

Contents

PREFACE – ABOUT THIS MANUAL.....	5
Organization	5
Conventions	6
Contact Us.....	7
 CHAPTER 1 – INTRODUCTION	 9
ProActive Tools Overview.....	10
New Features in Release 5.2	14
General Testing Information	15
Uses for ProActive Tools.....	17
Structure of a Test Scenario	18
The Scenario Wizards	20
 CHAPTER 2 – INSTALLATION OF PROACTIVE TOOLS	 21
System Requirements	21
Installing ProActive Tools	23
Installing the Scenario Wizard.....	26
 CHAPTER 3 – RUNNING THE “Out of the Box” SCENARIOS	 27
Structure and Purpose of the “Out of the Box” Scenarios	27
Running ProActive Assistant	28
Running ProActive Load	30
Running ProActive Reporter	35

CHAPTER 4 – DETAILED STRUCTURE OF SCENARIO FILES	41
Scenario File Overview	42
The General Section	43
Activity Sets Section	49
Activity Routines Section	54
 CHAPTER 5 – BUILDING & RUNNING YOUR OWN TEST SCENARIOS	 61
Increasing Your Understanding of ProActive Tools	62
Developing the Test Profile	63
Designing the Test Scenario	65
Using the Scenario Wizards	68
Debugging the Test Scenario	68
Running the Test Scenario.....	71
Analysis of the Results	73
Tips and Techniques	74
 CHAPTER 6 – THE PROACTIVE REPORTER	 77
ProActive Reporter Overview.....	78
Importing Result Files	80
Assigning Test Names.....	82
Selecting Test Results	83
Standard Report Selection.....	85
Developing Custom Reports using the Report Wizard.....	94
Deleting Test Files from the Reporter Database	100

CHAPTER 7 – PROACTIVE SCENARIO WIZARDS	103
ProActive Scenario Wizards Overview.....	104
Using the Scenario Wizard to build a Scenario File.....	105
Using the Scenario Wizard to build an Activity Routine.....	107
Exporting Scenario Files.....	108
 APPENDIX A – SCENARIO FILE PARAMETERS	 109
General Section	110
Monitor Section.....	115
Activity Sets Section	118
Calendar Activity	121
Compose Activity	124
Copy Database Activity	127
Edit Activity.....	129
Find Activity	133
Index Activity.....	135
Mail Activity	137
Read Documents Activity.....	140
Rebuild View Activity	142
Reset Result Activity.....	144
User Defined Activity	146
Views Activity	155
Text Parameters	156
Invoke Method Parameters	164

APPENDIX B – SAMPLE SCENARIO FILES	167
Sample Assistant Scenario File	167
Sample Load Scenario File	174
 APPENDIX C – ERROR MESSAGES/TROUBLESHOOTING..	 181
ProActive Assistant Error Messages	181
ProActive Load Error Messages & Trouble Shooting	185
 APPENDIX D – FILE LAYOUTS	 189
Table of Limits	189
Result File Layout.....	190
TimeLine File Layout.....	191
 Glossary.....	 193
 Index.....	 197

Preface – About This Manual

Organization

This manual is designed to take the person new to ProActive Tools through the process of installing ProActive Tools, running the “Out of the Box” scenarios, and then moving progressively into the details of the scenario file and its construction. The appendices provide a convenient reference for the parameters used in Activity Routines, sample scenario files, and troubleshooting information. This makes the manual extremely usable for both the novice and experienced tester.

A brief description of each chapter is as follows:

Chapter 1 – Introduction. A general description of ProActive Tools and how they are used to test Lotus Notes applications and Lotus Domino server.

Chapter 2 – Installation. A guide to the simple installation of ProActive Tools.

Chapter 3 – Running the “Out of the Box” Scenarios. ProActive Tools is installed with a sample scenario file for each ProActive Assistant and ProActive Load. The running of these scenarios provides a validation of the installation and a starting point for the new user to become familiar with the functioning of ProActive Tools.

Chapter 4 – Detailed Structure of Scenario Files. A detailed explanation of the scenario file and the parameters that are used in their construction.

Chapter 5 – Building & Running Your Own Test Scenarios. Having run the “Out of the Box” scenarios and reviewing the scenario file structure, you are now ready to start building test scenarios to be used in your own environment.

Chapter 6 – ProActive Reporter. Provides a more detailed explanation of how to use ProActive Reporter in the analysis of response times generated with the ProActive Assistant scenario.

Chapter 7 – The ProActive Scenario Wizard. The Scenario Wizard is a learning tool that assists the new user in understanding and developing test scenarios.

Appendix A – Activity Routine Parameters. A handy reference of all of the parameters that are used by the Activity Routines. Also contains an explanation of the use of text parameters, parameters used in the General Section, and samples of Activity Routines.

Appendix B – Sample Scenario Files. Scenario files for both ProActive Assistant and ProActive Load that demonstrate many of the Activity Types available for developing test scenarios.

Appendix C – Error Messages and Troubleshooting. Sample error messages and what to look for if problems occur when running ProActive Tools.

Appendix D – File Layouts. A description of the structure of the various output files generated by ProActive Assistant and ProActive Load.

Conventions

Names of keys appear in bold, capital letters: **ENTER**

Information you type appears in a smaller, bold typeface: type **a:\setup**

Menu commands appear in bold: click on the **File** menu

Parameters are placed in sections using the bracket notation common to most .INI files. For example, [**General**].

Optional elements are enclosed in curly braces: {**optional keyword**}

Contact Us

With your help, future versions of G2's ProActive Tools will assist Lotus Notes administrators in even more ways to evaluate and fine-tune their systems. Please contact us with your problems, comments and suggestions.

G2 Associates Inc.
6723 Whittier Avenue
McLean, Virginia 22101

800-884-5876
703-288-2944
703-288-2946 (Fax)

Please send e-mail to ProActive@g2sys.com or visit our web site at **www.ProActiveTools.com**

Chapter 1 – Introduction

Chapter 1 provides an overview of G2's ProActive Tools product suite including:

ProActive Tools Overview

- ProActive Assistant
- ProActive Load
- ProActive Reporter
- New Features in Release 5.2

General Testing Information

- Real-World Answers to Real-World Questions
- Lotus Notes/Domino Testing

Uses for ProActive Tools

Structure of a Test Scenario

- The General Section
- The Activity Sets Section
- The Activity Routines Section
- Events
- The Result Files

The Scenario Wizards

ProActive Tools Overview

ProActive Tools can be **used to *analyze existing or planned Domino configurations*** and provide empirical data to help plan for new Notes applications, new Notes releases, server upgrades, and upgrades to the network infrastructure. It is ***highly customizable*** and enables administrators to mimic very specific sets of conditions and events within an application or on the server.

ProActive Tools was designed to provide Notes engineers and administrators a method of testing Notes applications and Domino servers in the lab *before* they are implemented for the business community. If Notes applications are to be implemented successfully, Notes engineers and administrators need to understand system and application performance. There is plenty of information about Notes and Domino performance available from Lotus and the hardware vendors, but all of it general in nature. Engineers often need information specific to their company's environment, hardware configuration, and usage patterns.

ProActive Tools was developed with exactly those needs in mind. ProActive Assistant is the only tool that operates with the Notes User Interface and accurately measures user response time for a variety of Notes activities. ProActive Load has the same easy-to-use scripting language as Assistant, but it does not operate through the Notes UI. Rather, ProActive Load is a multi-threaded tool that allows you to simulate hundreds of Notes user sessions from a single workstation. ProActive Reporter is used to develop response time analysis and reports from the data generated by ProActive Assistant.

Used together, these products are an ideal tool for Lotus Notes administrators and engineers to do capacity planning, application response testing and performance analysis. They can easily be set up in the customer's environment, and the necessary scripting can be done for different test scenarios.

ProActive Assistant

ProActive Assistant is a *workstation-based* performance testing, monitoring, and analysis tool for Lotus Notes clients and servers, networks, and Notes applications. With ProActive Assistant, Notes administrators can create repeatable performance test scenarios that run in either lab or production environments to measure individual system components or *enterprise-wide* performance. Because it is workstation-based, ProActive Assistant gives a true picture of performance from the end-user perspective.

ProActive Assistant initiates Notes activities and events whose resulting response times are measured at the Notes client workstation. This end-user perspective enables the entire system's performance to be evaluated, not just the Domino server. If network protocols or routers affect response times, their impact will be measured by Assistant. The variety of activities available to the test designer allows the testing to be tuned to focus on specific aspects of server or system performance. For example, mail routing performance can be stressed, or full-text search response times can be measured, depending on the actual problem being addressed in the test.

Assistant is controlled by a series of parameters contained in an ASCII parameter file referred to as the scenario file. This scenario file allows the workstations to be configured independently, with each performing its pre-defined set of Lotus Notes activities. Alternatively, all workstations may use the same scenario from a shared network drive.

The parameters in the scenario file also allow each workstation to pause for a specified amount of time before proceeding to the next activity. The capability to adjust wait times is one of the most powerful features in ProActive Assistant. A few workstations with very short wait times between activities can be used to simulate multiple users. Scenarios with longer wait times represent lighter server loads and more accurately simulate real world operations. Since ProActive Assistant represents only one Notes session per workstation, it can be used equally well in both production and test environments.

ProActive Assistant can also perform real-time monitoring of production servers. Assistant parameters can specify that certain activities be performed periodically during the day. Parameters can also specify that alerts should be sent when

response times for specific activities exceed threshold values. These alerts can be sent via Notes or SNMP trap messages. By measuring response time from the user perspective, Assistant can monitor actual system performance, rather than just server performance.

ProActive Assistant can also be effective in testing system performance across wide-area networks. This is useful for organizations that need to determine how best to support their mobile users, or to enhance system performance for Notes users who are dispersed geographically.

ProActive Assistant produces a comma separated variable (CSV) output file that can be viewed/analyzed with a text editor or spreadsheet application. This output file is also used as the input to ProActive Reporter for developing response time comparison reports.

ProActive Load

ProActive Load is a *multi-threaded Notes API-based* performance testing and analysis tool for Lotus Domino servers and Lotus Notes applications. With ProActive Load, Notes administrators can create repeatable performance test scenarios that let them address a variety of performance and capacity questions for Notes applications and Domino servers. With several workstations, the activities of thousands of Notes users can be simulated for one or more Domino servers. Used in conjunction with ProActive Assistant, ProActive Load provides an accurate and complete picture of performance from both the user and system perspectives.

While a single workstation running ProActive Load can simulate hundreds of Notes users (or threads), the maximum number of threads run on a workstation is limited by the power of the workstation, the complexity of the scenario, and the ability of the network to carry the generated load. For large tests, you can set up multiple workstations to run the appropriate number of threads. ProActive Load is extremely customizable, in that each thread can perform the same set of activities or different activities by thread or groups of threads. This allows engineers to design a variety of test scripts that can accurately create the same server load that actual users would develop. Each Load thread represents a single Notes session on the Domino server, and therefore, normally a single Notes user.

The ProActive Load test scenarios are defined by the same type of parameter driven scenario file that is used by ProActive Assistant. The actual syntax of the scenario file is similar for both Load and Assistant and most of the same activities and features are supported in both. ProActive Load supports multiple threads and pushes API calls to the Notes server, while ProActive Assistant uses the Notes UI to create Notes activity. This is the main reason for the difference in supported activities and features.

ProActive Load produces two output files. The first output file has the same structure as the output file created by ProActive Assistant and records response times in similar fashion. These results are mainly used to ensure that the scenario is working as designed, but also can be compared over time to evaluate differences in API response times. This file can become very large when multi-threaded scenarios are run and API response times are not as meaningful as the response times reported by ProActive Assistant. As mentioned above, we recommend running ProActive Assistant along with ProActive Load to monitor server response time. The second output file is the timeline file, which collects the information displayed on the Load UI, and writes it to a text file, at one minute intervals.

ProActive Reporter

ProActive Reporter processes the results files created by the Assistant workstations, and produces analysis and comparison reports. The results files from all or a subset of the workstations in a test can be combined within the Reporter. Running under Windows 95/98 or NT, the Reporter contains seven predefined reports and a Report Wizard for developing custom reports.

Each report includes the number of times an activity was repeated, average response times, minimum and maximum response times, standard deviations and variance from a user-selectable benchmark or baseline test. With the Report Wizard, users can select the specific data they want included, as well as specify how to sort and group the data in the report.

All ProActive Assistant results are reported to the result file in a comma separated variable format. This allows the results to be imported by spreadsheet programs and report-generation packages for additional analysis.

Additional information on ProActive Reporter is available in Chapters 3 and 6.

Chapter 2 describes the use of ProActive Reporter by actually stepping through importing a result file and producing the reports. Chapter 6 provides a more detailed explanation of the features of the ProActive Reporter.

New Features in Release 5.2

ProActive Assistant

- Reset Results is a new Activity type for ProActive Assistant that allows users to reset the CSV file so that it appears that Assistant has been restarted for a separate test without having to restart Assistant.
- The German Language version of NT is now a supported platform for ProActive Assistant.
- Invoke Method Parameters provide another way of executing functions in the Notes UI without relying on duplication of individual keystrokes. These parameters rely on the use of OLE and tend to be more reliable and predictable than the normal keystrokes.
- Monitor has been upgraded to work with new features of Domino R5 servers.

ProActive Reporter

- Reporter now supports European date formats.
- Import error checking has been improved

ProActive Load

- Realtime graphing has been added to Load's Front End. An optional Graph window graphically displays the progress of the six counters that are reflected on the Main screen.
- Load reports can now be saved as a text file.
- Users can now specify a size for FieldSize in the Compose, Edit, Mail, and User Defined Activities in ProActive Load.
- Load can now handle lists and null fields in the User Defined parameters.
- The <iteration> text tag can now be used in User Defined activities in Load. The <iteration> tag can include both negative and positive numbers and ranges. For example, <iteration -1>, or <Iteration +1-5>.
- You can now combine <thread> with @Variable to allow each thread to get a different variable out of a text file.

General Testing Information

Real-World Answers to Real-World Questions

The use of ProActive Tools can address many of the everyday issues faced by Notes system administrators. Some of the issues might include:

- What is the impact on user response time when changing from one network protocol to another, e.g., NetBios to TCP/IP?
- Is there any noticeable impact on user response time caused by varying tuning parameters or server configurations?
- How will the new release of Notes/Domino impact performance?
- How do wide-area connections between users and servers impact user response time?
- Which operating system/hardware platform is best for this environment?
- Is this server continuing to provide adequate response time to users as new applications are added?
- Is the new server configuration capable of handling the projected number of users?
- What is the user capacity of the current server configuration?
- Can we tell how well this server is performing compared to last month?

Lotus Notes/Domino Testing

Several items should be considered when planning performance testing for Lotus Notes and Domino.

- The first is always the most critical. What is to be tested, or ***“What is the question to be answered?”*** All testing is based on the desire to answer some question or resolve some perceived problem. A clear definition and understanding of the question or problem is critical before any sort of test plan can be developed.
- Consider the development of a test environment. Server and application load testing usually requires a separate server. Test environments should always be maintained separately from the production environment.
- ***The ultimate measure of good performance is what the user experiences.*** As Lotus Domino and the hardware platforms that support it become increasingly robust and reliable, user response time testing becomes an even more critical measure of application and server performance.
- Different types of tests will require different approaches, and will dictate the types of user profiles and scenarios defined. For example, performance analysis of an application will generally require more complex scenarios than a test to compare different server configurations. However, in almost all cases, we believe that the user response times measured by ProActive Assistant should be a key factor to be considered in analyzing performance.
- Focus on the peak hour and maximum user concurrency when designing tests. This is where performance and response time are critical.
- Before the actual test scenario can be constructed, you should develop a profile of Notes activities that will be executed as part of the test. The information in this profile will become the basis for the scenario files. For example, in messaging tests the profile would include the number of messages to be delivered, average message size, number of attachments, size of attachments, and number and names of local and remote addressees. There may be multiple profiles, each representing a different type of user: power, heavy, light, or casual.

- G2 Associates, Inc. recommends running each test at least twice to ensure consistent results. If test results are not consistent, something in the environment may be incorrectly set and you may not have an accurate or repeatable test.
- In designing a complete performance analysis effort, it is generally useful to plan a series of tests that will provide multiple data points. This is true for most types of performance questions. The first data point is usually a single user. Other data points should build incrementally to a maximum. For example, in evaluating capacity of a mail server, you might want to have response time information for 1 user, 100 users, 500 users, and 1,000 users. This will show you the change in response time as load increases, and let you determine the capacity of a server or application.
- If you are running a test with lots of Load threads (users), consider the ramp-up time when designing the scripts. With several hundred threads, it can take several minutes for all of the threads to become active and the server's performance to stabilize. Any response times measured during this ramp-up time will probably not be accurate. Assistant scenarios should be run after Load and the Domino server have reached a 'steady state'.
- Develop a profile of the users and their activities that will be used for designing the actual test scenarios. More detail on developing these profiles is included in Chapter 5.

Uses for ProActive Tools

Used together, ProActive Assistant and ProActive Load create a way for Notes engineers to mirror their production environment, allowing for stress-testing, performing 'what-if' scenarios, and testing applications. These two products are designed to work together to provide a complete set of testing tools for a Lotus Notes/Domino implementation.

ProActive Assistant is the best tool to measure the actual performance that a user would see. ProActive Assistant can be used in the production environment to measure user response times in different situations. In these cases, Assistant is run without Load, emulating a real user. Tests can be designed to compare response times between different locations within a company, or on different

workstation configurations. Assistant scenarios can also be designed to monitor the response times of production applications and servers. If desired, response time problems can be reported in real-time using Assistant's monitor feature.

ProActive Load simulates a large number of Notes users performing various functions in order to replicate the load on the system that actual Notes users would create.

ProActive Load scenarios can easily be designed to run different numbers of threads. This allows you to create a series of tests at different load levels. For example, to determine the capacity of a server, you might want to run tests at load levels both above and below the expected capacity level.

Structure of a Test Scenario

ProActive Assistant and ProActive Load support a variety of activities that simulate real-world Notes usage. These activities are combined into a scenario parameter file for these products to run. The variety of activities available to the scenario writer allows the testing to be tuned to focus on specific aspects of server performance. See Chapter 4 for details on scenario file structure.

The actual scenario file used by either Assistant or Load is composed of three sections. The General Section, the Activities Set Section, and the Activity Routines Section. All three sections must be present in a valid scenario file. Each of these sections is described briefly below.

General Section

The General Section for each Assistant and Load contains the general scenario parameters for managing the execution of a specific test and the parameters that are default values for all of the Activities Routines that are to be executed as part of the scenario.

A listing and description of all the parameters that can be used in the General Section of scenario files is contained in Appendix A.

Activity Sets Section

The Activity Sets Section represents the order and timing of the Activity Routines that are to be executed during the running of the test scenario. The Activity Sets Section has the same structure for both Assistant and Load, except that ProActive Load can have multiple Activity Sets that can be assigned to specific threads. Since ProActive Assistant represents a single Notes session (equivalent to a single thread in ProActive Load), it can have only one Activity Set per scenario.

Not only can the individual Activity Routines be “called” in the Activity Sets Section, but this section also can use SleepTimes to control the time between the execution of Activity Routines and Do Groups to repeat single or groups of Activity Routines. Both of these capabilities are covered in detail in Chapter 4, along with examples of Activity Sets sections.

Activity Routines Section

The Activity Routines Section contains the details of the Activity Routines that were called in the Activity Sets Section. Each Activity Routine is defined individually and can be located in any order in the Activity Routines Section, since the order of execution is defined in the Activity Sets Section. Each Activity Routine can represent a single or multiple Notes Activities, such as reading documents, sending a mail message or creating documents.

ProActive Tools contains eleven predefined activities for ProActive Assistant and ten for ProActive Load. These predefined activities provide for the development of Activity Routines for common Notes activities without having to specify all of the required information. Both products provide a User Defined Activity Type that can be used to develop Activity Routines for custom databases and applications.

Events

Each Notes activity consists of a series of Events: opening a database, opening a view or document, entering text, sending a message, etc. It is these events that are actually executed by Assistant and Load and their response times are recorded in the Result Files of the two products. It should be noted that while the response times generated by ProActive Assistant are user response and are

significant in the performance analysis of Lotus Notes applications, the response times generated by ProActive Load are response times of API calls and are not representative of any user activity.

Result Files

ProActive Assistant and ProActive Load generate a comma separated variable (CSV) Result File that is used to analyze the results of the tests. The Result File is a record of each event executed by the scenario and the resulting response time for that event. The Result File generated by ProActive Assistant is used as the input to the ProActive Reporter, which is used for the analysis and reporting of the results of tests. The structure of the Result File is shown in Appendix D.

ProActive Load also generates a TimeLine File that collects the results of the progress counters from the Load User Interface. This information is collected at one-minute intervals. The format of the TimeLine File is also described in Appendix D.

The Scenario Wizards

ProActive Tools contains two Scenario Wizards, one each for ProActive Assistant and ProActive Load, that can be used to develop the actual scenario files used by these products.

The Scenario Wizards are Notes databases, which are used to step you through the selection of the parameters necessary to construct the scenario files run by ProActive Load and ProActive Assistant. The Wizards are used primarily as learning tools that help new users develop the ProActive scenarios. Experienced users can continue to use the Scenario Wizards or edit existing scenarios with a text editor employing a cut-and-paste method.

The Scenario Wizards provide assistance in selecting the appropriate parameters to be used in the scenario file and provide help in understanding and choosing the correct value for each of the parameters.

A detailed description of how to use the Scenario Wizards is contained in Chapter 7, while the installation of the Wizards is in Chapter 2.

Chapter 2 – Installation of ProActive Tools

This Chapter covers the System Requirements for ProActive Tools and the installation of ProActive Tools onto test workstations. In addition, the installation of the ProActive Scenario Wizards is included.

The installation instructions step new users through the easy installation process and discuss some of the installation options that are available.

