of measurements back to the server, which stored them in the database. This was done to synchronize many clients and to create a heavy workload on the server with concurrent

Table 2. Action steps performed during runs of method 2

0. [Automouse] Start MSIE, navigate to IWS homepage, log in, request list of patients
1. [Automouse] Select patient and position mouse pointer on "get image" button then wait 60 seconds
2. [External PC] Short-circuit physical mouse button of client and start timer
3. [External PC] Wait until ADC channel 1 near black signaling switch to display mode
4. [External PC] Wait until ADC channel 2 brighter than black for at least 1 sample
5. [External PC] Wait until ADC channel 3 near black for at least 600 consecutive samples
6. [External PC] Timer stop at first sample of black period of ADC channel 3
7. [Automouse] After 60 seconds (see 1). Return to patient list, select next patient
8. [External PC] Wait until ADC channel 1 brighter than black signaling switch to patient list mode. Wait 60 seconds. Proceed with 2
9. [Automouse] After download of fifth image, exit MSIE, proceed with 0

clients performing identical operations and requesting images at the very same time. Up to 16 synchronized desktop clients created coordinated stress on the IWS.

More than 500,000 operations were timed by using method 3 alone for testing different circumstances, including evaluation of alternate desktop clients and server hardware as well as the impact of concurrent upload of images via DICOM (digital imaging and communication in medicine). The results of these measurements are reported by Bergh et al.[9–11] Methods 1 and 3 were combined during the investigation of upload capacity.[11] Performance counters were used to measure activity on the server, whereas method 3 produced client-side load by requesting images and measuring TTDs. This resulted in interesting insights into the processes on the server side; however, the discussion of which is outside the scope of this article. It can be found Bergh et al.'s work.[11]

Series of Tests

Several test runs were conducted to compare the three defined methods. During all tests, the same images were downloaded.

Using method 1, five CR images, two 4 × 4 CT tableaus, and two 4 × 4 MR tableaus were repeatedly downloaded. These tests were performed overnight, while during this time 58 test runs were performed.

Using method 2, the test accessed the same five CR images as used for method 1. A total of 84 test runs were performed in one night. For technical reasons, tableaus of CT and MR images could not be measured using this method.

Method 3 was used to download the same CR images and tableaus of CT and MR images as used for the other methods. For this study the download was repeated 120 times within one night. Three additional CT and MR series were downloaded to provide comparability with later tests conducted with multiple clients.

RESULTS

Quality of Results

Table 3 shows the results for all of the four methods. In any case, method 3 produced the smallest mean error. Depending on the image type, it was almost 2 or 3 times smaller than that of method 1 and a bit smaller than that of method 2.

Table 3. Results of different methods

	CR1	CR2	CR3	CR4	CR5	CT1	CT2	MR1	MR2
Method 1: performance counters internal logging									
Average	1.88	1.13	1.09	1.11	1.14	3.75	3.20	2.67	2.32
Min	1.70	0.92	0.88	0.92	0.95	3.05	2.76	1.74	1.70
Max	3.10	1.32	1.30	1.29	1.37	4.25	3.48	3.11	2.66
mean error	0.18	0.08	0.09	0.08	0.08	0.16	0.17	0.21	0.15
Method 1b: performance counters external logging									
Average	1.83	1.05	1.03	1.05	1.09	3.87	3.37	2.85	2.42
Min	1.01	0.79	0.75	0.79	0.57	3.34	3.10	2.39	2.21
Max	2.19	1.41	1.70	1.47	1.40	5.36	3.69	3.36	2.76
mean error	0.12	0.11	0.11	0.10	0.09	0.16	0.09	0.16	0.11
Method 2: phototransistors									
Average	1.80	1.03	1.05	1.06	1.07				
Min	1.64	0.88	0.88	0.93	0.94				
Max	1.98	1.21	1.21	1.27	1.20				
mean error	0.09	0.08	0.06	0.06	0.06				
Method 3: color of screen pixels									
Average	1.06[a]	1.02	1.03	1.04	1.06	2.84[a]	3.11	1.68[a]	2.05
Min	0.88	0.94	0.94	0.94	0.94	2.75	2.98	1.61	1.97
Max	1.18	1.10	1.12	1.15	1.20	2.90	3.23	1.73	2.12
mean error	0.07	0.05	0.04	0.06	0.07	0.04	0.07	0.03	0.04