In addition to the files that are installed on the Windows workstation during the installation process, there are Help files in a PDF format and the Scenario Library Notes database that are included on the ProActive Tools CD in the Documentation subdirectory.

System Requirements

ProActive Assistant

Operating System:

- Microsoft Windows 95
- Microsoft Windows 98
- Microsoft Windows NT 4 (English or German)
- Microsoft Windows 2000 (English or German)

Hardware:

- Pentium (200 MHz or higher) processor
- 32 MB system RAM
- 2 MB of free disk space

Software Requirements:

- Lotus Notes Release 4.5x, 4.6x or 5.0x (North American, International English and Global English version)
- Lotus Notes must be on the system path for Monitor function to work.

ProActive Load

Operating System:

- Microsoft Windows NT 4
- Microsoft Windows 2000

Hardware:

- Pentium (200 MHz or higher) processor
- 64 MB system RAM (256 MB or higher recommended)
- 5 MB of free disk space

Software requirements:

- Lotus Notes Release 4.5x, 4.6x or 5.0x
- Lotus Notes must be on the system path.

ProActive Reporter

Operating System:

- Microsoft Windows 95
- Microsoft Windows 98
- Microsoft Windows NT 4
- Microsoft Windows 2000

Hardware:

- Pentium (200 MHz or higher) processor
- 32 MB system RAM
- 12 MB of free disk space

Software requirements:

- None

Installing ProActive Tools

ProActive Tools software is easy to install and only requires a few minutes. There are several installation options from which to choose. We recommend that new users accept the first install option, "Default - Install ProActive Tools (Entire Toolset)."

To install ProActive Tools, follow the steps below.

1. Check to be sure Notes is installed on the workstation. If not, install Notes before installing ProActive Tools.
2. Close all other programs.
3. Insert the installation CD in your CD ROM drive.
4. Click the Windows Start button.
5. Select run from the Start menu.
6. Browse the CD containing the ProActive Tools installation software.
7. Select Setup.exe and run the setup program.
8. Click Next to Accept the "Welcome" and "License" panels.
9. Select one of the four options presented on the "Setup Type" panel. These options are explained below in the section titled "Installation Option Explained".
10. Choose a program directory for the installation. The default is C:\Program Files\PATools.
11. Enter the Notes\Data directory for the workstation. The Install procedure will default to the directory with the first instance of Names.nsf found on your workstation. Sample.nsf, an R4 discussion database is copied to the specified Notes\Data directory. The sample scenarios installed with the product look for Sample.nsf in this directory.
12. Select a Program Group for a shortcut in the Windows Start Menu. The default is "G2's ProActive Tools".
13. A final screen reviews the choices made. At any time, use the "Back" buttons to return to previous screens.

Description of Installation Options

The four installation options referenced in Step 9 above, are explained below.

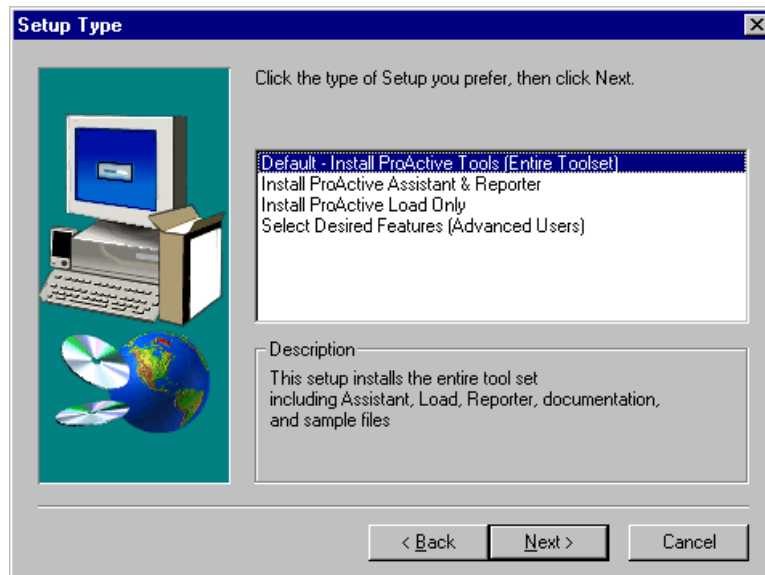


Figure 2-1: Setup Type

Default - Install ProActive Tools (Entire Toolset)

The default option installs the entire toolset including Assistant, Load, Reporter, documentation, and sample files. This option is **recommended for new users**.

Install ProActive Assistant & Reporter

This option installs Assistant, Reporter, documentation, and sample files.

Install ProActive Load Only

This option installs Load, documentation, and sample files.

Select Desired Features (Advanced Users)

This option allows advanced users to install desired components.

Documentation and Sample Files

Documentation and sample files are installed as described below unless deselected with the fourth option, Select Desired Features (Advanced Users). These files are also located in the Documentation directory on the installation CD. Remove the “Read Only” property from files copied from the CD.

PAToolsDoc.pdf is installed in the PATools directory. This file contains the same information as the printed software manual. Adobe Acrobat 4.0 or higher is recommended for viewing this file.

PAAQuickStart.pdf guides new users through running the Assistant “Out-of-the-Box” scenario and is located in the Assistant program directory.

PALQuickStart.pdf guides new users through running the Load “Out-of-the-Box” scenario and is located in the Load program directory.

There are four Notes databases installed in the directory specified by the user as described in step 11 above. These databases can be added to the Notes desktop and accessed by the Notes client.

Sample.nsf is a simple R4 database used to test the installation of ProActive Tools. It is also useful for first attempts at building scenarios. In order for the “Out-of-the-Box” scenario to run without error, either Sample.nsf must be located in the default Notes data directory or the scenario must be modified accordingly.

PAToolsScenarioLibrary.nsf is an R4 Library database containing example scenarios for Load and Assistant.

Awizard.nsf is an R4 database that provides a GUI method for building Assistant scenarios. This database is only installed if Assistant is installed.

Ldwizard.nsf is an R4 database that provides a GUI method for building Load scenarios. This database is only installed if Load is installed.

Installing the Scenario Wizards

ProActive Tools Scenario Wizards can be used to create scenario files to be run by ProActive Load and ProActive Assistant. These Notes databases, Awizard.nsf and LdWizard.nsf, are primarily learning tools that help new users build activities and scenarios.

The Wizards are automatically installed with the first three installations described above. The Wizards can also be installed from the CD by the following procedure.

- Insert ProActive Tools CD into the CDROM drive.
- Locate the documentation directory.
- Copy both LdWizard.nsf and AWizard.nsf to a directory where you can open the databases from Notes. Remove the read-only attributes from both files.
- Place the databases on your Notes desktop by selecting **File, Database, Open.**

Chapter 3 – Running the “Out of the Box” Scenarios

This chapter includes instructions for running the “Out of the Box” scenarios that are automatically installed with ProActive Tools. This chapter contains the following components:

- General description, structure and purpose of the “Out of the Box” scenarios that are included with ProActive Tools and what you should expect when they are run.
- Instructions for running the ProActive Assistant “Out of the Box” scenario.
- Instructions for running the ProActive Load “Out of the Box” scenario.

Structure and Purpose of the “Out of the Box” Scenarios

The “Out of the Box” scenarios for both ProActive Assistant and ProActive Load are used as tests to insure the proper installation of ProActive Tools and provide working examples of scenario files.

These scenarios work in local mode so they don’t require a Domino server, and they function with the Sample.nsf database. Sample.nsf can also be found in the documentation directory on the installation cd.

Running the ProActive Assistant “Out of the Box” Scenario

Go to the **Start** menu, select **Programs** and go to the **G2’s ProActive Tools** folder and select **ProActive Assistant**. Following the initial ProActive Assistant splash screen, the **Start** screen for ProActive Assistant will appear. See *figure 3-1* below.

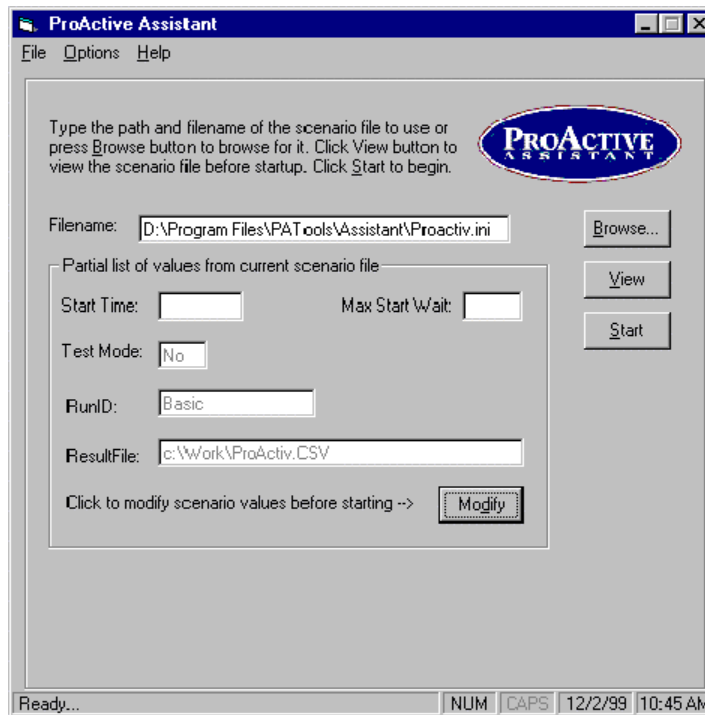


Figure 3-1: Assistant Start Screen

The ProActive Assistant Start screen displays the key information required to identify the scenario that has been selected to be executed by ProActive Assistant.

Using either the **Browse** button or by manually typing, specify the location of the scenario file that contains the scenario you wish to run. The “Out of the Box” scenario file is named PROACTIV.INI and is located in the <drive>\Program Files\PATools\Assistant subdirectory. While PROACTIV.INI represents the default name for a scenario file, they can have any valid file name and be located in any subdirectory.

The **View** button allows you to view, but not edit, the selected scenario.

When ready to proceed with the execution of the scenario, press the **OK** button.

The **Modify** button will allow you to modify some of the parameters in the scenario file at the time the scenario is run. For more information about this feature, see the “Debugging The Test Scenario” section of Chapter 5.

During the execution of the Assistant “Out of the Box” scenario, you will be able to view each of the activities as they take place on the Notes client. The status box in the lower right corner of the screen keeps you constantly informed about the progress of the scenario and which activity is being executed.

In this case, the ProActive Assistant “Out of the Box” scenario consists of four activities: Read Documents, Find, a Copy Database and a User Defined.

When the execution of the scenario has completed, the following dialog box will appear:

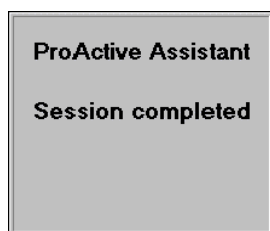


Figure 3-2: Assistant Complete

Running the ProActive Load “Out of the Box” Scenario

From the **Start** menu, select **Programs**, then the **G2'sProActive Tools** folder, and **ProActive Load**. After the splash screen, the Startup screen for ProActive Load will appear.

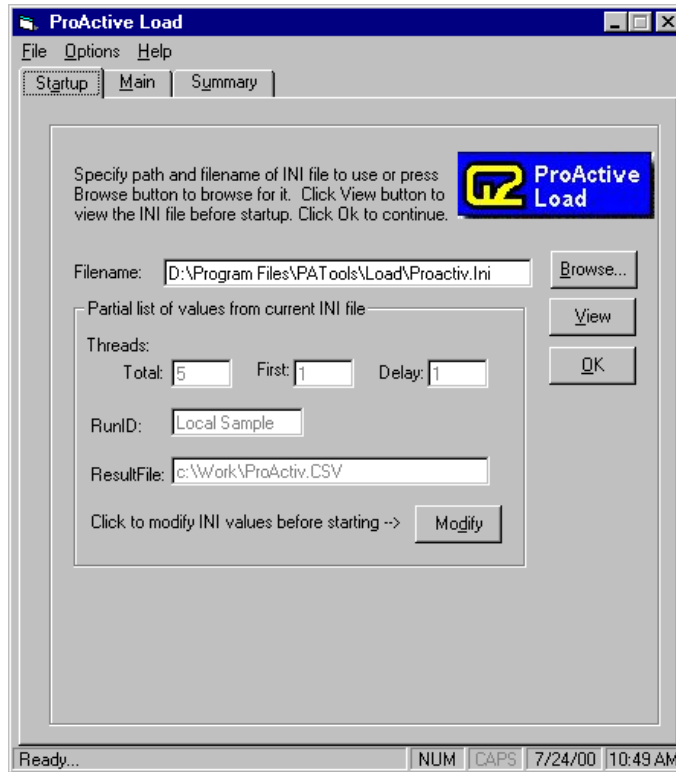


Figure 3-3: Load Startup Screen

The Startup screen displays the key information required, identifying the scenario selected to be executed by ProActive Load.

Using either the **Browse** button or by manually typing, specify the location of the scenario file that contains the scenario you wish to run. The “Out of the Box” scenario file is named PROACTIV.INI and is located in the <drive>\Program Files\PATools\Load subdirectory. While PROACTIV.INI represents the default name, Scenario files can have any valid file name and can be located in any subdirectory.

The **View** button allows you to view, but not edit the selected scenario.

When you are ready to proceed with the execution of the scenario, press the **OK** button.

The **Modify** button will allow you to modify some of the parameters in the scenario file at the time the scenario is run. For more information about this feature, see the “Debugging The Test Scenario” section of Chapter 5.

After you press the **OK** button, you will be moved to the Main screen of ProActive Load. Once there, press the **Start** button to initiate the execution of the scenario.

The progress of the “Out of the Box” scenario is shown on the Main screen and will require just over 2 minutes to complete. The status information on the screen will automatically be refreshed every 10 seconds during the execution of the sample scenario.

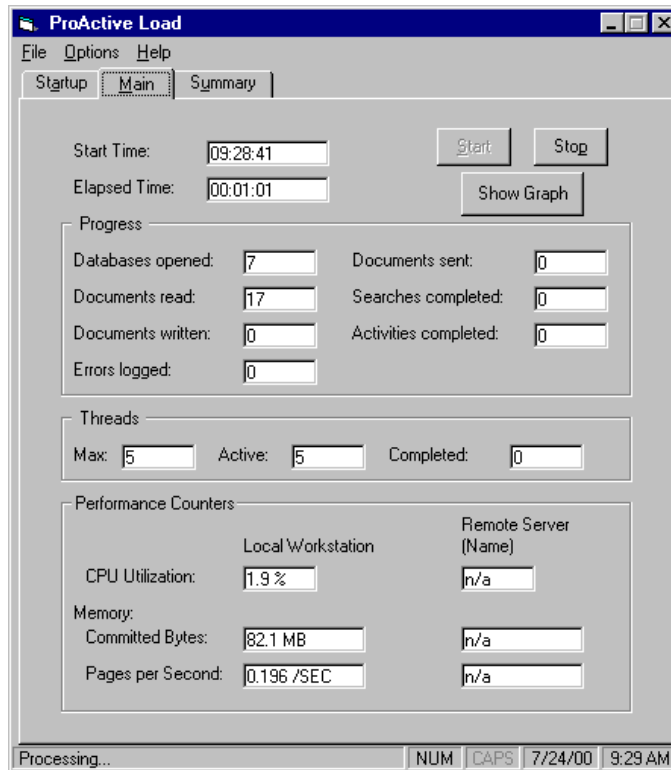


Figure 3-4: Load Main Screen

When ProActive Load is running, the Local Workstation Performance Counters will display the workstation's memory and CPU utilization. These numbers help ensure that the workstation is not pushed too hard. Exceeding the workstation's capability may invalidate tests.

You should not experience any errors during the execution of the “Out of the Box” scenario.

Once the execution of the scenario starts, a **Show Graph** button will appear on the Main screen just below the **Start** and **Stop** buttons. When the **Show Graph** button is pressed, ProActive Load will bring up an optional Graph window, shown below, that will graphically display the progress of the six counters that are reflected on the Main screen. The “Out of the Box” does not generate a very significant load on the workstation, so the Graph does not indicate much activity. When you run other scenarios with a heavier load, the Graph will prove to be more useful.

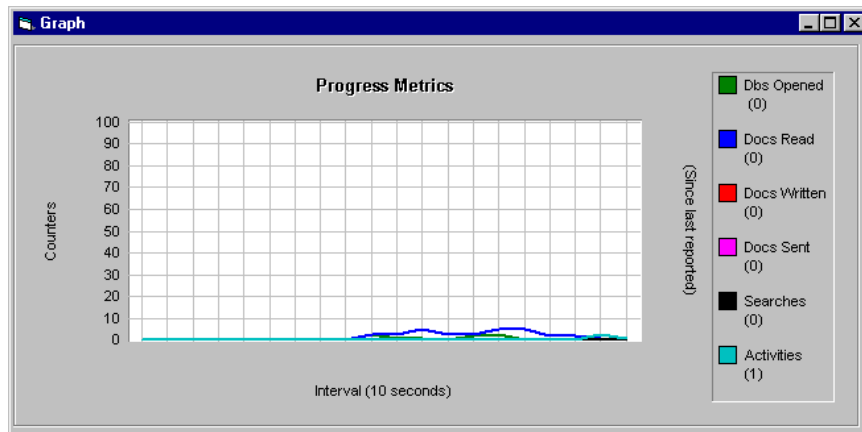


Figure 3-5: Load Graph Screen

Once the Graph is displayed, the **Show Graph** button will change to **Hide Graph**. This will allow you to remove the Graph screen.

When the test is complete, you will be prompted to see results on the Summary screen. Click the **Yes** button to move to the Summary screen to view the final results of the test. If you Press the **No** button you will remain on the Start screen and can restart the scenario. If you press the **No** button and the scenario has generated errors, you will be additionally prompted to view the errors.

ProActive Load

File Options Help

Startup Main Summary

Start Time: 10:43:30 End Time: 10:45:46 ElapsedTime: 00:02:16

Save Results

Progress

Databases opened: 10 Documents sent: 0

Documents read: 30 Searches completed: 0

Documents written: 0 Activities completed: 5

Threads

Max: 5 Active: 0 Completed: 5

Performance Peak Values

	Local Workstation	Remote Server (Name)
CPU Utilization:	6%	n/a
Memory:		
Committed Bytes:	119.7 MB	n/a
Pages per Second:	116/Sec	n/a

PALoad processing complete on 10:45:46 NUM CAPS 8/30/00 10:46 AM

Figure 3-6: Load Summary Screen

The Summary screen shows the final values for the ProActive Load progress counters and peak values for the NT performance counters are shown at the bottom of the screen.

From here you can save the summary report to a file or print the results by pressing the **Save Results** button. Comments relating to the test may be added to the report. Results can be saved as text or CSV files. When saved to the same filename, multiple runs are appended to the bottom of the file. Whether you print the report or save it to a file, you will be returned to the Summary screen when that activity is completed.

Running ProActive Reporter

Go to the **Start** menu, select **Programs** and go to the **G2's ProActive Tools** folder and select **ProActive Reporter**. Following the initial splash screen, the Main screen for ProActive Reporter appears.

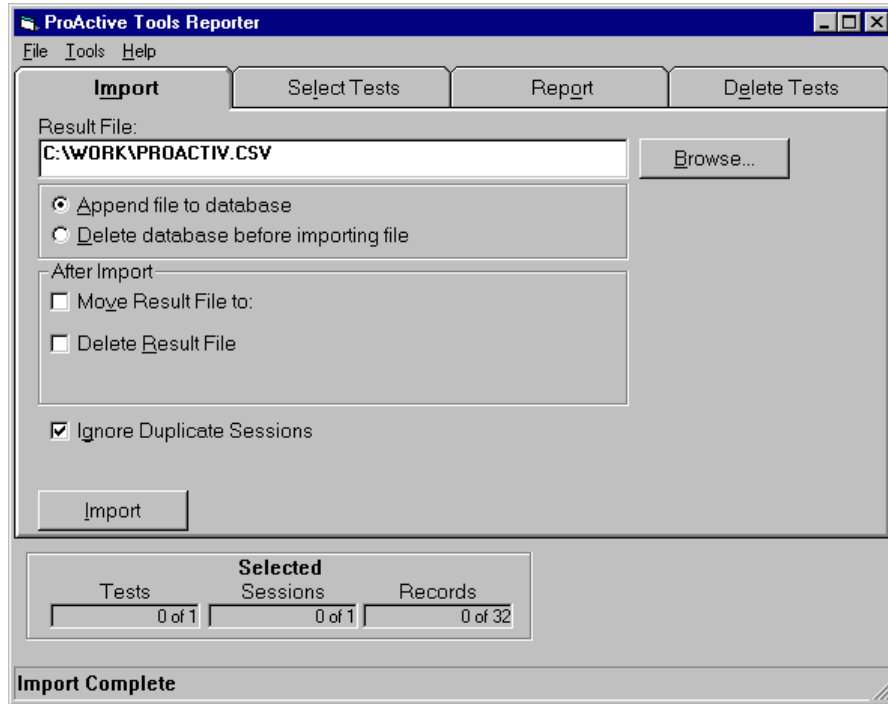


Figure 3-7: ProActive Reporter Import

The default Result File shown on the Reporter Import Panel is the same as the one generated by the ProActive Assistant “Out of the Box” scenario when it was run. Press the Import button and this result file will be loaded into Reporter for analysis.

Once the import of the result file is complete, you will be requested to enter the Test Name and Description for the result file just imported. See *figure 3-8*. This information is not required, but is recommended, as it helps distinguish this test from the others that will be imported in the future. Once the Test Name has been entered, press the Done button at the bottom left of the screen and you will be returned to the Import Panel. See *figure 3-7*.

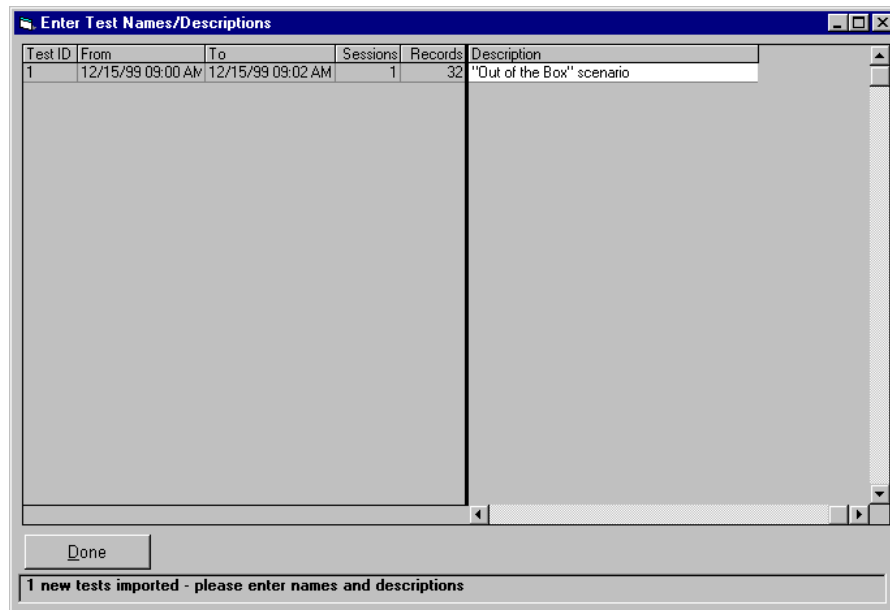


Figure 3-8: Reporter Test Name

Next, select the Select Tests Panel, see *figure 3-9*, and use the mouse to select the test by clicking on the space to the left of the test name. Only the tests that are “selected” will be included in the reports generated by ProActive Reporter.

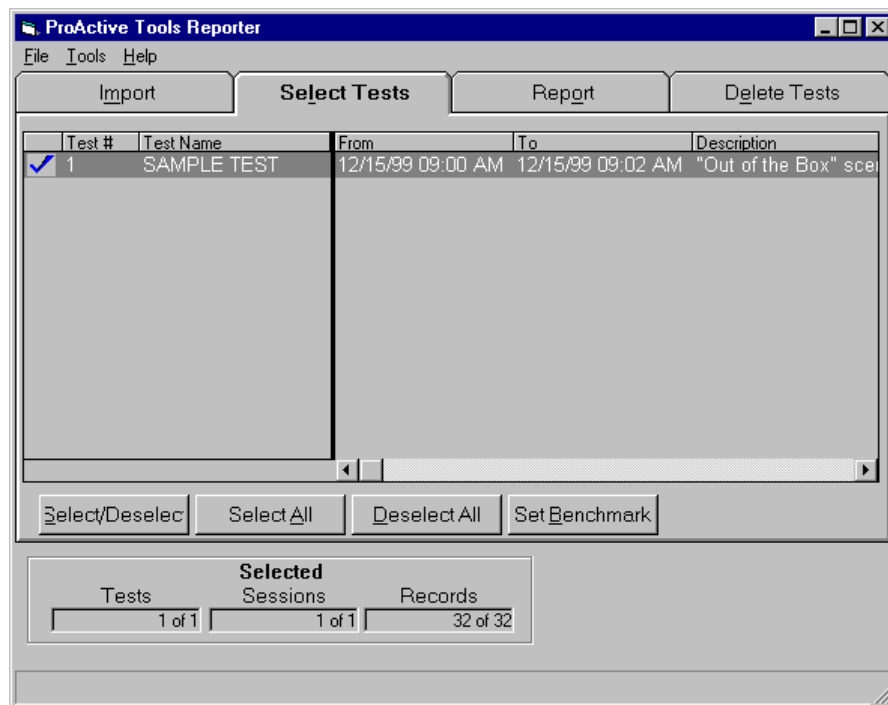


Figure 3-9: Reporter Select Tests

In addition to the Test number and the Test Name that you previously entered, the Select Tests panel displays the test date, test start and stop times, and the test description, if one was entered previously.

Note: There is a **Change Test Names** selection under the **Tools** menu item that can be used to make changes to the Test Name and Description.

Next, select the Report panel, see *figure 3-10*. The Report panel displays the reporting options available for ProActive Reporter. In this case, select Analysis of Events to see a summary report of response times for all of the events from the execution of the “Out of the Box” scenario.

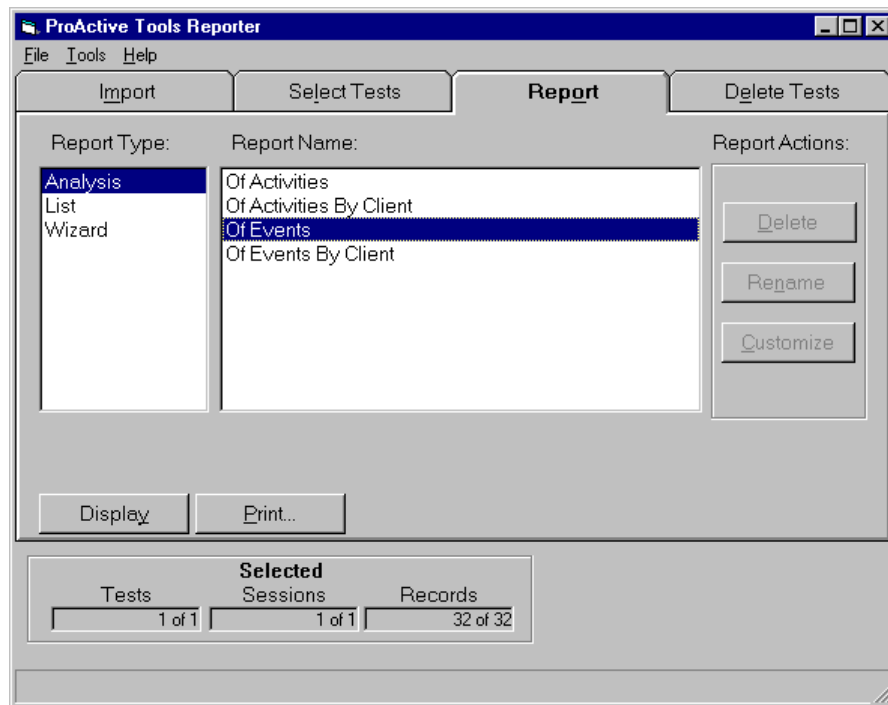


Figure 3-10: Reporter Report Panel

Press the Display button to display the Analysis of Events report, see *figure 3-11* on the next page.

Note: A description of the other Report Names and Report Types are described in Chapter 6.

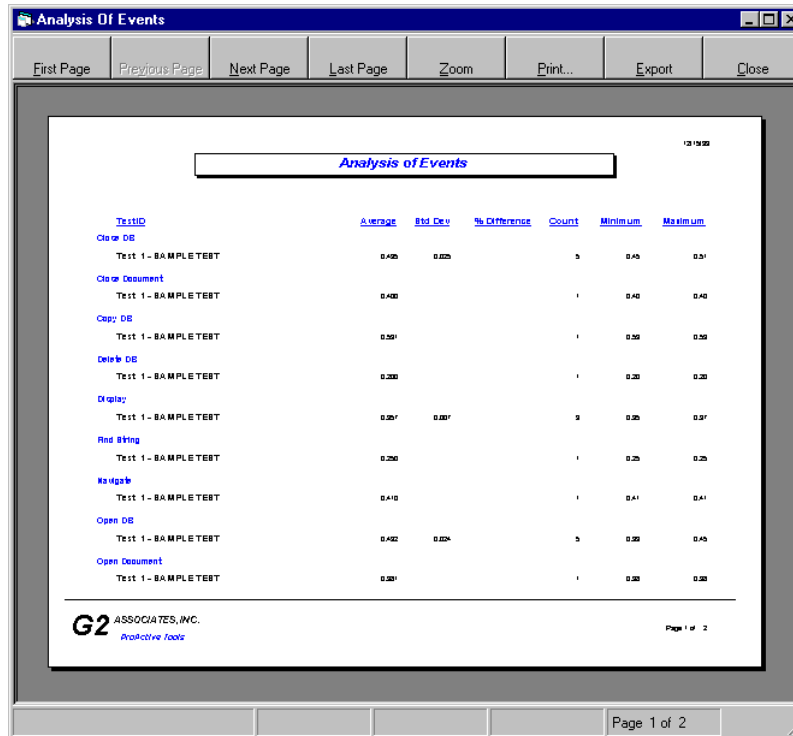


Figure 3-11: Reporter Analysis of Events Report

This report displays the average response time, the standard deviation, % difference, count, and the minimum and maximum response times for all of the seven events that were executed during the running of the “Out of the Box” scenario.

The % difference column normally displays the difference in the average response times for selected tests. It is blank in this case because results for only one test are shown.

Note: A more detailed explanation of the elements of this and other report formats included in ProActive Reporter can be found in Chapter 6.

Chapter 4 – Detailed Structure of Scenario Files

This Chapter will build on the information that has been provided in the manual so far by discussing in some detail the structure and parameters of a test scenario file. If you have followed the progress of the manual, then you have now installed ProActive Tools and have run the “Out of the Box” scenarios. This chapter will deal with each of the following areas:

- Scenario File Overview
- The Sample Scenarios
- Defining Simulation with the Scenario File
- The General Section
- The Activity Sets Section
- The Activity Routines Section
- Product Specific Routines
- Output Files

Scenario File Overview

ProActive Assistant and ProActive Load support a variety of activities that simulate real-world Notes usage. These activities are combined into a scenario parameter file for these products to run. The variety of activities available to the scenario writer allows the testing to be tuned to focus on specific aspects of server performance.

While the structure of the scenario files and the parameters used in them are very similar between ProActive Assistant and ProActive Load, they can never be exactly the same because of the actual functional differences in the products. Due to the User Interface orientation of ProActive Assistant, you are able to navigate through Notes and emulate Notes user activity. ProActive Load is a multi-threaded Notes API based product. Therefore, it is only concerned with the activities on the Domino server and requires parameters for managing the multiple threads. Load is not aware of the Notes UI or any activity that takes place on the Notes Client workstation. This distinction is important to remember when developing test scenarios for each of the products.

The scenario files are composed of three sections. The General Section, the Activities Set Section, and the Activity Routines Section. All three sections must be present in a valid scenario file. In the descriptions that follow, the structure of each of these sections will be covered in detail. The scenario file examples that will be used in this chapter will be primarily those from the sample scenario file shown in Appendix B, PALSsample.ini for ProActive Load and PAASample.ini for ProActive Assistant.

The General Section

The General Sections for both Assistant and Load contain the general scenario parameters for managing the execution of a specific test and the parameters that are default values for all of the Activities. Below is an example of a General Section for ProActive Assistant and ProActive Load, presented side-by-side so that you can easily compare the similarities and the differences.

This example could be considered to be a “complete” General Section for these products since it includes almost all of the parameters that would normally be placed in the General Section of a scenario. While many of these parameters are optional, it is always recommended that they be left in the scenario and just commented out with an apostrophe at the beginning of the line if not required.

Many parameters also have defaults for values not specified in the General Section. If these parameters are not specified in the scenario file, the default values will be used, and they may cause your results to differ from your expectations.

In *figure 4-1* on the following page, the shaded items represent those parameters that are used in both Assistant and Load. Those items that are not shaded are unique to the product indicated.

While there are descriptions of all the parameters that can be used in the General Section contained in Appendix A, some of those parameters require additional explanation. The explanations for these parameters are shown on the following page.

ProActive Assistant	ProActive Load
<div>[General]</div> <div>RunId = PAASample</div> <div>Path = c:\progra~1\patools</div> <div>Client = @notes.ini.keyfilename</div> <div>Server = @Notes.ini.mailserver</div> <div>NotesEXENAME = Notes.exe</div> <div>CloseNotesWhenDone = No</div> <div>MaxWait = 500</div> <div>RunType = Count</div> <div>NumSets = 1</div> <div>SleepTime = 10</div> <div>ResultFile = c:\Work\PAAComp.CSV</div> <div>MailResults = no</div> <div>MailResultsTo = Test UserA01</div> <div>MailDBName = @notes.ini.mailfile</div> <div>MailServer = @notes.ini.mailserver</div> <div>DeleteResultFile = no</div> <div>StartTime = 14:30</div> <div>MaxStartWait = 2</div> <div>TestMode = No</div> <div>Random = Yes</div> <div>DbName = @notes.ini.mailfile</div>	<div>[General]</div> <div>RunID = PALSample</div> <div>Path = c:\progra~1\patools</div> <div>NotesIniPath = c:\winnt</div> <div>Client = @notes.ini.keyfilename</div> <div>Server = @Notes.ini.mailserver</div> <div>NTServerName = G2Server01</div> <div>RunType = Count</div> <div>NumSets = 1</div> <div>SleepTime = 10</div> <div>ResultFile = c:\Work\PALComp.CSV</div> <div>ResultOutput = Session, Sleep, Activity, Event</div> <div>TimeLineFile = c:\Work\PALComp1tl.CSV</div> <div>ErrorFileName = c:\Work\ErrorLog.lst</div> <div>StartTime = 14:30</div> <div>MaxStartWait = 2</div> <div>TestMode</div> <div>Random = Yes</div> <div>ListNoteID = Yes</div> <div>Threads = 5</div> <div>ThreadDelay = 1</div> <div>FirstThread = 1</div> <div>Dbname = mail\tusera<thread>.nsf</div> <div>Thread1 = Activity1</div> <div>Thread2-3 = MailCal</div>

Figure 4-1: Sample General Section

Delayed Start

The Delayed Start feature allows you to start both ProActive Load and ProActive Assistant at a specified time, allowing better synchronization between multiple tests. The parameters used in the General Section for both Assistant and Load are the same and function in the same way.

Two General Section parameters are used for initiating a Delayed Start. The **StartTime** parameter indicates the time the scenario is to start and is entered in a 24 hour format. For example, a start time of 2:30 PM would be indicated as **StartTime=14:30**. The **MaxStartWait** parameter is used with **StartTime**. It is the grace period for the **StartTime** and is expressed as <hours>:<minutes> or <hours>.

In the Delayed Start shown in the General Section above, the **StartTime** equals 14:30 and the **MaxStartWait** equals 2. In this case, the scenario will start execution immediately if initiated any time between 14:30 and 16:30, the 2 hour grace period. If initiated before 14:30, then it will wait until then to start. If initiated after 16:30, it will wait until the next day at 14:30 to start.

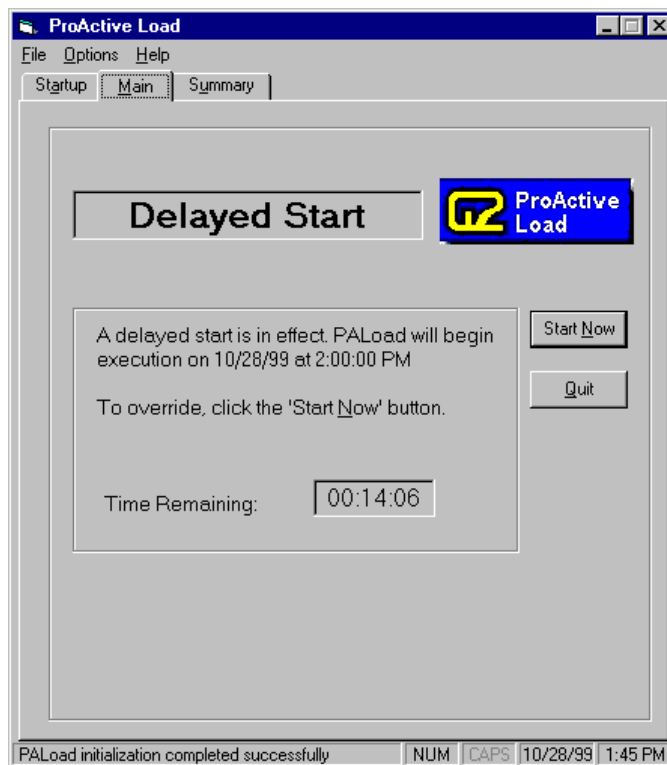


Figure 4-2, ProActive Load Delayed Start

When a scenario is executed in ProActive Assistant and a Delayed Start is designated, Assistant will indicate the start time in the status box at the lower right corner of the screen. With ProActive Load, you will see the screen shown in *figure 4-2* above that indicates the time for the execution to start. From this screen you could also initiate the execution of the scenario immediately, or cancel the Delayed Start.

Timed Versus Count

In ProActive Tools there are two different methods for controlling the number of times each Activity Routine is executed. This is determined with the **RunType** parameter. If the Activity Routine is to be limited by time, then **RunType=Timed** would be used along with **TestDuration=** specifying the number of minutes for the Activity Routine to run. If no TestDuration is specified, the default is 5 minutes. If the Activity Routine is to be limited to a specific number of iterations, then **RunType=Count** would be used along with **NumSets=** to specify the number of iterations. If **NumSets** is not specified, the default value is 5 times. If neither parameter is included in the scenario, then **RunType=Timed** is the default.

Even though the **RunType** is normally specified in the General Section of the scenario, this value can be overridden in any Activity Routine by entering a different value for the **RunType** parameter in that activity. If the **RunType** is not changed in the activity, but the **NumSets** or **TestDuration** is different for the Activity Routine, then the **RunType** does not need to be restated.

Assigning DBName to Specific Threads (ProActive Load Only)

DBName identifies the database that is to be used for any activity. DBName is usually specified in the General Section of the scenario and/or in any individual Activity Routine. A DBName specified in an Activity Routine will have precedence over DBName specified in the [General] section. A DBName specified in an individual Activity will pertain to only that Activity and will not be carried over to any other Activity.

DBNames may be identified for specific threads or range of threads. Adding the thread number or range to the end of the DBName does this.

If mail\tusera.nsf is used for thread number 9, the DBName would be:

DBName9 = mail\tusera.nsf

If mail\tusera.nsf is used for threads 9 through 15, the DBName would be:

DBName9-15 = mail\tusera.nsf

If the same database is used for multiple threads that are not consecutive, the DBName parameter will have to be entered multiple times. If the database mail\tusera.nsf is to be used for thread number 9 and thread number 15, the DBName would be entered as follows:

DBName9 = mail\tusera.nsf

DBName15 = mail\tusera.nsf

If the same databases are to be assigned to the same threads throughout the entire script, it is best to specify them in the [General] Section so they don't have to be specified in each activity.

A variable named "Thread" can be used to substitute the thread number where the variable appears in the script. This feature is especially useful for large numbers of threads using different databases. For example, a scenario with 500 threads, each assigned to different mail databases, can easily be set up as follows:

DBName = mail\tuser<Thread>.nsf

To support the above example, set up 500 mail databases for the scenario to access. If FirstThread is set 1, the databases is named mail\tuser1.nsf, mail\tuser2.nsf, mail\tuser3.nsf,mail\tuser500.nsf.

With thread substitution you can vary the starting thread for additional variation. For example, by setting the **FirstThread** to 201 and the number of threads to 50, the above example would use mail\tuser201.nsf through mail\tuser250.nsf. If running a test on multiple workstations, this would allow you to use the same scenario file and just alter the **FirstThread** on each to use unique mail files on each workstation.

The multiple mail databases required for testing with ProActive Load can be created using the Notes bulk User Registration process or using ProActive Load with its Copy Database Activity. A sample of a Copy Database scenario used for creating mail databases is included in the ProActive Tools Scenario Library Database.

Assigning Activity Routines to Specific Threads (ProActive Load Only)

In the General Section of a ProActive Load scenario, each thread may be assigned to its own Activities Set. If left unassigned, a thread will execute the Activity Routines in the Activities Set assigned by the DefaultActivity parameter. The default for “DefaultActivity” is Activities. In order to invoke thread-specific activities the following commands should be used:

Thread1	= ActivitySet1
Thread6	= Activities
Thread7	= ActivitySet3
Thread3-5	= Activity Set three to five
DefaultActivity	= DefaultAct (default is Activities)

Activity Sets Section

The Activity Sets Section represents the order and timing of the Activity Routines that are to be executed during the running of the test scenario. The Activity Sets Section has the same structure for both Assistant and Load, except that ProActive Load can have multiple Activity Sets that can be assigned to specific threads in the General Section. Since a workstation running ProActive Assistant represents a single Notes session (equivalent to a single thread in ProActive Load), it can have only one Activity Set per scenario. Figure 4-3, below, contains an example of an Activity Sets Section for each product. This example is again taken from the sample scenarios that are shown in Appendix B.

ProActive Assistant	ProActive Load
Activity Sets Section	Activity Sets Section
[Activities] Do 3 Times Call = ReadDocs SleepTime= 5 Call = ChangeViews SleepTime= 5 Call = TestMail End Call = TextIndex SleepTime= 5-10 Call = RebuildViews SleepTime= 5 Call = Find1 SleepTime= 5-10 Call = ComposeDocs SleepTime= 5 Call = ComposePhone Do for 5 minutes SleepTime= 5 Call = Calendar SleepTime= 10 Call = UserReply End	[Activity1] Do for 10 minutes Call = ComposePhone SleepTime= 1-10 Call = RBView SleepTime= 10-20 Call = EditDocs1 End [MailCal] Do 2 Times Call = TestMail SleepTime= 5 Call = Calendar1a Call = Calendar1b Call = Calendar1c SleepTime= 5 Call = Find1 End [Activities] Call = ReadDocs SleepTime= 5 Call = Find1 SleepTime= 5 Call = ComposeDocs SleepTime= 5

Figure 4-3: Sample Activity Sets Section

Note: ProActive Assistant can only have one Activity Set and it is always named [Activities]. ProActive Load, however, can contain multiple Activity Sets, and any name may be used. The name must be enclosed in “[]” and spaces are not recommended.

Parameters Used in the Activity Sets Section

Call =

This parameter indicates the name of the Activity Routine that is to be executed. The value of the Call= parameter must have an Activity Routine in the Activity routines section with the same name. For example, **Call=TestMail** would cause the Activity routine **[TestMail]** to be executed.