[a]During measurements with Methods 1, 1b, and 2, Internet Explorer and the IWS client applet were restarted between downloads for technical reasons. Later tests showed that the first image-loading operation after the initial start of the applet takes considerably longer. As a result of this behavior, results for CR1, MR1, and CT1 cannot be compared between Methods 1, 1b, and 2, and Method 3. They can, however, be compared among Methods 1, 1b, and 2.

Time-to-display (TTD) of different image types as measured with different methods. Times are given in seconds.

In general, in respect to CR, methods 1, 1b, 2 and 3 produced comparable results. The mean error of method 1 was 0.1. The mean error of method 1b was slightly higher at 0.11, whereas the mean error of methods 2 and 3 was 0.07 and 0.06, respectively. With respect to CR images, it can be seen that method 1 produced between 4% and 10% slower results than method 2, whereas method 3 produced 0–3% faster results.

Method 1b, which produced faster results for CR, delivered significantly slower results for CT and MR tableaus. In both cases, method 3 produced by far the fastest results. As described earlier, no tests were conducted using method 2 during CT and MR tableau downloads.

It is important to note that we discovered during further testing that the first image download after starting the IWS client applet software took considerably longer than any later download. This may be caused by processes within the IWS. Because this substantially affects our measurements, the results for CR1, CT1, and MR1 using methods 1, 1b, and 2 cannot be compared to method 3.

Usability and Stability

All methods required manual interaction in case of the IWS applet displaying an error message. Such an error message stops the measurement process because it requires a manual intervention, e.g., click the "OK" button.

One key issue in method 1 was that periods of high CPU usage could not easily be attributed to specific operations. Automated sorting according to rules could only partially mitigate this issue. Manual sorting of the data and confirmation of results was therefore necessary.

With method 2, download times could very easily be attributed to specific images because the external PC started download operations with a mouse click.

After a substantial initial effort developing the application and download scripts, method 3 had a very stable run and failed only when the IWS client applet displayed error messages or did not successfully download an image at all. In the 4 × 4 scenarios of MR and CT tableaus, pixel location had to be carefully chosen as cranial CT and some MR images contain numerous black or very dark gray areas. If the monitored pixel was located in one of these areas, the arrival of the image could not be detected accurately. Because of this, creation of the scripts proved to be a tedious task. A manual sorting of results was not necessary as all results were orderly stored in a database.

DISCUSSION

The authors gathered some experience in manual measurements during evaluation of no-cost teleradiology systems[12] and found this to be a very tedious task producing a large mean error. This was acceptable at that time, because only a few parameters were measured and only a limited number of images were transmitted. The author's experience showed that any human observer cannot continuously produce good results for more than 10 minutes at a time. The series planned here call for the download of five CR images and two tableaus each of CT and MR images, resulting in nine measurements per round, which takes about 10 minutes to complete if performed manually. For the present investigation, manual measurements were performed at first, but not standardized and soon abandoned because it was clear that there was no possibility to archive a relevant number of rounds.

Quality of Results

All evaluated methods generated good results. The standard error is within an acceptable range for all the different methods. However, method 3 is superior in that respect. Note that IWS, operating system, and network also contribute to standard error. Within one method all results are reproducible. In comparing the different methods, certain differences in absolute values do exist, but can be explained by later findings and technical information. These differences, however, are minor with respect to the overall time and remain below 10%. If different systems are to be examined and compared, only one method should be chosen and maintained for all tests.

Method 1b cannot be recommended as it generates heavy network traffic and slows down network operations. This leads to longer TTDs for CT and MR tableaus. The fact that method 1b produced faster TTDs for CR, but slower ones for

CT and MR, tableaus is consistent with later findings.[10,11] CT and MR downloads seem to be much more susceptible to network load.