IDFile =

This instructs ProActive Assistant (Only) to locate the Notes ID and perform a Switch ID operation.

Location =

This instructs ProActive Assistant (Only) to switch to the named location.

SleepTime =

This specifies the delay in seconds between activities. Specify a number or a range of numbers separated by a dash. If a range is specified, Load or Assistant will wait a random number of seconds, where the random number is within the range.

SleepUntil = NEXT *interval* {ON *minute*}

This instructs the Assistant or Load to delay until the next defined interval.

where:

interval is a value between 1 and 60 denoting the maximum number of minutes to wait

minute is an optional value between 0 and 59 denoting the minute to start (if ON is omitted, it defaults to 0)

for example:

SleepUntil = Next 30

Starts on the next half-hour at 0 and 30 (1:00, 1:30, 2:00, 2:30, etc.)

SleepUntil = Next 30 On 15

Starts on the next half-hour at 15 and 45 (1:15, 1:45, 2:15, 2:45, etc.)

SleepUntil = Next 5 on 3

Starts at 5-minute intervals starting at :03 (1:03, 1:08, 1:13, 1:18, etc.)

SleepUntil = Next 2 on 45

Starts at 2-minute intervals starting at :45 (1:45, 1:47, 1:49, 1:51, etc.)

Note that if the current time is 1:00, it will wait until 1:45, then every 2 minutes thereafter. Once the time reaches :00, it will wait until :45 to start again.

Do Groups

Activities can be repeated by placing them in a Do group. By enveloping one or more activities with any of 4 forms of **Do** and **End**, a set of activities will repeat. Do groups can be nested to any number of levels.

The forms of Do/End are:

1. **Do** <n> {**times**} this set of activities will repeat n times
<n> is a number between 1 and 32767
2. **Do For** <n> {**minutes**} this set of activities will repeat for n minutes
<n> is a number between 1 and 32767
3. **Do Until** <time> this set of activities will repeat until the specified time
<time> is in the form hh:mm AM/PM or hh:mm

4. **Do Forever** this set of activities will repeat indefinitely

If this form is used, the scenario must be terminated manually. ProActive Load is terminated by interrupting the session with the **STOP** button. Once **STOP** is pressed you must wait until Load completes the current activity for each thread. ProActive Assistant is terminated with **Shift-Pause**.

Examples:

Do Forever

Do For 30 minutes

Call = Test Read

Call = Copy

Do 3 Times

Call = Rebuild Views

SleepTime = 10 - 15

Call = Test mail

End

End

Do Until 5:30 PM

Call = Test Views

Call = Send Mail

End

End

Activity Routines Section

The Activity Routines Section is the heart of the scenario file. It contains the details of the Activity Routines that were called in the Activity Sets Section that will actually execute the Notes functions.

ProActive Assistant	ProActive Load
<pre> ***** ' Activity Routines Section ***** [TestCompose] Activity = Compose RunID = Comp DBName = discuss.nsf Server = TestServerA1/G2Test NumSets = 2 Save = Yes Form = MainTopic Text = This is the Subject Text = <TAB 2> Text = This is in the body field [TestMail] Activity = Mail RunID = MailTest Save = Yes Send = Yes SendTo = Test UserA02, Test UserA03 SendSubject = This is the Subject </pre>	<pre> ***** ' Activity Routines Section ***** [TestCompose] Activity = Compose RunID = Comp DBName = discuss.nsf Server = TestServerA1/G2Test NumSets = 2 Save = Yes Form = MainTopic Field = Subject = This is the Subject Field = Body = This is in the body field [TestMail] Activity = Mail RunID = MailTest Save = Yes Send = Yes SendTo = Test UserA02, Test UserA03 SendSubject = This is the Subject </pre>

Figure 4-4: Sample Activity Routines Section

The shaded items in *figure 4-4* above represent parameters that were specified in the General Section, but were changed in the specified activities. If these parameters, as specified in the General Section, were not different for these

activities, they would not need to be repeated in these activities. If they are repeated, but with different values from those contained in the General Section, then the values specified in the Activity Routine override the values specified in the General Section.

ProActive Tools contains eleven predefined activity types for ProActive Assistant and eight for ProActive Load. These predefined activities provide for easy development of Activity Routines for common Notes activities. Both products also support a User Defined Activity Type that can be used to develop Activity Routines for custom databases and applications. A listing of the activity types is shown in *figure 4-5* below.

Activity Type	Description	Used in Assist?	Used in Load?
Calendar	Performs all Notes Calendar functions	Yes	Yes
Compose	Composes a Notes document	Yes	Yes
Copy Database	Copies a Notes database	Yes	Yes
Edit	Edits a Notes document	Yes	Yes
Find	Conducts text search from a view	Yes	Yes
Index	Creates full text index for a Notes database	Yes	No
Mail	Composes mail message using standard Notes memo form	Yes	Yes
Read Documents	Reads Notes documents	Yes	Yes
Rebuild View	Rebuilds Notes view indexes	Yes	Yes
Reset Result	Starts new test session in Result File	Yes	No
User Defined	Used to develop Activity Routines for custom applications. Can perform any of the above activities, but requires all parameters to be specified.	Yes	Yes
Views	Cycles through all views in a database	Yes	No

Figure 4-5: Activity Types

A description of all parameters used for the above activity types is described in detail in Appendix A. The table of contents contains the specific page numbers in Appendix A for each of the activity types.

Structure of an Activity Routine

Depending on what is to be accomplished, an Activity Routine can be very simple or very long and complex. The predefined activities simplify the task of constructing Activity Routines that represent standard Notes functions, such as creating mail messages and reading documents.

Events

As mentioned above, the execution of an Activity Routine simulates Lotus Notes activity. Activities consist of a series of events that make up the activity. For example, the reading of a document may include the following events:

1. Open Database
2. Open View
3. Navigate to Document
4. Open Document
5. Close Document
6. Close Database

ProActive Assistant and ProActive Load execute this same set of events and record them with their associated response times in the Result File.

In Appendix A, the events executed for each Activity Type supported by ProActive Tools are shown preceding the listing and description of the activity parameters.

Sources of other Information

There are several sources to help familiarize you with the structure of effective Activity Routines. These include the Scenario Library provided on the ProActive Tools CD, the sample scenarios in Appendix B, and the samples that follow the parameter descriptions for each of the Activity Types in Appendix A.

Note: If you chose the third installation option, “Select ProActive Tools and Optional Files,” the Scenario Library database was added to your Notes data directory listed in the installation.

Description of Other Features of Activity Routines

SendTo Options

There are two primary methods for specifying SendTo lists to be used in ProActive Tools scenarios for sending mail messages. The most common method is to manually enter the specific names and /or groups into the scenario for addressing the messages:

SendTo = Test UserA10, Test UserA50, Test UserA99

SendTo = Group10, GroupB, Test UserA56

Another method would be to read into the scenario a list of names or groups from a text file using the ProActive Tools @Variable function:

SendTo = @File.txt.SendGroup1

In the case above, an entry in the text file, File.txt, could appear as follows:

SendGroup1 = Test UserA10, Test UserA50, Test UserA99, GroupB

Note: The use of text files allows the assignment and modification of SendTo lists without having to modify the scenario file each time a change is made. A description of the @Variable function follows this section in this chapter.

A third method for developing SendTo lists is the random selection of SendTo addresses from a Notes database. This method is only available for use with ProActive Load. With this method the SendTo addresses are selected from the Name and Address Book (names.nsf) on the mail server or any database that has the NAB design. This method will work equally well for Notes R4 and R5 environments. The syntax for sending messages to 50 randomly selected people in the NAB would be as follows:

SendTo = Names<select 50>

In this case, ProActive Load will assume that the server, where the Names.nsf file is located is the one specified in the "Server=" parameter in the scenario. To use other databases on other servers for selection of the SendTo addresses, the specific server and database are indicated as follows:

SendTo = Server2!!Names9<select 50>, to use Names9.nsf on Server2 or

SendTo = Local!!Names9<select 50> if the Names9.nsf database is on the local workstation.

Note: If the select number is greater than the number of person documents in the database, ProActive Load will default to the number of person documents in the database. This is due to the requirement that ProActive Load not duplicate a user name for any single SendTo list for a specific thread.

The @Variable Function

ProActive Assistant and ProActive Load can substitute parameter values from other files. These external files must be in ASCII format with records in the form of **keyword = value**. To substitute values, use the following format:

@<file name>.{<file extension>}.<keyword>{.}

where:

<file name>	Name of the file which contains the substitutable value
<file extension>	File extension
<keyword>	Keyword in the file
{.}	(optional) a period to separate the value from the remainder of the line the trailing period is only required if additional characters follow the @variable

For example:

Server = @Notes.Ini.MailServer

This instructs ProActive Assistant/Load to find the MailServer keyword in the Notes.Ini file and substitute that value. For example, if the MailServer parameter in the Notes.Ini file is "CN=TestServerA1/O=G2Test", then Assistant/Load will interpret the line above as

Server = TestServerA1/G2Test

Nested Substitution within the @Variable Function

In order to provide for unique substitution of text for each thread in a ProActive Load scenario, you can place the thread substitution variable in the keyword of the @Variable substitution. In the example below, each thread would substitute a different text string from the text file Data.txt.

Text = @Data.txt.field<thread>

If the scenario executed 100 threads, then Data.txt must contain at least 100 entries labeled "field1=" through "field100 =."

Note: When entering an address that contains an @ sign into a SendTo field, you must use two @ signs so that it will not be interpreted as an @Variable. The address "joe.smith@company.com" would be included in the scenario file as "SendTo = joe.smith@@company.com."

Entering Comments into Scenario Files

Comments may be added anywhere in the scenario file. ProActive Assistant/Load ignores characters to the right of the quote character (') placed anywhere on a line unless it is inside a string surrounded by double quotes ("). Additionally, if the character string REM begins any line, the entire line is treated as a comment.

Generalized Thread Substitution (ProActive Load Only)

In an Activity Routine of a ProActive Load scenario, the number of the current thread can be substituted almost anywhere in the scenario where a number would appear to the right of an "=" sign. This is accomplished by replacing a number with the <thread> substitution variable in a manner similar to that used with the DBName command. Some examples of this type of substitution would be as follows:

Examples of numeric thread substitutions:

Navigate	= <thread> (in User Defined activity)
NumberOfDocuments	= <thread> (in Read Documents activity)
NumSets Section)	= <thread> (in any activity and [General]
SleepTime Section)	= <thread> (in any activity and [General]
ReadTime	= <thread> (in Read Documents activity)

Examples of thread substitutions used with text:

Attach	= File <thread>
Subject	= Document created by thread No. <thread>
Text	= Document created by thread No. <thread>
Text	= @Data.txt.field<thread>
Field = Field<thread>	= Entry created by Thread No. <thread>
Server	= TestServer<Thread>
View	= TestViewName<Thread>

Chapter 5 – Building & Running Your Own Test Scenarios

Now that you have installed ProActive Tools, run the “Out of the Box” scenarios, and reviewed the structure of the scenario file, you are ready to start developing a test scenario that can be used for testing in your own environment. Tests developed with ProActive Tools are highly customizable and can be designed to duplicate your own Lotus Notes production environment.

This chapter will help you prepare for, build and run test scenarios.

The topics covered in this chapter are as follows:

- Increasing Your Understanding of ProActive Tools
- Developing a test profile to summarize the requirements of the test before the scenario(s) is developed
- Designing the Test Scenario
- Using the Scenario Wizards to develop test scenarios
- Debugging the Test Scenario
- Running the Test
- Analysis of the Test Results
- Tips and Techniques

Increasing Your Understanding of ProActive Tools

Before you start to develop your own scenarios with ProActive Tools, it is best to experiment with the “Out of the Box” scenarios and the scenarios in the Scenario Library that ship with ProActive Tools. After you have run the “Out of the Box” scenario, you should review its syntax to understand how it functions. You can then modify it by moving the Sample.nsf file to a test server and adding some additional Activity Routines.

You can also update the Sample Scenarios (shown in Appendix B and included in the Scenario Library) to run in your environment. These scenarios each contain eleven different Activity Routines that demonstrate much of the capability of ProActive Tools. Once you have done this, you are probably ready to start developing your own scenarios that will perform the activities you need to test your environment.

Note: ProActive Tools executes Notes functions and adheres to Notes conventions and requirements. You cannot perform activities not allowed by Notes & Domino.

Developing the Test Profile

A test profile is a method of documenting the details of the user activity to be included in the test. An actual test may consist of several scenario files. For any test there will usually be at least two scenario files, one for ProActive Load and one for ProActive Assistant. A test profile should be developed for each of the scenario files (in the case where the Load and Assistant scenario files are to be the same, one test profile is sufficient). While there is no specific format for such profiles, a sample ProActive Load profile is provided in this section as an example.

If the test's focus is the volume generated per user, the profile should be developed on this basis. This profile can represent the volume for the average user, or multiple profiles (and resulting scenarios) can be developed for different types of users, such as heavy, light, and casual. The time period for the profile is usually a peak hour, since this is when performance tends to be a problem.

The next step in writing the profile is to determine the specific activities generated by the users and the volume for each of those activities. As an example, the profile for a messaging server might include the following items (expressed per user for a single peak hour):

- Number of messages sent
- Number of invitations sent
- Size of the average message
- Size of invitations
- Number of attachments sent
- Average size of attachments
- Number of addressees per message/invitation
- Number of documents read
- Documents read and/or created in other databases
- Number of searches

Sample Profile Form

Profile Name:

Percent of test user population		100% if there is only one profile
Percent Out of Office		
Average beginning size of mail files		Needed to configure test setup
Mail file full text indexed (y/n)		Impacts required disk space

Document counts (per peak hour)

Messages sent	
Invitations sent with Free Time Search	
Messages read	
Messages sent with attachments	
Invitations sent with attachments	
Searches	
Documents created ("other" database)	
Documents read ("other" database)	

Document sizes

Average text size of messages (kb)	
Average attachment size (mb)	

Other items

Number of addressees per message	
Number of addressees per invitation	
Name of "other" database	
Name of "searched" database	

Figure 5-1, Volume Profile

Once the profile information has been defined, the scenarios for Assistant and Load can be developed.

Designing the Test Scenario

The user profile, as well as the testing principles discussed in Chapter 1, is the basis for designing the test scenario.

An important part of the design process is remembering the question to be answered. Different questions will lead to different approaches in designing the test scenarios. For example, determining the capacity of a mail server will require significantly different scenarios than analyzing the performance of a custom Notes application. The general guidelines for planning and executing performance tests apply, but the tests themselves will be different.

ProActive Assistant and ProActive Load are used together in most types of performance testing. ProActive Assistant can interface with a variety of Notes applications via the Notes UI, and capture user response times for a variety of Notes events. Assistant scenarios are typically designed to recreate a specific set of user activities. ProActive Load can simulate different numbers of users on a server, also interacting with a variety of Notes applications. Load scenarios are typically designed to emulate a specific user population, or simply to create different amounts of load on a server.

There are a number of aids available to speed up the scenario design process. The ProActive Tools scenario library (a Notes database) is included with the software install, and a copy of this library is also in the documentation directory on the CD. This library contains sample scenarios that can be edited to customize them for your environment. As part of our product support, G2 Associates can also provide assistance in designing tests for your environment.

ProActive Assistant

In any test, user response times are the most important element of performance, and ProActive Assistant captures these. ProActive Assistant can either be used to measure general response times, such as document and database open times, or it can be used to measure the response times for specific parts of a Notes application. In either case, Assistant will see the same response times a user would see. It can be used stand alone with an unloaded server, with a production server, or together with Load against a test server.

ProActive Assistant scenarios are designed to mimic what users actually do. Since Assistant works with the Notes UI, scenarios can be written to use almost any Notes application in the same way that users do.

The basic steps for developing an Assistant scenario are:

Review the user profile. In general, the design of the test scenario is based on the user profile.

Develop the basic activities:

- Standard activities such as mail and read documents only require basic parameters, such as the server, database name, recipients, and document text.
- For custom applications, you will usually use the user-defined activity. User-defined activities provide more control, but a little more effort to develop scripts. User-defined activities generally recreate a series of keystrokes that a user might enter to walk through the application. Activities such as pressing action buttons and selecting entries from dialogue boxes are all supported.

After the basic activities are written, you should determine the length of the test. Scenarios can be written to run for a single iteration, or for a specific period of time. Based on this decision, you can set up sleep-times within and between activities, as well as do-loops to control the iteration through activities. Appendix B contains sample scenarios for Assistant and Load.

After the basic scenario is completed you can begin running and debugging the scenario.

ProActive Load

Since users seldom get their own dedicated servers, a realistic performance test should measure response times for a single user, as well as when the server is loaded. ProActive Load lets you write scenarios to create the load of multiple users against a test server. And Load will let you create exactly the type of user load you want for your performance tests.

Keep in mind when designing Load scenarios that tests can be run with different numbers of threads, allowing comparison of results at different load levels.

Load scenarios are very similar to Assistant's, in terms of both capability and syntax. The basic steps for developing a Load scenario are:

Review the user profile.

Develop the basic activities:

- Standard activities such as mail and read documents only require basic parameters, such as the server, database name, recipients, and document text.
- For custom applications, you will usually use the user-defined activity. User-defined in Load is different than in Assistant. Since Load works with the Notes API, keystrokes are not particularly relevant. Rather, user defined in Load is used to create and update specific types of documents in custom Notes applications. User defined also has the ability to emulate other types of transactions that might occur between the Notes client and server when a document is being created, such as DBLookup. Determining exactly how to write a user-defined activity for a complex Notes application will usually require some detailed analysis of the code and behavior within the Notes application.

The other thing to keep in mind when developing Load scripts is that you can run multiple threads, and different threads can work differently. The `<thread>` operand can be used within activities to provide differentiation. Also, different activity sets can be defined for different threads or groups of threads.

After the basic activities are in place, you should determine the length of the test, and the mix of activities to be performed by each thread. You also need to consider the ramp-up period for Load. If hundreds of threads are initiating sessions with a Domino server, performance may not be stable initially. We often start Load scenarios with a relatively long random SleepTime before the actual activities start to compensate for this.

After the basic scenario is completed you can begin running and debugging the scenario. Each activity set in a Load scenario should be run and verified.

Using the Scenario Wizards

ProActive Tools provides Scenario Wizards to assist in the development of the test scenarios. These Wizards (one each for ProActive Assistant and ProActive Load) are Notes databases that will step you through the development of a test scenario by providing help and guidance in constructing the Activity Routines and the other sections of a valid scenario file.

Details for the use of the Wizards are covered in Chapter 7.

Debugging the Test Scenario

Before the test scenarios are ready to be used for an actual test, they must be debugged to insure that they are working properly and all of the databases, views, and other design elements required for the test are available.

ProActive Assistant Scenarios

It is usually straightforward to debug ProActive Assistant scenarios since Assistant uses the Notes UI and you actually watch the execution of the scenario. During the debugging process you can stop the execution of the scenario and correct any error conditions that might exist.

You can put Assistant into Test Mode for the execution of the scenario. Test Mode will step through the individual events and stop after the execution of each event with a dialogue box that describes the event. Test Mode can be turned on for a single execution of the scenario from the Modify screen, see *figure 5-2*.

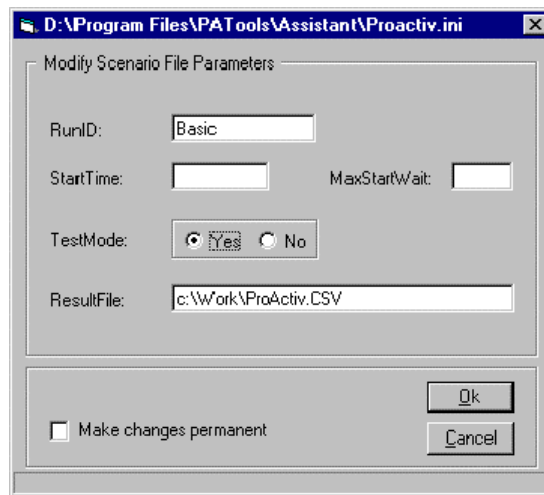


Figure 5-2, Assistant Modify Scenario Parameters

The ProActive Assistant result file is also useful for debugging scenarios. If any errors occurred during the execution of the scenario, Assistant will write an error message to the comment field of the results file. You can then match the event where the error occurred with the scenario file to determine the reason for the error condition.

The final method is to review the databases that were changed as a result of the execution of the scenario and verify that the expected actions did take place.

ProActive Load Scenarios

Debugging ProActive Load Scenarios require a different approach since the Notes UI is not available to view the scenario while it is running. Test Mode in ProActive Load will run the scenario one time ignoring all loops, but does not pause after the execution of each event as it does in Assistant. Test Mode can be activated for a single execution of the scenario from the Modify screen, as shown in *figure 5-3*.

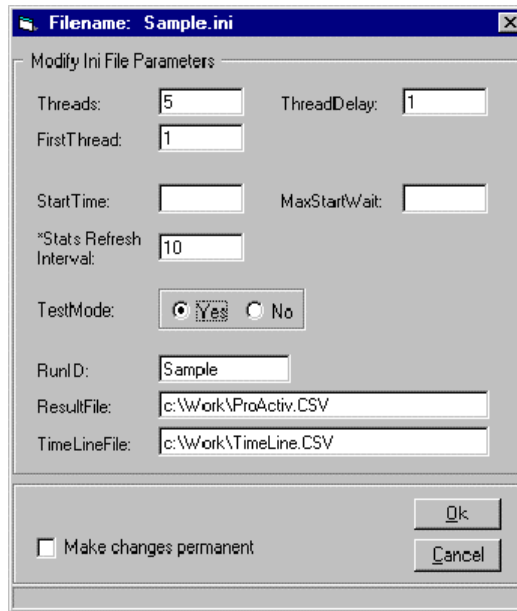


Figure 5-3: Load Modify Scenario Parameters

The most effective method for debugging ProActive Load scenarios is to execute each Activity Routine with a single thread and then review the result file and the changes in the affected database. Debugging Load using the results file works best when only a few threads are run. Otherwise, the results file will be too large for any effective analysis. The error log (designated in the General Section) will also report errors, especially Notes errors, such as ACL problems, missing databases or servers that can't be accessed. This error log can be reviewed at the completion of a test run.

The **ResultOutput** parameter for Load can be set to control the amount of data written to the results file. For debugging you may want to use the default (Session, Sleep, Activity, Event). For running production tests with hundreds of threads, the only record types needed may be Session and Activity.

Running the Test Scenario

Now that you have developed and debugged the scenario files it is time to run the actual tests. All of the ProActive Load workstations can be set to start at the same time, with the ProActive Assistant workstation(s) set to start after the ramp up period.

Test Duration

Tests are usually run for at least an hour, excluding the ramp-up time needed for Load. Tests may be run for much longer periods of time if needed to answer the performance question.

A normal test usually consists of a minimum of two runs. The tests are run twice and the results compared to verify that they are consistent, and therefore, that the test is valid. If the results are not consistent, you should investigate the test environment to insure that some factor external to the test is not impacting the results. All tests should be reproducible and produce consistent results each time.

Workstation Overload

When running large volume tests involving hundreds of concurrent users, and therefore, hundreds of ProActive Load threads, you will need to verify that the workstation being used in the testing can handle the projected number of threads. If the workstation becomes overloaded it will not generate the expected volume. Test the workstation by running ProActive Load at the maximum volume required during the testing, and then observe its performance.

The easiest way to detect a volume problem on a ProActive Load workstation is to check the workstation performance counters during the test. The committed bytes of memory, paging rate and CPU usage are displayed in the Local Workstation Performance Counters on the ProActive Load UI. There are no specific rules on what these values should be, but we use the following as guidelines:

- CPU should not exceed 70%.
- Paging should not exceed 10 per second.
- Committed bytes should not exceed twice the physical memory.

If any of these values are exceeded, the workstation may be running too many Load threads for an accurate test. You can also run NT Performance Monitor on the workstation during the trial run of the ProActive Load scenario. This will provide a more comprehensive measure of workstation performance.

Analysis of Test Results

ProActive Assistant and ProActive Load generate a comma separated variable (CSV) Result File that is used to analyze the results of the tests. The name and location of these files is specified in the General Section of the scenario file with the parameter **ResultFile=**. The Result file is a record of each event executed by the scenario and the resulting response time for that event. The structure of Result File is shown in Appendix D.

The primary purpose of the Result File generated by ProActive Assistant is to be used as the input to the ProActive Reporter. ProActive Reporter is used to generate response time analyses and reports from the results of the ProActive Assistant testing. See Chapters 3 and 6 for more information on the use of ProActive Reporter.

Another use of the Result File is for troubleshooting purposes. Since the Result File records every event that is executed, the file can be reviewed to verify the proper execution of the scenario file. Assistant and Load also put comments and error messages in the comment field, which can be used to isolate any error conditions. Since the response times generated by ProActive Load are the result of API calls, they are not user response times and not valid for response time analysis.

ProActive Load also generates a TimeLine File that records the progress counters from the Load User Interface. This information is collected at one-minute intervals. The format of the TimeLine File is also described in Appendix D.

<p>Note: Result Files can be viewed with either a text editor or a spreadsheet application but modifications should be done with a text editor.</p>
--

Tips and Techniques

Before running the scenario file, log on to Lotus Notes and perform similar activities to those to be performed by the ProActive Assistant or ProActive Load scenarios. Make sure that all databases, views, etc. are available and can be opened by the workstation(s) to be used in the testing.

After preparing the scenario file, test each activity by specifying TestMode = Yes in the General section or modify screen. In ProActive Assistant, each activity will run once (or for one minute). Assistant will stop at the end of each event and permit you to terminate the test. To validate each activity, observe the Assistant executions, then review the results file.

For ProActive Load, Test Mode will run through the scenario ignoring all loops and SleepTimes. For best result, it is recommended that Load scenarios be tested with a similar Assistant scenario.

Most of the predefined activities that are provided with ProActive Tools have been designed to allow for the development of test scenarios that are easily designed and used. They provide for most of the standard actions that would be performed by a normal Notes client. For more sophisticated testing with custom developed Notes applications, ProActive Tools provides a User Defined Activity that is more adaptable to special situations. Developing scenarios with the User Defined activity type will require a much better understanding of the application to be tested. Before moving on to the more sophisticated testing with the User Defined Activity, it is desirable to gain a working familiarity with the predefined activities and how they function.

ProActive Assistant and ProActive Load only support the opening of one database at a time when using the predefined activities. The User Defined Activity can be used in ProActive Assistant to open multiple databases by stepping through the Notes UI and pressing buttons, executing actions, and selecting appropriate menu options.

With both ProActive Assistant and ProActive Load, you can execute multiple predefined activities while keeping a single database open. This can be used to simulate what a user might do when reading mail, sending mail, and creating calendar documents while keeping the mail file open. This is done with an initial User Defined activity that opens a database, but does not close it before other activities are executed. The open and close database events in these activities will be ignored until the database is closed with another User Defined activity.

ProActive Assistant does not support Load Balancing and Failover in a Notes Clustered environment. It does not have the ability to change servers when the current one fails. ProActive Load will failover to another server under these circumstances, but there is probably little need to do so. We generally feel that an appropriate test would be to measure the impact of the Notes/Domino environment after the failover has occurred. Failing over during a test will not prove much other than invalidating the test. For more details on this issue please contact G2 Associates.

If a user ID is password protected, ProActive Assistant will stop at the password prompt if the Password = parameter is not supplied in the General section. Once the password is entered, remaining activities will continue normally.

Use @variables (for example, @NOTES.INI.Mailfile) to substitute different information for separate runs of the scenario file without having to rewrite the scenario file.

When running ProActive Assistant, disable the screen savers and other applications that could execute unexpectedly on the workstation. These applications can disrupt ProActive Assistant's execution and produce unexpected results. Also disable type-ahead and mail alerting in the Notes Client.

ProActive Assistant functions by executing same keystrokes that a normal Notes client would. Assistant will not function with Notes navigators. You must be sure that Forms, Views, and other required menu options have been added to the Notes database to be tested, so they can be accessed via the keyboard.

You can terminate a ProActive Assistant session at any time by pressing **SHIFT + PAUSE**.

You can terminate a ProActive Load scenario at any time by pressing the Stop button. Once the button is pressed, wait until Load completes the currently running activities.

ProActive Assistant can be initiated from a command line and bypassing the frontend by using the following syntax.

```
[<drive>:\<dir>] MTRUN.EXE [<drive>:\<dir>] PROACT32.PCD /C {PROACTIVE.INI}
```

An example would be as follows:

```
c:\PATools\Assistant\mtrun.exe c:\PATools\Assistant\proact32.pcd /C  
c:\PATools\proactiv.ini
```

Chapter 6 – The ProActive Reporter

This chapter describes the details of the ProActive Reporter and includes the following major sections

- ProActive Reporter Overview.
- Importing CSV Result files.
- Assignment of Test Names and Descriptions.
- Selection of Tests to be analyzed.
- Predefined Report Formats.
- Using the Report Wizard.
- Deletion of Tests from the Reporter Test Database.

ProActive Reporter Overview

The ProActive Tools Reporter is a reporting tool for analyzing ProActive Assistant results files. The location and name of the result file is specified in the General Section in the Result File Parameter. Any valid path and filename maybe used. By default, result files are located in the C:\Work\. See Chapter 3 for details about running the Reporter.

The Reporter produces seven different pre-defined reports. The Reporter also enables you to create custom reports. Each report provides different levels of detail and comparison information for some or all of the tests in the results database.

Each ProActive Simulator workstation generates a results file that may be written to disk or sent via Notes to a common user. These results files are in a Comma Separated Variables (.CSV) format, and can be imported directly into the results database by the Reporter.

Reporter Main Screen

From the Main Screen of ProActive Reporter the four tabs describe the four primary sections of the Reporter. These provide for the importing of ProActive Assistant result files, selecting tests for analysis and reporting, developing reports, and managing the tests in the ProActive Reporter Database.

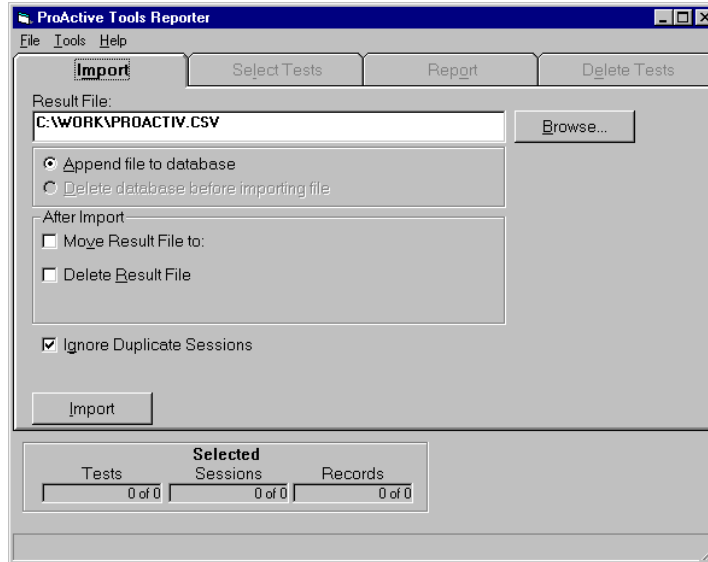


Figure 6-1: ProActive Reporter Main Screen

The **File** menu selection provides a method to select **Import**, **Select Tests**, **Report**, **Delete Tests**, or **Exit**. The **File** menu may be accessed using the mouse or by typing **ALT-F**. The functionality of each selection is described below.

The **Tools** menu selection provides a method to **Change Test Names** or **Compress Database**. The **Tools** menu may be accessed using the mouse or by typing **ALT-T**.

The **Help** menu contains two selections: **Contents and About ProActive Tools Reporter**. The **Help** menu may be accessed using the mouse or by typing **ALT-H**.

Importing Result Files

To create a report, you must first import the results(C:\work\Proactive.csv by default) data into the Reporter's database. All reporting functions draw upon this database for information. During the import procedure, a status message and a status bar appear on the bottom of the screen to show the current state of the import process. Multiple tests may be imported.

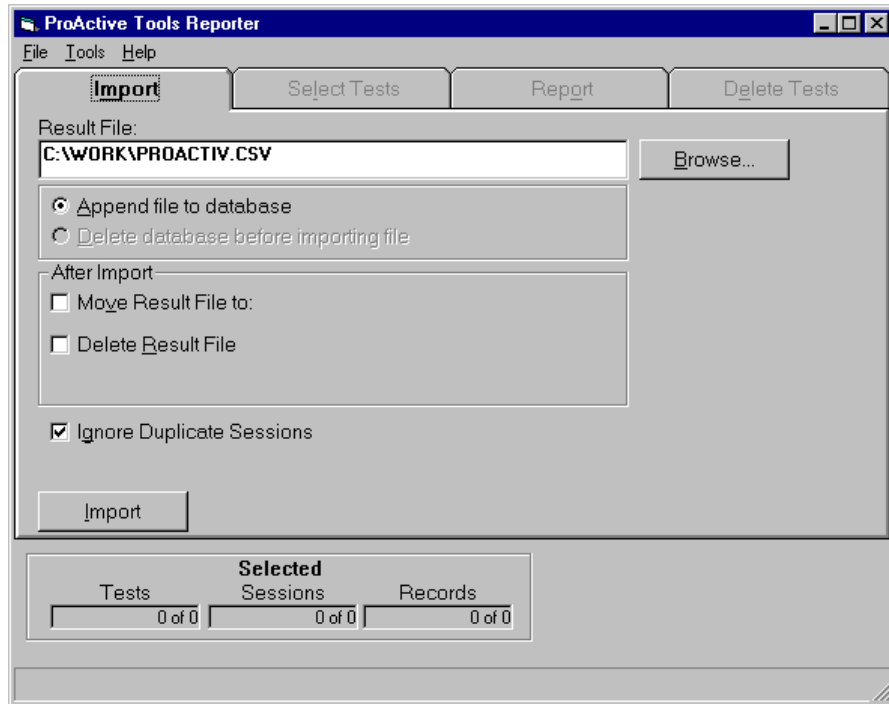


Figure 6-2: Importing Reporter Data

Use the controls on the **Import** folder to transfer ProActive Assistant result files into the Reporter database.

Result File: Type the fully qualified name (drive:\directory\...\filename) of the result file to be imported in the space provided. Optionally, use the **Browse** button to find the desired file.

Append file to database: By selecting this radio button, you tell the Reporter to add the new data to any existing information in the database.

Delete database before importing data: By selecting this option, you tell the Reporter to clean out all existing records in the database before importing the new data.

Import: Selecting this button initiates the import of data from the ProActive Assistant results file (CSV file) into the ProActive Reporter database.

After Import: After the data is imported to the Reporter database you may move the result file (the file that was imported, not the Reporter database file) to a specified location or instruct the Reporter to delete the results file.

When you select **Moye Result File to:** a dialogue is opened to allow you to enter the fully qualified location or you may navigate throughout your system to select a location.

When you select **Delete Result File** the result file will be erased after the import function is completed.

The **Selected** (Tests, Sessions, Records) window displays summary information on the number of tests, sessions, and records in the Reporter database.

The **Import** folder displays the current selection status of tests, sessions, and records to be used in the creation of reports as a subset of the total database. This display is shown on the **Import**, **Select Tests**, and **Report** folders. On the **Delete Tests** folder, this display indicates the number of tests, sessions, and records currently selected to be deleted from the database. Refer to Deleting Test Files from Reporter Database at the end of this chapter for more information.

Assigning Test Names

After the import procedure is completed, the **Test Names/Descriptions** dialogue is presented giving the user the opportunity to provide a name and description for each new test. The dialogue box shows the date and time for the start and stop of each test created from the results file, as well as the count of sessions and records for each test.

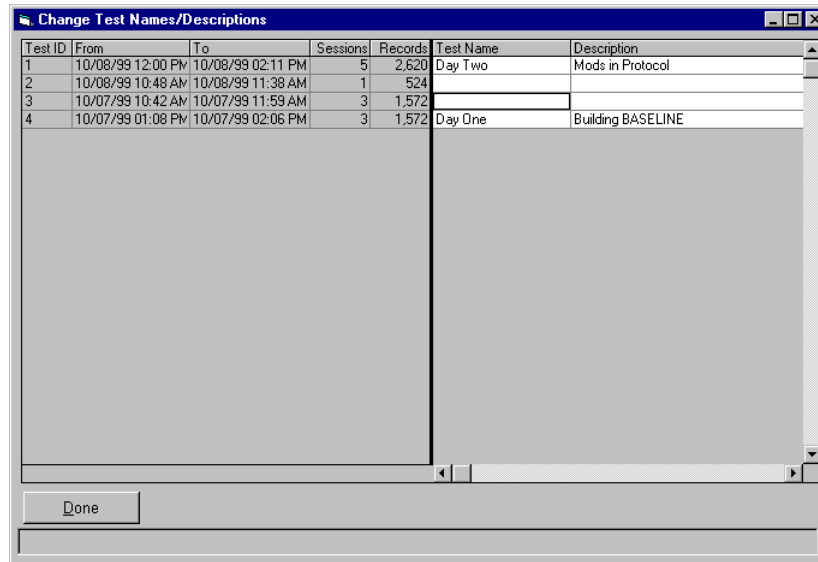


Figure 6-3: Test Names Dialogue Box

If this dialogue is invoked from the menu (**Tools/Change Test Names**), all of the tests in the database are displayed in the list, allowing test names and descriptions to be changed or added for any test.

Selecting Test Results

The Select Tests folder displays all the tests contained in the Reporter database (see Importing Result Files earlier in this chapter for instructions on importing ProActive Assistant results into the Reporter database) and allows you to choose tests to be included when creating a report. This folder displays all tests in the database, the optional test name, the time frame that the test executed (From, To), and the optional description.

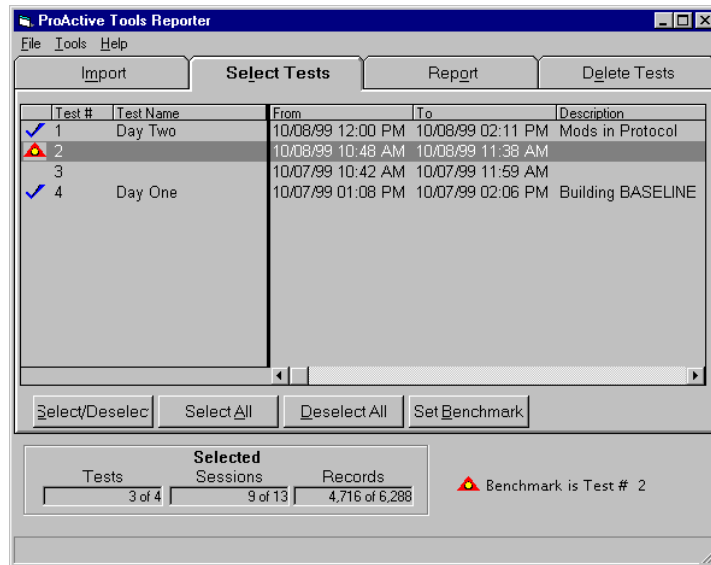


Figure 6-4: Select Tests

Selecting Individual Tests

Select/deselect by clicking on the test using the left mouse button or highlighting the test and clicking the **Select/Deselect** button (or pressing **ALT-S**). When a test is selected, a checkmark appears to the left of the test number.

Selecting/Deselecting All Tests

Select/deselect all tests by clicking on the **Select All** (or pressing **ALT-A**) or **Deselect All** (or pressing **ALT-D**) buttons. Pressing either button will clear the current benchmark (described below).

Setting a Benchmark

Optionally, select one test that the Reporter will use as a baseline to compare with all other selected tests (reports show a % Difference column that reflects the difference between response time averages for a given test and the Benchmark test). Select the benchmark by highlighting the test you would like to use as the Benchmark and click the **Set Benchmark** button (or press **ALT-B**). A triangular icon will appear next to the selected Benchmark test. If a Benchmark test is not specified, the first test included in the report will be used as the base for comparing other tests in the % Difference column.

Current Selections and Database Counts

The number of tests selected and the current Benchmark are displayed in the lower left corner of the folder. The total number of Tests, Sessions, and Records contained in the Reporter database are also displayed.

Standard Report Selection

The Report folder provides access to the standard reports included with the ProActive Tools Reporter and the Report Wizard. ProActive Reporter currently provides four analysis and three list reports in addition to the report formats that can be designed using the Report Wizard.

The Analysis Report Type provides for Analysis of Activities, Activities by Client, Events, and Events by Client. The List Report Type will show the list of Events, Sessions, and Tests.

To select a report, highlight the Report Type and then the Report Name. Reports may be displayed on the screen or sent directly to the printer. The Report folder also shows how many tests, sessions, and records are selected (see Select Tests) and the current Benchmark.

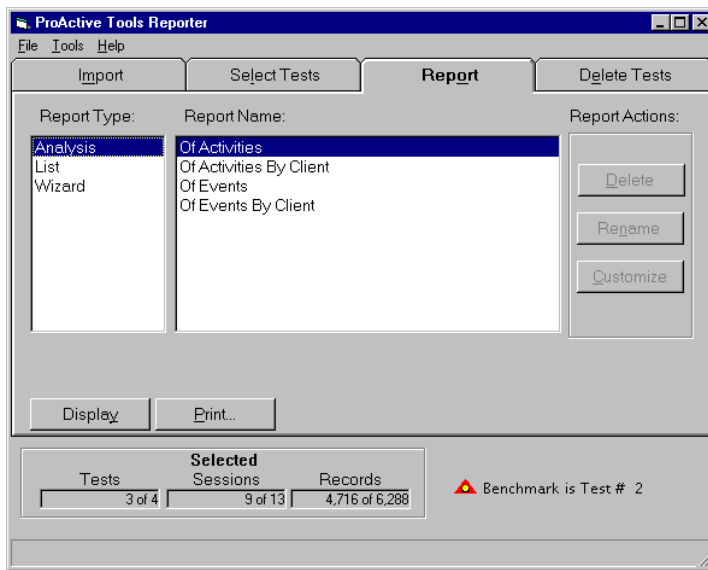


Figure 6-5: Select Reports

Report Layout

When you choose to display and/or print a report, the report will be displayed in a Print Preview window. Buttons across the top of the window provide functionality in the report display mode. To return to the main Reporter window, click the Close button.

The following is a description of buttons that appear at the top of all reports:

<u>F</u>irst Page	Returns to display the first page of a report.
<u>P</u>revious Page	Displays the previous page of a report.
<u>N</u>ext Page	Displays the next page of a report.
<u>L</u>ast Page	Displays the last page of a report.
<u>Z</u>oom	The initial report display shows a subset of the entire report on the screen (the full page does not fit into the report display window). Selecting Zoom allows you to increase and decrease the size of a report display. Window scroll bars (horizontal and vertical) will appear to aid in navigating.
<u>P</u>rint	Invokes the print function to send the currently displayed report to the printer.
<u>E</u>xport	Allows the user to export the displayed report to a separate file. Data formats include: <ul style="list-style-type: none">• Character Separate Variables (user will be prompted to select separation characters)• Comma Separated Variables (.CSV)• Data Interchange Format (DIF)• Excel .XLS (2.1, 3.0, 4.0, 5.0)• Lotus 1-2-3 (WK1, WK3, WK5)• Paginated Text• Record Style (columns of values)• Tab-Separated Text• Tab-Separated Values• Text Prompts will appear to allow the selection of Number and Data formatting and the naming of the export file and its destination.
<u>C</u>lose	Returns to the Report folder.