The fact that method 1 produced the longest TTDs may be explained by the fact that the IWS client applet is busy serving other processes such as preparing image downloads and downloading patient information before or after the image has completely arrived. Method 1 would be unable to detect the difference because it only records the activity of MSIE and the IWS client applet, whereas the other methods detected an image arrival as soon as it was displayed. Apart from that, the generation of counter values and the intense disk IO load of continuously writing performance data certainly has an impact on the measured performance. Method 3 did not inflict any hard disk IO during image downloads as the pixel locations to be monitored were stored in RAM.

The finding that method 3 produced slightly faster results than method 2 may be attributable to the slower refresh rate of the screen and processing within the video adapter. Image arrival can only be detected by phototransistors once the electron beam actually paints the image on the screen, whereas API calls will return color values as soon as the instruction to display the image has been sent to the video adapter through the PCI bus. Luminescence of the screen did not play a significant role as the phototransistors returned numerous low values even in bright areas.

Usability and Stability

Usability and stability affect the comparison of the methods to a high extent. Key aspects are the time and resources required to set up the tests, the number of errors and hangs produced by the system during the tests, and the effort to sort and evaluate results.

Method 1 was initially used because it was simple to set up. To start testing, only a script had to be developed using Automouse. Logging could be performed by either the Windows NT or Windows 2000 version of the Performance Monitor, or with any other tool that would allow fast logging of performance counters.

Problematic with this method is the identification that peak in CPU load is caused by which operation. Because no feedback mechanism exists between Automouse clicking locations on the screen, the image displayed on the screen, and the measuring process, this method can falsely assume that an image has arrived when in fact the download failed without an error message or proceeded so slowly that no distinct peak resulted. The only indications are the number of high CPU usages before the event, the elapsed time since the start of the series, and the duration of the event in respect to earlier events of the same type. None of these indicators is reliable, because all three vary considerably as a result of many factors. An additional manual review and confirmation of results was necessary in all cases. Long pauses helped in sorting the results, but slowed down this method considerably.

Method 2 was developed in order to completely eliminate the influence and interaction of the measuring process itself on the duration of operations on the desktop client computer. The monitoring does not affect the desktop client to be measured at all. All processes within the desktop client are independent from the measuring process. However, specialized hardware and software to control are necessary, which have to be developed or respectively purchased. The number of locations that can be monitored on the screen is limited, the locations have to have a minimum size, and have to change their brightness considerably. This method also relied on pauses between download operations to avoid desynchronization of the measurement PC, Automouse, and the IWS client applet on the desktop PC to be monitored.

A clear advantage of method 2 is that it can be employed in an environment where interaction with the computer on which measurements are to be performed is impossible. If images are manually selected instead of an Automouse script, the client computer remains totally untouched. This may be appropriate in scenarios where a highly specialized software is used and/or legal aspects prohibit changes to the system, which is quite frequent in medicine.

Method 3 eliminates the key disadvantages of the other methods. No additional hardware is required. Sorting of results is not an issue, because the measurements are performed by the same system that controls the mouse operations. All results are stored in a database, making evaluations very easy. As the system knows exactly when the first operation is completed, it can

immediately start the next download, thereby eliminating the need for long pauses. This speeds up the process considerably.

An additional advantage of method 3 is the fact that it allows synchronization of multiple clients to simulate peak load. During our tests, series with up to 16 concurrent desktop clients were conducted. This feature is necessary to create a stress level on the server in order to judge its performance. A large number of desktop clients may display images, but communication with the server requires resource usages, which only occur during download. Even in large institutions, multiple desktop clients will rarely request images at the very same time. However, during peak work hours, simultaneous requests will occur. In pretests using method 1, no slowing down caused by multiple clients could be detected if they did not request images simultaneously. Synchronization therefore is a requirement for high workload simulations.