Figure 6-6: Buttons on Reports

During the description of the actual report formats, the following common terms will be used:

Activity	ProActive Tools PreDefined activities
Event	Lotus Notes events
Client	Shows the individual client designated that ran the test.
TestID	Shows the test ID number automatically assigned by ProActive Reporter and the Test Name of the test being displayed
Average	The average response time (in seconds) for all the occurrences of this Activity or Event within each test.
StdDev	Shows the Standard Deviation for all occurrences of the Activity or Event within each test.
% Difference	Shows the percentage difference of the average response time as compared with the test designated as the Benchmark. In cases where the Benchmark test has not been designated, the report considers the first test to be the benchmark.
Count	Shows the number of times an Activity or Event was executed during the specified test.
Minimum	Shows the minimum response time for a given Activity or Event within a test.
Maximum	Shows the maximum response time (in seconds) for any given Activity or Event within a test
Iter.	The iteration of the item listed.
Start Date Time	Shows the date and time that the Session began.
End Date Time	Shows the date and time that the Session ended.
Status	Shows the status of the given Session (Complete or Incomplete).
Processor/OS/ Memory	Shows the CPU, operating system and the amount of memory for the workstation where the Simulator was executing.
Notes Version	Shows the release level for the Notes software the Simulator was executing.

Figure 6-7: Report Format Terms

A page count appears in the lower right corner for the displayed report.

Analysis of Activities

This report provides statistics for each activity with the individual tests listed under the title for the individual activities.

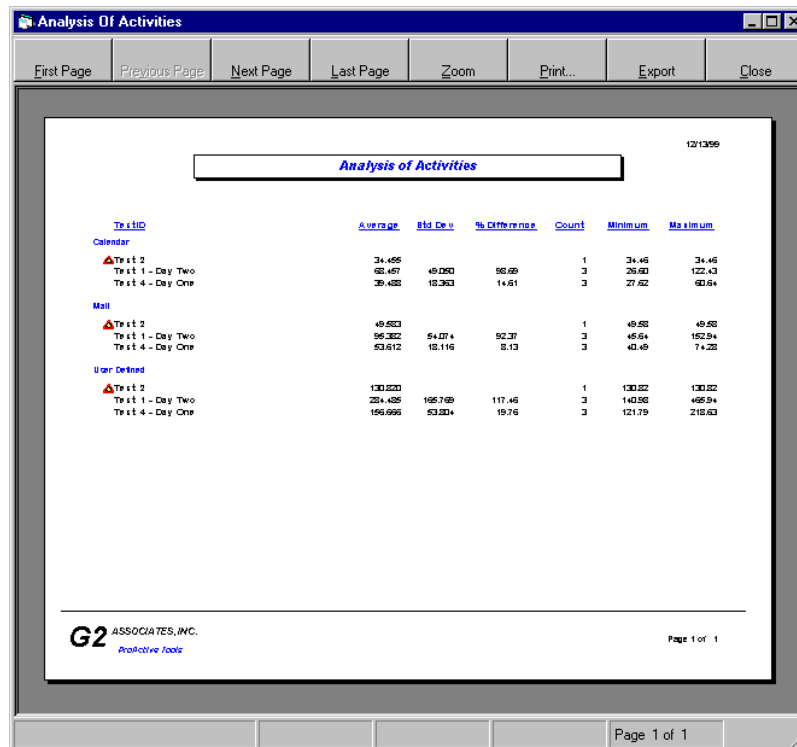


Figure 6-8: Analysis of Activities Report

Analysis of Activities by Client

This report provides statistics for each activity, broken down by individual workstation. It contains the same fields as the Analysis of Activities and adds the Client field to enable the identification of a workstation upon which the Simulator was run. The statistics for each client are provided on a separate line.

Analysis Of Activities By Client						
<div> First Page Previous Page Next Page Last Page Zoom Print... Export Close </div>						
<div> <div>Analysis of Activities by Client</div> <div>12/13/99</div> </div>						
Test ID	Client	Average	Min. Dev	% Difference	Count	Minimum
Calendar						
▲ Test 2		34.466			1	34.46
	Wkst02-64m b	34.466			1	34.46
Test 1 - Day Time		68.467	49.090	50.00	3	26.60
	Wkst01-120m b	122.430			1	122.43
	Wkst02-64m b	26.601			1	26.60
	Wkst03-120m b	56.340			1	56.34
Test 4 - Day One		38.466	18.363	14.61	3	27.62
	Wkst01-120m b	40.639			1	40.64
	Wkst02-64m b	30.204			1	30.20
	Wkst03-120m b	27.621			1	27.62
Mail						
▲ Test 2		49.583			1	49.58
	Wkst02-64m b	49.583			1	49.58
Test 1 - Day Time		95.282	54.074	50.37	3	46.64
	Wkst01-120m b	152.294			1	152.29
	Wkst02-64m b	46.640			1	46.64
	Wkst03-120m b	57.948			1	57.95
Test 4 - Day One		53.612	18.116	8.13	3	40.49
	Wkst01-120m b	74.281			1	74.28
	Wkst02-64m b	46.066			1	46.07
	Wkst03-120m b	40.490			1	40.49
User Defined						
▲ Test 2		130.820			1	130.82
	Wkst02-64m b	130.820			1	130.82
<div> G2 ASSOCIATES, INC. </div> <div> ProActive Tools </div>						
						Page 1 of 2

Figure 6-9: Analysis of Activities by Client Report

Analysis of Events

This report compares statistics for events within each activity.

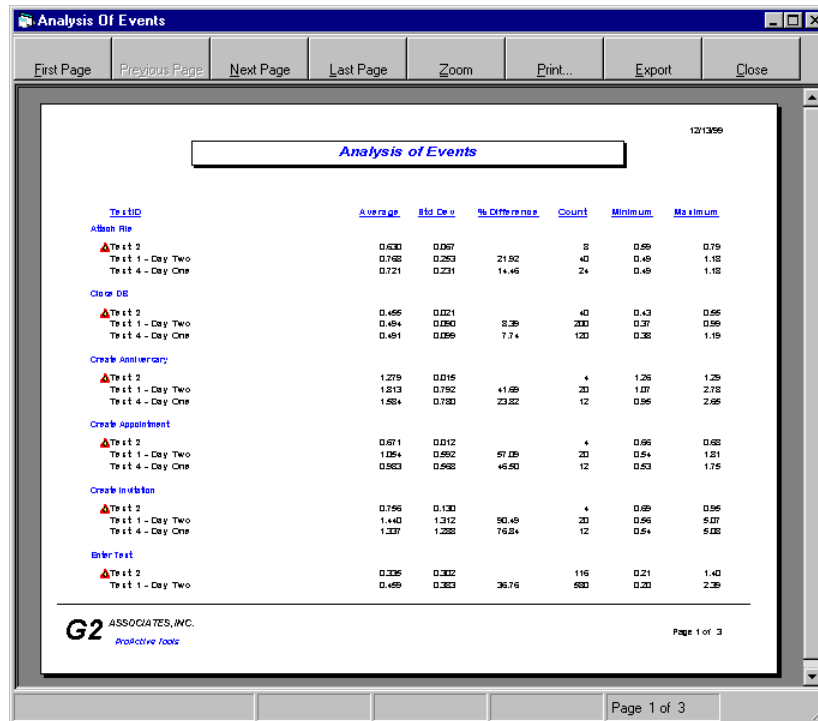


Figure 6-10: Analysis of Events Report

List of Events

This report is a listing of each record in the database. The records are presented in chronological order within client within test.

Activity	Event	RunID	Date	Time	Client	Server	Iter	Response
User Defined	Open D-B	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	4286
User Defined	Select View	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	180
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	441
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	431
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	2543
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	240
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	400
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	300
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	221
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	401
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	310
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	230
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	400
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	310
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	221
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	411
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	310
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	230
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	411
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	340
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	230
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	411
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	300
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	230
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	401
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	310
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	211
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	410
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	321
User Defined	Enter Text	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	230
User Defined	Mail Read 1	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	411
User Defined	Open Document	Mail Read 1	1999-10-03	AM	Uko v42-6mb	Test ServerA1	1	300

Figure 6-11: List of Events Report

List of Sessions

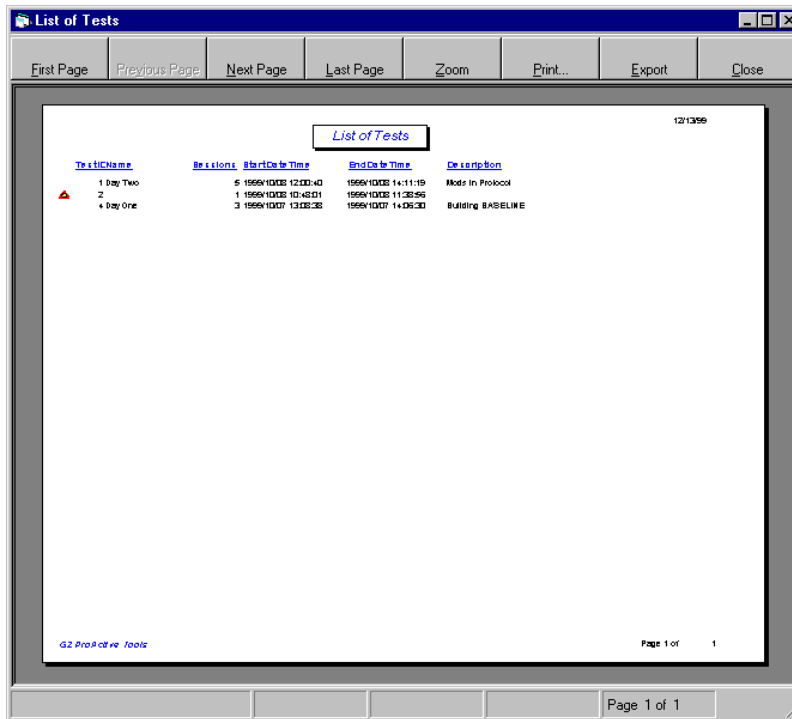
This report provides information for each Session within a Test broken down by Client.

List of Sessions						
Client	Start Date & Time	End Date & Time	Status	Process Location/Memory	Notes & Version	
Test 1 1 Day Two						12/13/99
Wks1-Wk3-123mb	1999/10/03 10:48:30.00	1999/10/03 11:29:25.00	Complete	Penium Windows NT Version 4.0 130,432 KB	Notes R4/Release 4.6.5 June 16, 1999	
Wks1-Wk3-123mb	1999/10/03 12:02:45.00	1999/10/03 12:52:44.00	Complete	Penium Windows NT Version 4.0 130,432 KB	Notes R4/Release 4.6.5 June 16, 1999	
Wks1-Wk3-123mb	1999/10/03 10:48:00.00	1999/10/03 12:51:43.00	Complete	Penium Windows NT Version 4.0 32,193 KB	Notes R4/Release 4.6.2 July 23, 1999	
Wks1-Wk3-123mb	1999/10/03 12:01:00.00	1999/10/03 14:11:19.00	Complete	Penium Windows NT Version 4.0 32,193 KB	Notes R4/Release 4.6.2 July 23, 1999	
Wks1-Wk3-4mb	1999/10/03 12:00:40.00	1999/10/03 12:48:45.00	Complete	Penium Windows NT Version 4.0 6+262 KB	Notes R4/Release 4.6.5 June 16, 1999	
Test 1 2 Wks1-Wk3-4mb	1999/10/03 10:48:01.00	1999/10/03 11:30:56.00	Complete	Penium Windows NT Version 4.0 6+262 KB	Notes R4/Release 4.6.5 June 16, 1999	
Test 1 4 Day One						
Wks1-Wk3-123mb	1999/10/07 13:13:54.00	1999/10/07 14:03:55.00	Complete	Penium Windows NT Version 4.0 130,432 KB	Notes R4/Release 4.6.2 July 23, 1999	
Wks1-Wk3-123mb	1999/10/07 13:12:34.00	1999/10/07 14:06:30.00	Complete	Penium Windows NT Version 4.0 32,193 KB	Notes R4/Release 4.6.2 July 23, 1999	
Wks1-Wk3-4mb	1999/10/07 13:08:38.00	1999/10/07 13:58:39.00	Complete	Penium Windows NT Version 4.0 6+262 KB	Notes R4/Release 4.6.5 June 16, 1999	
Q2 Pro# 0000 Tools						Page 1 of 1
						Page 1 of 1

Figure 6-12: List of Sessions Report

List of Tests

This report provides a listing of the tests in the database. For this report, it is especially useful if a description has been entered for each of the tests in the Reporter database.



The screenshot shows a window titled "List of Tests" with a toolbar at the top containing buttons for "First Page", "Previous Page", "Next Page", "Last Page", "Zoom", "Print...", "Export", and "Close". The main content area displays a table with the following data:

TestName	Sessions	StartTime	EndTime	Description
1 Day Two	5	1999/10/08 12:00:40	1999/10/08 14:11:19	Mode In Protocol
2	1	1999/10/08 10:48:01	1999/10/08 11:28:56	
4 Day One	3	1999/10/07 13:00:38	1999/10/07 14:09:30	Building BASELINE

At the bottom left of the window is the text "G2 ProActive Tools". At the bottom right, it says "Page 1 of 1".

Figure 6-13: List of Tests Report

Developing Custom Reports using the Report Wizard

There may be situations where the pre-defined reports do not provide the comparison information that you desire, or they may provide more information than is needed to answer the performance questions being asked. The Report Wizard allows you to create custom reports which can be tailored to your specific needs. It allows you to focus on response times for specific activities or events, and also to group and sort the data in the report so the appropriate comparisons can be made.

The Report Wizard assists you in the process of selecting results to include in the custom report, choosing a report presentation format, and saving the custom report for later re-use. Optionally, you may choose to include a sub-title and/or list the selected test identifiers, test names, and selected values.

Start the Report Wizard by selecting the Report Type **Wizard**, the Report Name **<New>**, and press the report action **Customize** button.

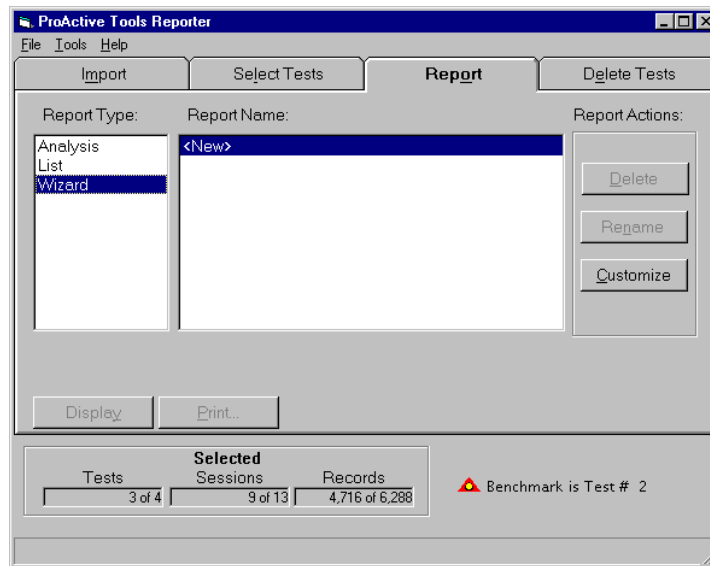


Figure 6-14: ProActive Reporter Report Wizard

Selecting the values to include in a Custom Report

A selected group of Simulator tests may contain results data for multiple activities, as well as for different clients, servers and Run IDs. The Report Wizard allows you to list the results from the selected tests and choose specific values to include in your customized report.

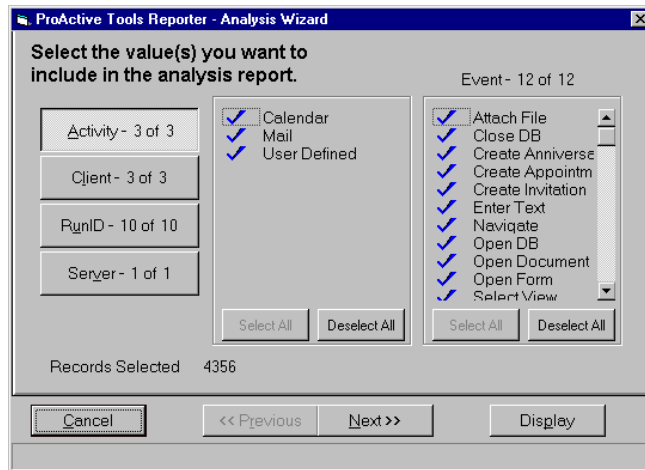


Figure 6-15: ProActive Reporter Report Wizard Selection

For example, by pressing the **Activity** toggle button, a list of activities and events included in the selected tests are displayed with all values selected. Using the mouse you may choose to de-select one or more values. When a value is selected to be included in the report, a check mark will be displayed to the left of the value. Optionally, you may use the **TAB** key, arrow keys, and spacebar to navigate to each list to select and de-select values.

When you have completed your selection, the **Records Selected** indicator is updated to reflect the number of Simulator records included in your custom report.

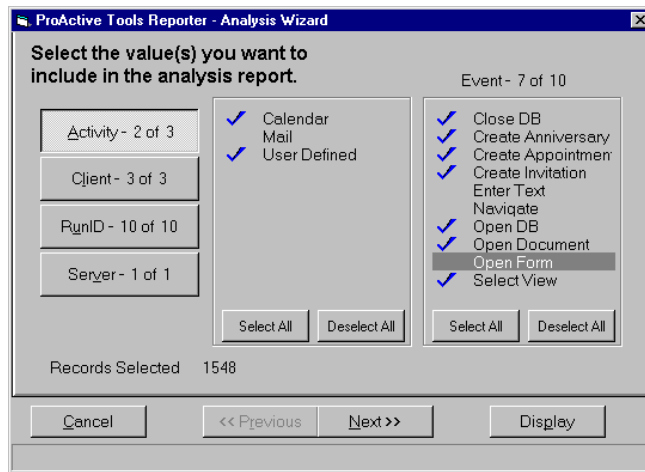


Figure 6-16: Reporter Report Wizard Activity/Event Selection

Press the **Next** button to proceed to format your custom report.

Selecting the report format for the Custom Report

The custom report may be formatted to display activities or events. You may also specify the criteria used for comparison in the report. For example if you want to compare response time from different servers, click servers, and the % Difference will be calculated based on the response times from different servers. You may use **Tests**, **Servers**, **RunIds**, and **Clients** for comparison criteria. After you have selected the criteria for comparison, you may also ask for additional report detail to be included, showing specific response time statistics for criteria not used in the comparison.

Use the mouse pointer to choose the **Format**, **Compare**, and **Detail** specifications for your customized report. A thumbnail sample of the report presentation is displayed based upon your choices.

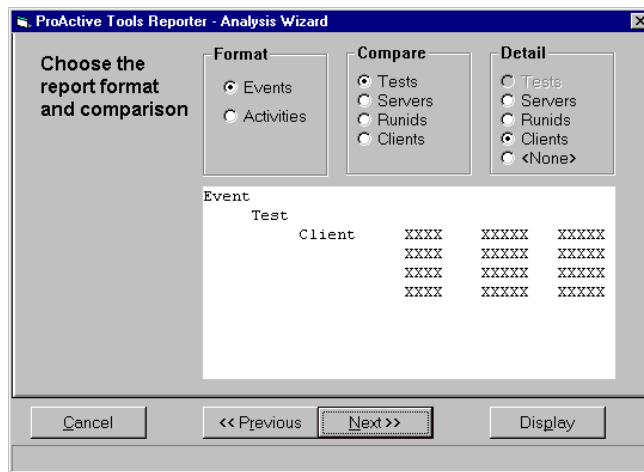


Figure 6-17: Reporter Report Wizard Report Format

Press the **Next** button to proceed with saving the custom report.

Saving the Custom Report

A completed custom report may be saved for later re-use or as a template for future custom reports. Check the **Save Report** checkbox to specify a name for the custom report.

Your free-form report comments are included as a sub-title for this custom report. Optionally, the saved custom report may include a list of the selected tests and the selected values included in the report.

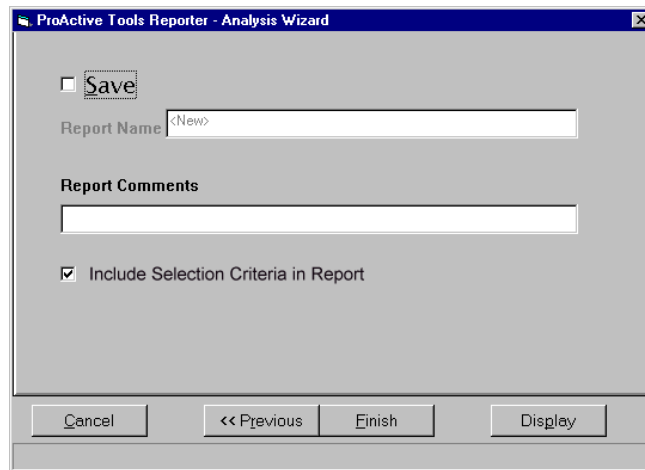
The image shows a Windows-style dialog box titled "ProActive Tools Reporter - Analysis Wizard". It has a standard title bar with a close button. The main area contains a checkbox labeled "Save" which is currently unchecked. Below this is a text field labeled "Report Name" containing the text "<New>". Further down is a section labeled "Report Comments" with a large empty text area. Below that is a checkbox labeled "Include Selection Criteria in Report" which is checked. At the bottom of the dialog are four buttons: "Cancel", "<< Previous", "Finish", and "Display".

Figure 6-18: Saving Custom Report

Press the **Finish** button to save the report and return to the ProActive Reporter.

Executing a Custom Report from the Reporter

Custom reports may be displayed and printed just as the predefined Analysis and List reports. Select the Wizard **Report Type** and choose the custom report to display or print. Then press the **Display** button to present the report on the screen or the **Print** button to print the report directly to the printer.

Delete a custom report by pressing the **Delete** button.

Rename a custom report by pressing the **Rename** button.

Select a custom report as starting point for a new custom report by choosing a current custom report and pressing the **Customize** button.