Prospects and Future

All the methods can principally be applied to any computerized system inside as well as outside the radiology or even outside the hospital environment. All the methods also submit mouse and keyboard operations to the system initiating some task. In all the cases, either the Automouse scripts (methods 1 and 2) or the scripts within the test suite (method 3) have to be modified accordingly. Monitoring the system's response is somewhat more complicated. To perform measurements with method 1, appropriate performance counters have to be selected. However, some client/server- or database-driven applications may not cause as high CPU usage on the client as the decompression of images in our case did. This could cause difficulties in detecting the completion of a task using performance counters alone. Method 2 requires large parts of the screen to considerably change the brightness in order for the phototransistors to detect the difference. Any system providing this feature can be analyzed by using method 2. Method 3, on the other hand, is based on the color of single-screen pixels that will be found in any application. Therefore, method 3 can be used to judge the performance of any computer system provided that the scripts are adjusted accordingly.

Requests for all programs and scripts mentioned in this paper can be made by e-mail to m.pietsch@gmx.de. Be advised, however, that all were especially developed for the purpose of these measurements and never passed the status of prototypes.

CONCLUSION

Our results show that monitoring screen-pixel color through the API as done in method 3 was very stable and produced easily interpretable results. In this regard, it is superior to all other methods. The initial work to code self-made software proved worthwhile. The other two methods are still an option in performing the measurements. Method 1 can be recommended for pretests. It is simple and stable, but interpretation of the results may be difficult. Analysis of performance counters, however, may provide additional insights into the architecture of an application.

REFERENCES

1. Terrier F: Web and PACS: heralding the new age of imaging in the health care community. Abdom Imaging 25(4):331, 2000

2. Bick U, Lenzen H: PACS: the silent revolution. Eur Radiol 9(6):1152–1160, 1999

3. Ratib O, Ligier Y, Bandon D, Valentino D: Update on digital image management and PACS. Abdom Imaging 25(4):333–340, 2000

4. Bennett WF, Spigos DG, Vaswani KV, Terrell JE: Cable modem access to picture archiving and communication system images using a web browser over the Internet. J Digit Imaging 13(2 Suppl 1):93–96, 2000

5. Bennett WF, Spigos DG, Tzalonikou MT, Terrell JE, Augustyn MA: Web-based viewing of picture archiving and communications systems images—Part I: Optimal personal computer configuration. J Digit Imaging 12(2 Suppl 1):112–115, 1999

6. Clark KW, Melson DL, Moore SM, James Blaine GJ, Moulton RA, Clayton WK, Peterson CS, Vendt BA: Tools for managing image flow in the modality to clinical-image—review chain. J Digit Imaging 16(3):310–317, 2003

7. Bellon E, Wauters J, Fernandez-Bayo J, Feron M, Verstreken K, Van Cleynenbreugel J, Van den Bosch B, Desmaret M, Marchal G, Suetens P: Using WWW and JAVA for image access and interactive viewing in an integrated PACS. Med Inf 22(4):291–300, 1997

8. Bennett WF, Spigos DG, Tzalonikou MT, Terrell JE, Augustyn MA: Web-based viewing of picture archiving and communications systems images—Part II: The effect of

compression on speed of transmission. J Digit Imaging 12 (2 Suppl 1):116–118, 2002

9. Bergh B, Pietsch M, Schlaefke A, Garcia I, Vogl TJ: Performance of web-based image distribution: server oriented measurements. Eur Radiol 13(11):2419–2424, 2003

10. Bergh B, Pietsch M, Schlaefke A, Vogl TJ: Performance of web-based image distribution: client oriented measurements. Eur Radiol 13(9):2161–2169, 2003

11. Bergh B, Pietsch M, Schlaefke A, Garcia I, Vogl TJ: Upload capacity and time-to-display of an image Web system during simultaneous up- and download processes. Eur Radiol 14(3):526–533, 2004

12. Bergh B, Schlaefke A, Pietsch M, Garcia I, Vogl TJ: Evaluation of a “no-cost” Internet technology-based system for teleradiology and co-operative work. Eur Radiol 13(2):425–434, 2002