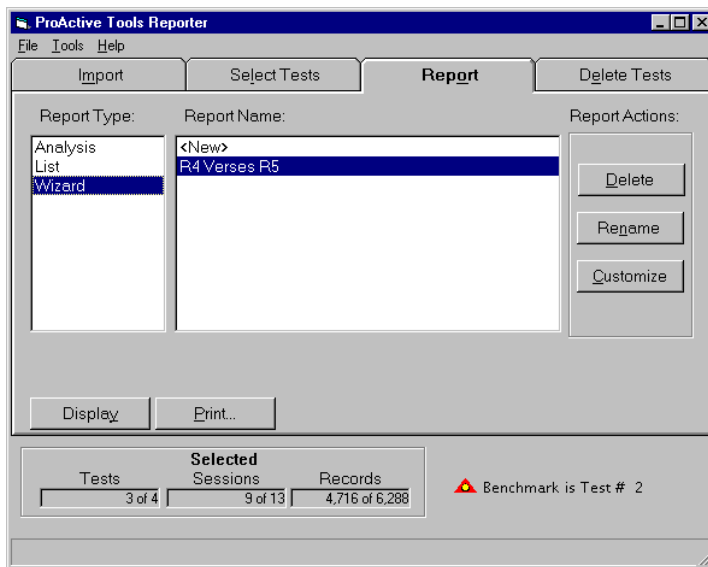


Figure 6-19: Executing a Custom Report

Deleting Test Files from Reporter Database

The Delete Tests folder allows you to choose tests for deletion from the Reporter database. The folder displays a list of all the tests in the database, the optional test name, the time frame that the test executed (From, To) and the optional description.

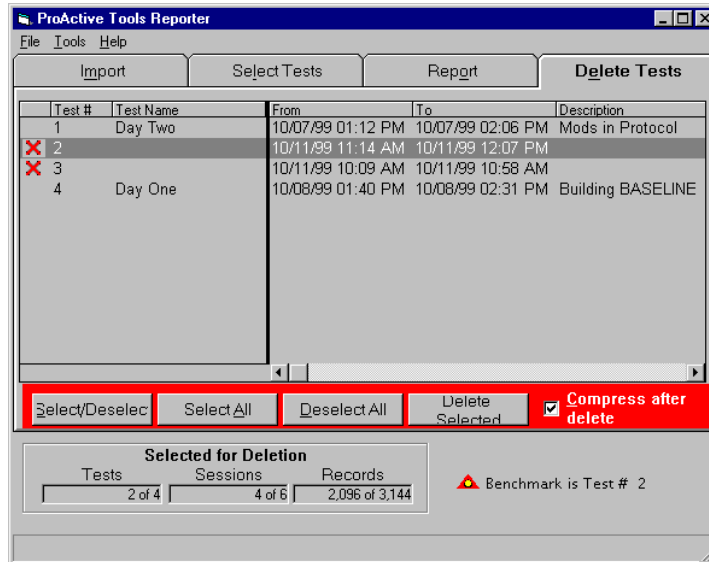


Figure 6-20: ProActive Reporter Delete Tests

Selecting Individual Tests - Select/deselect tests by clicking on the test with the left mouse button or highlight the test and click the Select/Deselect button (or pressing ALT-S). When a test is selected, an 'X' appears to the left of the test number.

Selecting/Deselecting All Tests - Select or deselect all tests for deletion by clicking on the Select All (or pressing ALT-A) or Deselect All (or pressing ALT-D) buttons.

Current Selections and Database Counts - The number of tests, sessions and records selected for deletion are displayed in the lower left corner of the folder.

Deleting Selected Tests - After selecting the tests to be deleted, click the Delete Selected button (or press **ALT-L**) to remove selected tests from the Reporter database. Optionally, you may compress the database file when the procedure is complete.

Prior to initiating this procedure, you will be prompted to be sure you want to delete database records. During the deletion process, a status message and status indicator will appear on the bottom of the screen.

Compressing the Database - When new records are added, the Reporter database size increases. As records are deleted, storage requirements for database records decrease. In order to recover this “empty” space, compress the database. Database compression is also available on the Tools menu.

Chapter 7 – ProActive Scenario Wizards

The Scenario Wizards were developed as a learning tool to provide new users with a structured method for developing scenario files. The Wizards, one each for ProActive Assistant and ProActive Load, are designed to allow you to step through the development of a scenario file and then export it as a text file to be used by Assistant or Load.

This chapter consists of the following sections:

- Scenario Wizards Overview.
- Using the Scenario Wizards to develop Scenario files.
- Using the Scenario Wizards to build Activity Routines.
- Exporting the scenario file.

ProActive Scenario Wizards Overview

The ProActive Scenario Wizards are Lotus Notes applications that create the scenarios run by ProActive Load and ProActive Assistant. For more information on the details of the structure of the Scenario file, see Chapter 4.

The ProActive Scenario Wizards allow you to easily create scenario files. Activity, Scenario and Thread Activity Section documents, the three major components of the application, are composed and stored as Lotus Notes documents in the Scenario Wizard databases. The Wizards' scenario document contains the General Section, and points to the Activity documents and Activity Set documents included in the scenario. Sample documents are included in the databases.

To run a completed scenario, the script is "exported" to the PROACTIV.INI file by clicking the Export to PROACTIV.INI button on the action bar of the scenario document.

Using the Scenario Wizards to build a Scenario file

There are four main sections the Scenario Wizards use to create a scenario; the Header Section, the General Section, the Activities Section and the Scenario Section. To create a scenario file, complete the first three sections of the scenario document and then push the "Generate Scenario" button. The Wizard then looks up, and pulls in, referenced activity detail and thread activity information (for ProActive Load scenarios), adds required formatting and syntax, and creates a Scenario File ready for export and use by ProActive Load or ProActive Assistant.

Header Section

The Header Section contains the scenario name, descriptions and comments. The scenario name is a required field and should be a unique name for identification purposes.

General Section

The General Section provides global information that will be used as the default for much of the scenario. Complete the General Sections as required. Parameters in this section serve as the default value for any activity. For ProActive Load scenario files, the Thread parameters are also assigned in the General Section. To create a Thread Activity Section document, click the Assign Thread button in the General Section. Repeat for multiple entries.

Activities Sets Section

ProActive Load scenario files may have multiple Activity Sets, which define the Activities Routines to be run and their order of execution. Thread specific Activity Sections are setup in the General Section where threads, or ranges of threads, can be assigned to the named thread activity section. Use the "Assign Threads" button and follow the prompts. Thread Activity Sections are created in separate Notes documents and referenced in the Scenario document. Threads not assigned to specific Thread Activity Sections use the section on the Scenario document named "Activities." If desired, an existing Activity Set can be designated as the Default." All Activity Sets are created by selecting from a list of existing Activity Sets documents. Since Activities Routines are stored as

documents, they can be reused. If you need an Activity Routine that does not exist, create a new Activity document, and save it. Once an Activity exists, you can use the Insert/Append Parameter button to select and add it to the scenario. The Scenario Wizard will add Activities that you check from the Activity list in the order that they are checked. Add Do loops to repeat these Activities for a certain amount of time or a number of iterations.

Note: To review or edit the parameters for an activity while in a scenario document, go to the Activity Window, place the cursor anywhere on the line containing Call= and the desired activity, and click the "Open Activity" button located on your scenario document's action bar. This opens the referenced activity document.

Scenario Section

The Scenario Section compiles the PROACTIV.INI file. After the General and Activity Sections of the Scenario document have been completed, click the Create Scenario button to generate the information and the format required.

The scenario is regenerated every time the button is clicked. If you change parameters in the General or Activity Sections, click the Generate Scenario button to pull in the new information. If you change parameters in an Activity document referenced in the scenario, clicking the Generate Scenario button will pull in the new information from the Activity document.

The **Generate Scenario** button is located near the bottom of the Scenario document in the Scenario Section.

The **Clear Scenario** button will delete the scenario text only. The rest of the data entered on the Scenario form will not be changed, and of course, the referenced documents will not be changed.

Using the Scenario Wizards to build an Activity Routine

To build an Activity, open the Scenario Wizard database and create an Activity document. This can be done by clicking the Activities View button on the opening navigator and then clicking Create Activity from the view action bar. You can also create activity documents from within a scenario document by clicking the Create Activity button on the form action bar. This will ask you for the type of activity that you wish to create and then prompt you through the Activity form.

The Activity forms have typical Notes keyword fields and simple text fields requiring data entry. Fields left blank and referenced as "From General" will use values in the General Section of the Scenario document. This feature allows repetitive data to be kept in the Scenario document, thereby minimizing data entry.

When you finish with the Activity document you will be prompted to save it. Use a descriptive name as this will remain in your library of Activity documents for future use.

<p>Note: To create an Activity similar to an existing one, copy an Activity document at the Notes view level and paste the copy back into the view. Edit the copy as desired.</p>
--

Exporting Scenario Files

The final product of a Scenario document is the ProActive Load parameter file (PROACTIV.INI). Here are the steps required to generate the PROACTIV.INI file.

Create a scenario with General Sections and Activity Sections complete.

Push the **Generate Scenario** button to compile and display the syntax in the Scenario Window.

Click the **Export to PROACTIV.INI** button.

Assign any filename for the PROACTIV.INI file. Any filename valid in your Operating System will work.

Select the Location for your PROACTIV.INI file. Include a valid path to the file without repeating the filename. If you have entered location values in the field located at the top of the Scenario document, these values will be listed, or choose "Other" and enter the location.

You have an option to save the exported file in the scenario document by choosing "Yes" as shown below. The scenario script is then saved in a text field in the document.

If you have an existing PROACTIV.INI file at the specified location with the same filename, you will be warned that the new file will overwrite the existing file. Click "Replace" to put the new file in place.

Set Word Wrap to a value of 200, to eliminate wrapping in the PROACTIV.INI file.

Click **"OK"** and the scenario will be exported and can be run.

Appendix A – Scenario File Parameters

This Appendix provides a detailed listing and description of all the parameters and their related values that are used in ProActive Tools scenario files. This section is a reference guide for those who are developing test scenarios with ProActive Tools.

This Appendix is formatted with a section for each of the predefined Activity Types, with a first section dedicated to the General Section parameters. Following the General Section parameters, the remaining predefined Activity Types are listed in alphabetical order.

Each section describes the events that are executed, when that Activity Type is used in a scenario file, a table of valid parameters, and an example of the Activity Type used in a scenario file.

The last section of this Appendix describes how Text parameters are used in both ProActive Assistant and ProActive Load scenarios.

General Section Parameters

This section documents all of the parameters that are supported in the General Section of the scenario file. For a complete description of the General Section see Chapter 4.

General Section Parameters:

Keyword	Value	Default	Used in Assist?	Used in Load?
Client	Name of the workstation	n/a	Yes	Yes
CloseNotesWhen Done	Yes Closes Notes after Assistant session No Leaves Notes up after the session	No	Yes	No
DBName ¹	Default database name	n/a	Yes	Yes
DefaultActivity	Assigns the name of the Default Activity Set that is to be used for threads that have not been assigned to a specific Activity Set. See Thread[n] Default is Activities		No	Yes
DeleteResultFile	Yes Delete Result file after successfully mailing it (Only if MailResults=Yes) No Don't delete Result file	No	Yes	No
ErrorFileName	Specifically names the file to be used to log errors. Default is c:\temp\G2paerr.lst		No	Yes
FirstThread	Number assigned to the first thread of Load test. Subsequent threads are incremented by 1.	1	No	Yes
IDFile	Default Notes ID file to switch to before the activity	n/a	Yes	No
ListNoteID	Yes Includes the Notes Document ID number of the document opened in the comments section of the CSV file. No Does not include Notes document ID	No	No	Yes
Location	Default Name of location to switch to before the activity	n/a	Yes	No

General Section Parameters (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Logoff	Yes Log the user off after each activity No Don't Log the user off	No	Yes	No
MachineInfo	Additional information about the workstation to be written to the session record in the Result file. (For example, machine's clock speed)	n/a	Yes	No
MailBoxes	Enables multiple mail.box files on R5 Server (only) MailBoxes= [specifies number of mail.box files specified in Server Configuration document] Example: MailBoxes = 3	1	No	Yes
MailDBName	Name of database from which to mail the Results file (Only if MailResults=Yes)	n/a	Yes	No
MailServer	Name of server from which to mail the Results file (Only if MailResults=Yes)	n/a	Yes	No
MailResults	Yes Mail the Results file after the Assistant test session ends No Don't mail the Results file	No	Yes	No
MailResultsTo	Recipient of Results file (Only if MailResults=Yes)	n/a	Yes	No
MaxStartWait ²	Grace period for StartTime, expressed <hour>:<minutes> or <hour> (Only if StartTime is not blank)	n/a	Yes	Yes
MaxWait	Number of milliseconds to wait for Windows to go idle	500	Yes	No
NotesIniPath	The path used by the scenario to locate the notes.ini file	n/a	No	Yes
NTServerName ³	The NT name of the server to be monitored for remote counters (Used with NT servers only)	n/a	No	Yes
NumSets	Default number of times to run an Activity (Only if RunType=Count)	5	Yes	Yes

General Section Parameters (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
NotesExeName	Name of Notes executable file	Notes.Exe	Yes	No
OnError	Ask Prompts to continue or quit when an error is encountered Continue Continues if an error is encountered Quit Quits if an error is encountered	Ask	Yes	No
Password ⁴	Text of password	n/a	Yes	No
Path ⁵	Set of directories (separated by semicolons) to search for files referenced in the scenario file.	n/a	Yes	Yes
Random	Yes Sets a random seed using the last four digits of the system timer. No Random number not seeded.	No	Yes	Yes
ResultFile	Path and Name of the Result file Default is c:\work\proactiv.csv		Yes	Yes
ResultOutput	Type of records to be included in the Result File. Default is all Load record types: Session, Sleep, Activity, and Event		No	Yes
RunID	Value to be placed in the RunID column of the Result file for this activity	n/a	Yes	Yes
RunType	Count Runs an activity for NumSets iterations Timed Runs an activity for TestDuration Minutes	Timed	Yes	Yes
Server	Lotus Domino Server Name	n/a	Yes	Yes
SleepTime	Default time (in seconds) to sleep between iterations	n/a	Yes	Yes
StartTime ³	Time to start the test session (in 24 hour format)	n/a	Yes	Yes
TestDuration	Default time (in minutes) to run an activity (Only if RunType=Timed)	5	Yes	Yes

General Section Parameters (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
TestMode	Yes Runs the scenario in Test Mode No Runs the scenario in Normal run Mode	No	Yes	Yes
Thread[n]	Assignment of a thread or range of threads to a specific Activity Set	n/a	No	Yes
Threads	Number of threads to be run during the test	n/a	No	Yes
ThreadDelay	Time (in seconds) to delay between starting each thread	n/a	No	Yes
TimeLineFile	The fully qualified path and file name for the TimeLine output file	n/a	No	Yes

¹ In ProActive Load, there are three ways to use this parameter. If just one database name is listed, as it would be in ProActive Assistant, it will be the default for all activities that do not explicitly list a database name in the activity. Multiple databases can be listed and associated with threads. Thread substitution can be used to associate a unique database with each thread while using only one line in the scenario. For additional information, see the Thread explanation in the product specific parameters in Chapter 4.

² For a complete description of the use of these parameters, see the “Delayed Start” explanation in the General Section of Chapter 4.

³ When this parameter is included in the General Section of a ProActive Load scenario, the NT server statistics will be displayed on the Main screen of the Load frontend along with the name of the server. These values are also recorded in the TimeLine file.

⁴ If password is supplied, enter a string when the Enter Password box is presented. Please note that passwords in Lotus Notes are case sensitive. The password string must be entered in the correct case. This parameter is intended to be used for test user Ids, please check your organization’s security policies before using this parameter.

⁵ If multiple directories are listed, the entire text string must be enclosed in quotes. For a single directory, no quotes are required. For ProActive Load to execute properly, the Notes executables must be in the system path. The path statement included in the Load scenario cannot be used to locate executables.

General Section Scenario Examples:

ProActive Assistant	ProActive Load
[General]	[General]
RunId = PAASample	RunID = PALSample
Path = c:\progra~1\patools	Path = c:\progra~1\patools
	NotesIniPath = c:\winnt
Client = @notes.ini.keyfilename	Client = @notes.ini.keyfilename
Server = @Notes.ini.mailserver	Server = @Notes.ini.mailserver
NotesEXEName = Notes.exe	NTServerName = G2Server01
CloseNotesWhenDone = No	
MaxWait = 500	
RunType = Count	RunType = Count
NumSets = 1	NumSets = 1
SleepTime = 10	SleepTime = 10
ResultFile= c:\Work\PAACompose.CSV	ResultFile = c:\Work\PALCompose.CSV
MailResults = no	ResultOutput = Session, Sleep, Activity, Event
MailResultsTo = Test UserA01	TimeLineFile = c:\Work\PALComp1tl.CSV
MailDBName = @notes.ini.mailfile	ErrorFileName = c:\Work\ErrorLog.lst
MailServer= @notes.ini.mailserver	
DeleteResultFile = no	
StartTime = 14:30	StartTime = 14:30
MaxStartWait = 2	MaxStartWait = 2
TestMode = No	TestMode
Random = Yes	Random = Yes
	ListNoteID = Yes
	Threads = 5
	ThreadDelay = 1
	FirstThread = 1
DbName = @notes.ini.mailfile	Dbname = mail\tusera<thread>.nsf
	Thread1 = Activity1
	Thread2-3 = MailCal

Monitor Section

The optional Monitor Section specifies parameters that set up and enable the monitoring function. Alerts are sent to the person(s) named in the AlertWho parameter when thresholds for an event set with the Threshold parameter are exceeded. The Event, set with the Event parameter, can be any valid Notes event measured by ProActive Assistant. When the scenario runs, if the event specified in the Monitor section occurs in any of the activities in the scenario, the event is monitored by ProActive Assistant, and alerts are sent when the threshold is exceeded. To use the monitor feature, first set up the Monitor parameters as desired and then add one or more activities to the scenario that trigger the event that you want to monitor.

Alerts can be sent via a Notes message, SNMP message, or Local message on the workstation status bar. The Notes alert messages contains server name, database name, event name and the threshold exceeded, the number of occurrences, the time of the first occurrence, and how many times the alert has been generated.

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
AlertBell	File name of .WAV file to be played when alert is sent (Only if AlertType contains Local)	n/a	Yes	No
Alertcc ¹	Copy to recipient(s) for Notes alert messages (Only if AlertType contains Notes)	n/a	Yes	No
AlertLog	Yes Writes each alert to Alert.Log No Don't write Alert.Log	No	Yes	No
AlertMessage	Message to associate with the Lotus Message alert. Multiple message lines may be used for multi-line message.	n/a	Yes	No
AlertType	One or more of the following, separated by comma Local Display alert on status panel of simulator machine Notes Send a Lotus Notes message SNMP Send an SNMP trap	Local	Yes	No

Parameters used in test scenarios (continued):

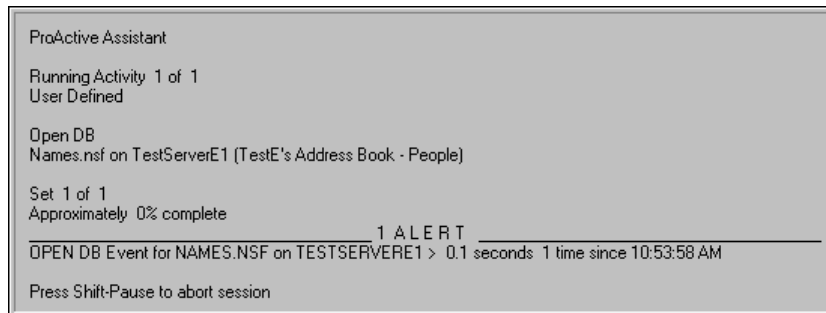
Keyword	Value	Default	Used in Assist?	Used in Load?
AlertWho ¹	Recipient(s) of Lotus Notes alert message. Separate multiple recipients with commas (Only if AlertType contains Notes)	n/a	Yes	No
DBName	Name of database to be monitored	n/a	Yes	No
Event	Event to be monitored	n/a	Yes	No
FirstAlert	Number of times in a row the threshold is exceeded before alerting.	n/a	Yes	No
NextAlert	Number of times in a row the threshold is exceeded once the first alert has been sent before sending another alert. (Zero indicates alert only once)	n/a	Yes	No
Server	Server to be monitored	n/a	Yes	No
SNMPSendTo	IP address(es) or Hostname(s) to send SNMP trap (Only if AlertType contains SNMP)	n/a	Yes	No
Threshold	Alert threshold (in <seconds>.<tenths>)	n/a	Yes	No

¹ Since Notes alerts are sent via the Notes API there is no validation of these addresses.

Notes: Server, DBName, Event, Threshold, FirstAlert, and NextAlert are treated as a set. Any number of sets can be entered, but a server must be the first parameter for each set. AlertType, AlertWho, AlertMessage, AlertLog, and AlertBell may only be entered once.

Before the Monitor function can be used, the Notes executables must be in the system path of the workstation running ProActive Assistant. Installing Notes does not put the executables in the system path.

When this example Monitor scenario runs, the alert is displayed on workstation screen in the lower right each time the threshold is exceeded, as shown below.



Example:

```
[Monitor]
'Password      = Password
Server         = TestServerE1
DBName         = Names.nsf
Event          = Open DB
Threshold      = .1           ` One tenth of a second
FirstAlert     = 1
NextAlert      = 1
'AlertBell     = d:\Winnt\Media\msremind.wav
AlertBell      = d:\Winnt\Media\tada.wav
AlertType      = Notes,Local
AlertWho       = Test UserA01
Alertcc        = Test UserA2
AlertMessage   = This is an Monitor1 test
AlertLog       = Yes
```

' Activity Sets Section

[Activities]

Call = TestMonitor

' Activity Routines Section

[TestMonitor]

```
Activity      = User Defined
Server        = TestServerE1
DBName        = Names.nsf
RunType       = Count
NumSets       = 1
Command       = Open Database
View          = People
Command       = Close Database
```

Activity Sets Section

The Activity Sets Section represents the order and timing of the Activity Routines that are to be executed during the running of the test scenario. The Activity Sets Section has the same structure for both Assistant and Load, except that ProActive Load can have multiple Activity Sets that can be assigned to specific threads in the General Section. Since a workstation running ProActive Assistant represents a single Notes session (equivalent to a single thread in ProActive Load), it can have only one Activity Set per scenario.

In a ProActive Assistant scenario, the single Activity Set must be named **[Activities]**. In a ProActive Load scenario there are no required names for the Activity Sets, except for the default Activity Set which is normally named **[Activities]**. By default, all unassigned threads use the Activity Set named **[Activities]** or the Set defined by the Default Activity Parameter. This default can be changed by using the **DefaultActivity=** parameter.

This section documents all of the parameters that are supported in the Activity Sets Section of the scenario file. For a complete description of the Activity Sets Section see Chapter 4.

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Call	Indicates the name of the Activity Routine to be executed. This Activity Routine MUST be in the Activity Routines Section of the scenario file.	n/a	Yes	Yes
Do	Scenario will repeat all instructions between the Do statement and the next End statement for the specified number of times. e.g., Do 5 times	n/a	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Do for	Scenario will repeat all instructions between the Do statement and the next End statement for the specified number of minutes. e.g., Do for 10 minutes	n/a	Yes	Yes
Do Forever	Scenario will repeat all instructions between the Do statement and the next End statement until manually stopped.	n/a	Yes	Yes
Do Until	Scenario will repeat all instructions between the Do statement and the next End statement until a specific time is reached. e.g., Do Until 5:30 PM	n/a	Yes	Yes
End	Indicates the End of a Do Loop	n/a	Yes	Yes
IDFile ¹	Instructs ProActive Assistant to find the Notes ID indicated and perform a Switch ID operation.	n/a	Yes	No
Location	Instructs ProActive Assistant to switch to the Location indicated	n/a	Yes	No
SleepTime	Specifies number of seconds for the scenario to delay execution.	n/a	Yes	Yes
SleepUntil ²	Specifies the exact time for the scenario to delay execution.	n/a	Yes	Yes

¹ ID file indicated must be located in the subdirectory with the ProActive Assistant executables.

² For detailed description of the options available for SleepUntil see Chapter 4. See SleepUntil in the Index for the specific page number.

Comparison Sample Activities Set Section for Load and Assistant

ProActive Assistant	ProActive Load
Activity Sets Section	Activity Sets Section
[Activities] Do 3 Times Call = ReadDocs SleepTime= 5 Call = ChangeViews SleepTime= 5 Call = TestMail End Call = TextIndex SleepTime= 5-10 Call = RebuildViews SleepTime= 5 Call = Find1 SleepTime= 5-10 Call = ComposeDocs SleepTime= 5 Call = ComposePhone Do for 5 minutes SleepTime= 5 Call = Calendar SleepTime= 10 Call = UserReply End	[Activity1] Do for 10 minutes Call = ComposePhone SleepTime= 1-10 Call = RBView SleepTime= 10-20 Call = EditDocs1 End [MailCal] Do 2 Times Call = TestMail SleepTime= 5 Call = Calendar1a Call = Calendar1b Call = Calendar1c SleepTime= 5 Call = Find1 End [Activities] Call = ReadDocs SleepTime= 5 Call = Find1 SleepTime= 5 Call = ComposeDocs SleepTime= 5

Calendar Activity

Calendar Activities open a Notes database, create calendar entries, and optionally send invitations. At your option, Calendar Activities look up free time for invitations. This activity can generate a heavy load on the server, especially if a free time search is applied against many users, or across multiple servers or Notes domains.

Events

Open DB - Opens the selected database

For each iteration:

Open Form - Opens Calendar Entry form (Assistant Only)

Create <Calendar Type> - Creates Calendar Entry of type Calendar Type

Enter Text - Types characters into the description field of the Calendar entry

Find Free Time - Invokes the Lotus Find Free time function for Invitations

Following the final iteration:

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Calendar			
CalendarType	Appointment Invitation Event Reminder Anniversary	n/a	Yes	Yes
CalendarLocation	Sets the Location field in the Calendar entry. Only available for R5 mailfiles	n/a	Yes	No
Category	Sets the Categories field in the Calendar Entry. Only available for R5 mailfiles	n/a	Yes	No

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Date	Date of Calendar Entry	n/a	Yes	Yes
DBName	Database name to use	From General	Yes	Yes
Description ¹	Short Description	n/a	Yes	Yes
Duration	Number of days of Event type Calendar Entry	n/a	Yes	Yes
FindFreeTime	Yes Invokes Lotus Notes FindFreeTime No Do not invoke FindFreeTime (only valid for Invitations)	No	Yes	Yes
KeepPrivate	Yes Make the Calendar Entry private No Do not make the Calendar Entry Private	No	Yes	No
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
OptionalAttendees	Name(s) of people to invite in the OptionalAttendees field (only valid for Invitations) Can contain up to 1,000 characters		Yes	Yes
PencilIn	Yes Pencil in the Calendar Entry No Do not pencil in the Calendar Entry	No	Yes	Yes
RequiredAttendees	Name(s) of people to invite in the RequiredAttendees field (only valid for Invitations)		Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
SendInvitation	Yes Send invitations No Do not send invitations (only valid for Invitations)	Yes	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Server	Lotus Domino server	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes
Text[n]	Detailed Description field of the Calendar Entry. Multiple Text parameters may be specified See Appendix B for more information	n/a	Yes	Yes
Time	Time or Time range of Calendar Entry	n/a	Yes	Yes

¹ In Notes R5, Description is now referred to as the Subject.

This example activity can be used in both Assistant and Load.

[Calendar]

Activity = Calendar
RunID = CalAnniv
RunType = Count
NumSets = 1
CalendarType= Anniversary
Description = Test Anniversary
Date = 8/15/00
Text1 = This is my anniversary

Compose Activity

The Compose Activity opens a Notes database, composes, and optionally sends documents. This activity can be used to compose documents using any form in any database. When using the memo form in a mail database, this is like the Mail activity, but allows customization to handle different forms. This activity can generate light or heavy loads, depending on the size of the addressee lists in the documents being mailed. With large addressee lists, it can generate significant loads on the router task in the server.

Events

Open DB - Opens the selected database

For each iteration:

Open Form - Opens <formname> (Assistant Only)

Enter Text - Types characters into the form <formname>

Attach File - Attaches a file (or files) to the form (if applicable)

Send - Sends the form (if Send is Yes)

Save - Saves the form (if Save is Yes)

Discard - Discards the form (doesn't save or send it, if Send is No & Save is No)

Following the final iteration:

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Compose		Yes	Yes
Attach	Name of file(s) to be attached to the document. Multiple files can be separated by commas. Wild cards are not permitted. Attachments only occur after last line of text.	n/a	Yes	Yes
Compress	Yes compresses the attachment No does not compress the attachment	Yes	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
DBName	Database name to use In PA Load, DBName can be thread specific.	From General	Yes	Yes
Encrypt	Yes encrypts the document No does not encrypt the document	No	Yes	Yes
Field ¹	Parameter format is fieldname=value, e.g. Field=Subject=Document Subject	n/a	Yes	Yes
FieldN	See User Defined – Field Parameters	n/a	No	Yes
FieldTD	See User Defined – Field Parameters	n/a	No	Yes
Form	Form to compose. If the form is in a secondary menu, use the primary menu name followed by a backslash (\), then the secondary menu name. For example (Special\Phone Message).	n/a	Yes	Yes
GoToField	Field name to move cursor to	n/a	Yes	No
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RunID	Value placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
Save	Yes saves the document No does not save the document	No	Yes	Yes
Send	Yes sends the document No does not send the document	Yes	Yes	Yes
SendTo	Recipients(s) of Sent Note		No	Yes
Server	Lotus Domino server	From General	Yes	Yes
Sign	Yes signs the document No does not sign the document	No	Yes	Yes
SleepTime	Number or range of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number or range of minutes to run a timed activity	From General	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Text ¹	Parameter format is fieldname=value, e.g. Text=Subject=Document Subject	n/a	Yes	No
Text[n]	Text of the memo Multiple Text parameters may be specified See Appendix B for more information		Yes	Yes

¹ ProActive Assistant will only allow the use of the **Field=[fieldname]=value** and **Text=[fieldname]=value** for fields that appear in the User Interface. This is not supported for hidden fields. With Notes R5, it is recommended to include the statement **Field=Repeats=0** with Anniversary activities. R5 automatically sets the Repeats field to Yes, this results in the creation of an extra unusable document. For a more in-depth explanation of the use these parameters see description of text parameters at the end of Appendix B.

The example activity below can be used in both Assistant and Load.

[ComposeDocs]

Activity = Compose
RunID = ComposeDocs
RunType = Count
NumSets = 1
DBName = mail\tusera1.nsf
Form = Memo
Save = Yes
Send = Yes
Field = Sendto = Test UserA10
Field = Subject = Test ProActive Compose Activity
Text1 = ComposeDocs - This memo was composed by the
Text2 = <enter> ProActive Assistant Compose Activity

Copy Database Activity

The Copy Database Activity makes a Notes Database Copy (not a replica copy) on the server specified in the Activity.

Events

Open DB - Opens database to be copied

Copy DB - Creates a new database from the copy

Close DB - Closes database that was copied

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Copy Database		Yes	Yes
CopyAccessList	Yes copies Access Control List from original database to the new database No does not copy Access Control List	Yes	Yes	Yes
CopyDesignOnly	Yes copies only the design of the database and none of the documents in the database No copies the design and all of the documents	No	Yes	Yes
CopyTitle	The database Title to be used for the new database		Yes	Yes
CopyToFile	The file name of the new database		Yes	Yes
CopyToServer	The server the new database is to be copied to		Yes	Yes
DBName	Database to be copied DBName can be thread specific.	From General	Yes	Yes
DeleteCopy	Yes Deletes the copy after completing the activity No keeps the copy after completing the activity	Yes	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
ReplaceCopy	Yes replaces an existing copy of the new database No will not replace an existing copy of the database, therefore, no new copy of the database will be made.	No	Yes	Yes
Server	Lotus Domino server	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes

The example activity below can be used in both Assistant and Load.

[CopyDB]

Activity = Copy Database
RunID = CopyDB
RunType = Count
NumSets = 1
DBName = discuss.nsf
CopyAccessList = Yes
CopyDesignOnly = No
DeleteCopy = No
CopyTitle = Discussion Database
CopyToServer = TestServerB1/G2Test
CopyToFile = discuss2.nsf

Edit Activity

The Edit Activity opens a Notes database, opens and edits a document, then saves it. Before the document is edited, a copy can also be made. Like the Read Documents and Mail activities, this is a fairly normal user activity with a light load on the server.

While the end result is similar with both ProActive Assistant and ProActive Load, they each execute this activity in a slightly different manner. ProActive Assistant Edit looks for a specific document to edit in the specified view. ProActive Load does not select specific documents for edit, but starts with the first document in the view and then edits the number of documents indicated in the scenario. ProActive Assistant always edits the original document. ProActive Load will also edit the original document unless a copy of the original document was made. In this case, ProActive Load will edit the copied document.

Events

Open DB - Opens the selected database

Find Document – Selects document to be edited. Load selects the first document in the view, while Assistant uses the FindSubject parameter to select a specific document.

Copy Document - Copies the document if CopyDocument is Yes

Prepare Document - Inserts a unique string into the document subject and saves the document (Assistant Only)

For each iteration:

Edit Document(s) – Edits the document and saves it if Save=Yes.

Following the final iteration:

Delete Document - deletes the edited document if DeleteDocument is Yes

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Edit		Yes	Yes
Attach	Name of file(s) to be attached to the document. Multiple files can be separated by commas. Wild cards are not permitted.	n/a	Yes	Yes ¹
Compress	Yes compresses the attachment No does not compress the attachment	Yes	Yes	Yes
CopyDocument	Yes makes a copy of the document before editing No edits the document without making a copy	Yes	Yes	Yes
DBName	Database name to use In PA Load, DBName can be thread specific.	From General	Yes	Yes
DeleteDocument	Yes deletes the document after completing the activity No preserves the document after completing the activity	Yes	Yes	Yes
Encrypt	Yes encrypts the document No does not encrypt the document	No	Yes	Yes
Field	Parameter format is fieldname=value, e.g. Field=Subject=Document Subject		Yes	Yes
FieldN	See User Defined – Field Parameters	n/a	No	Yes
FieldTD	See User Defined – Field Parameters	n/a	No	Yes
FindSubject	String used to locate document to edit This string must (at least partially) match a field that appears in the view (typically the Subject field).		Yes	No
NumberOf Documents	Number (or numeric range) of documents to edit in one iteration	10	No	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
Save ²	Yes saves the document No does not save the document	Yes	Yes	Yes
Send ³	Yes sends the document No does not send the document	No	Yes	Yes ⁴
SendTo	Recipient(s) of the memo Multiple recipients may be specified (separated by commas) Multiple SendTo parameters may be specified. The activity will cycle through each SendTo parameter, sending a message to each set of SendTos	n/a	No ²	Yes
Server	Lotus Domino server	From General	Yes	Yes
Sign	Yes signs the document No does not sign the document	No	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes
Text[n]	Text of the document Multiple text parameters may be specified See Appendix B for more information		Yes	Yes
View	View to open. If the view is in a secondary menu, use the primary menu name followed by a backslash (\) for example, Main view\details	Existing Notes view	Yes	Yes

¹ In Load, all attachments are shown at the end of the document and not in a specific field.

² Load saves the doc if Save=Yes, but the edited document is deleted when the activity completes if DeleteDocument = Yes. If DeleteDocument=No, then the documents are saved in the database.

³ In ProActive Assistant, the SendTo field must already exist in the document for it to be Sent.

⁴ In ProActive Load, if the SendTo field does not already exist, it can be created with the “SendTo=” keyword.

ProActive Assistant Example:

[EditDocs1]

Activity	= Edit
RunID	= EditDocs
DBName	= Lotus\Discuss.nsf
View	= (\$All)
FindSubject	= PAA Edit
Deletedocument	= Yes
CopyDocument	= Yes
Save	= Yes
Text	= This document has been Edited
Attach	= File.txt

ProActive Load Example:

[EditDocs1]

Activity	= Edit
RunID	= EditDocs
DBName	= Lotus\Discuss.nsf
View	= \$All
NumberOfDocuments	= 3
Deletedocument	= Yes
CopyDocument	= Yes
Save	= Yes
Field	= Subject = Test of Edit Document
Field	= Field1 = Just Testing.
Field	= Field2 = Testing Again.
Attach	= File.txt

Find Activity

The Find Activity opens a Notes database and searches for specified strings. Databases must be full-text indexed or error messages will result.

Events

Open DB - Opens the selected database

For each iteration:

Find String - Finds a string within the database

Following the final iteration:

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used In Assist?	Used in Load?
Activity	Find		Yes	Yes.
DBName	Database name to use In PA Load, DBName can be thread specific.	From General	Yes	Yes
FindString	String to find. Multiple strings may be specified	n/a	Yes	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
Server	Lotus Domino server	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used In Assist?	Used in Load?
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes
View	View to open If the view is in a secondary menu, use the primary menu name followed by a backslash (\) then the secondary menu name for example, Main view\details	Existing Notes view	Yes	Yes

The example activity below can be used in both Assistant and Load.

[Find]

Activity = Find
RunID = FindHelp
RunType = Count
NumSets = 3
DBName = Sample.nsf
FindString = NetBios
FindString = TCP/IP
FindString = Help

Index Activity

This activity opens a Notes database and creates a full-text index. Very few users would actually perform this activity. It generates significant CPU and I/O loads on the server, and might be executed by one workstation in a scenario. Full-text indexing a large database can take several minutes. **This Activity is only available for ProActive Assistant.**

Events

For each iteration:

Open DB - Opens the selected database

Delete Index - If the database is already indexed, deletes the index

Create Index - Creates a full-text index for the database

Delete Index - If DeleteIndex is Yes, deletes the full-text index

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Index		Yes	No
CaseSensitive	Yes creates a case-sensitive index No creates a case-insensitive index	No	Yes	No
DBName	Database name to use	from General	Yes	No
DeleteIndex	Yes deletes the index after indexing No retains the index	No	Yes	No
IndexAttachments	Yes indexes attachments No does not index attachments	No	Yes	No
IndexBreaks	Sentence indexes on word, sentence, paragraph Word indexes on word only	Word	Yes	No

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
IndexEncrypted	Yes indexes encrypted fields No does not index encrypted fields	No	Yes	No
NumSets	Number (or numeric range) of times to run a count activity	from General	Yes	No
RunID	Value to be placed in the RunID column of the Results file for this activity	from General	Yes	No
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	from General	Yes	No
Server	Lotus Domino server	from General	Yes	No
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	from General	Yes	No
StopWords	Yes uses a stopword file No does not use a stopword file	No	Yes	No
StopWordFile	Name of stopword file. (Only used if StopWords= Yes)		Yes	No
TestDuration	Number (or numeric range) of minutes to run a timed activity	from General	Yes	No

ProActive Assistant Scenario Example:**[TestIndex]**

Activity = Index
RunID = TestIndex
DBName = Sample.nsf

Mail Activity

The Mail Activity opens a Notes database, composes and sends memos. This activity can generate light or heavy loads, depending on the size of the addressee lists in the documents being mailed. With large addressee lists, it can generate significant loads on the router task in the server. Large contents or large attached files can be used to generate network loads.

Events

Open DB - Opens the selected database

For each iteration:

Open Form - Opens the Memo form (Assistant Only)

Enter Text - Types characters into the message document

Attach File - Attaches a file (or files) to the form (if applicable)

Send - Sends the document to the recipient in the SendTo parameter (if Send is Yes)

Save - Saves the document (if Save is Yes)

Discard - Discards the document (doesn't save or send it, if Send is No and Save is No)

Following the final iteration:

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Mail		Yes	Yes
Attach ¹	Name of file(s) to be attached to the document. Multiple files can be separated by commas. Wild cards are not permitted.		Yes	Yes ²
Compress ¹	Yes compresses the attachment No does not compress the attachment	Yes	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
DBName	Database name to use In PA Load DBName can be thread specific	From General	Yes	Yes
Encrypt ¹	Yes encrypts the document No does not encrypt the document	No	Yes	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
Save ¹	Yes saves the document No does not save the document	No	Yes	Yes
Send ¹	Yes sends the document No does not send the document	Yes	Yes	Yes
SendSubject	Subject of the memo		Yes	Yes
SendTo ³	Recipient(s) of the memo Multiple recipients may be specified (separated by commas) Multiple SendTo parameters may be specified. The activity will cycle through each SendTo parameter, sending to each		Yes	Yes
Server	Lotus Domino server	From General	Yes	Yes
Sign ¹	Yes signs the document No does not sign the document	No	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes
Text[n]	Text of the memo Multiple Text parameters may be specified See Appendix B for more information		Yes	Yes

¹ When this parameter is on the same line as a SendTo parameter (i.e., separated by semicolon), the parameter applies only to that recipient(s). If it is on a line separate from a SendTo parameter, it applies to all recipients in that Activity.

² Only after the end of the entered text.

³ See section on SendTo Options for the various methods that can be used with ProActive Load for developing addressee lists.

ProActive Load Scenario Example:

[TestMail]

Activity = Mail
DbName = Mail\tusera1.nsf
SendTo = Test UserA5,Test UserA12
Save = Yes
SendSubject = Mail from Test Mail Activity by ProActive Load
Text1 = This document was sent to you from the ProActive Load
Text2 = by G2 Associates testing Notes R5.<ENTER>
Text3 = The Mail Activity composed this memo and sent it to<ENTER>
Text4 = recipients specified in the scenario file, PROACTIV.INI.
Attach = c:\program files\patools\load\Proactiv.ini

ProActive Assistant Scenario Example:

[TestMail]

Activity = Mail
DbName = Mail\tusera1.nsf
SendTo = Test UserA5,Test UserA12
Save = Yes
SendSubject = Mail from Test Mail Activity by ProActive Assistant
Text1 = This document was sent to you from the ProActive Assistant
Text2 = by G2 Associates testing Notes R5.<ENTER>
Text3 = The Mail Activity composed this memo and sent it to<ENTER>
Text4 = recipients specified in the scenario file, PROACTIV.INI.
Attach = c:\program files\patools\load\Proactiv.ini

Read Documents Activity

The Read Documents Activity opens a Notes database, reads documents, and closes the database. This activity places a relatively light load on the server, and is intended to simulate actual users browsing their mail or any database.

Events

For each iteration:

Open DB - Opens the selected database

Display - Displays one document for each of NumberOfDocuments parameter

Page Down - Displays the next page for each of PageDown parameter

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	User in Assist?	Used in Load?
Activity	Read Documents		Yes	Yes
DBName	Database name to use In PA Load, DBName can be thread specific	From General	Yes	Yes
NumberOf Documents ²	Number (or numeric range) of documents to read in one iteration	10	Yes	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
PageDown	Number (or numeric range) of pages to display after opening the document	n/a	Yes	No
ReadTime	Number (or numeric range) of seconds to wait after the document is opened (simulating reading time)	10	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	User in Assist?	Used in Load?
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
Server	Lotus Domino server	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes.	Yes
View ¹	View to open. If the view is in a secondary menu, use the primary menu name followed by a backslash (\) then the secondary menu name, for example, Main view\details.	From General	Yes	Yes

¹ In ProActive Load, a valid view must be included in the activity. Failure to do so may open double sessions on the Notes server.

² NumberOfDocuments (this parameter is one word).

The example activity below can be used in both Assistant and Load.

[ReadDocs]

Activity	= Read Documents
RunID	= ReadDocs
DBName	= Sample.nsf
View	= \$All
NumberOfDocuments	= 5
ReadTime	= 4-20

Rebuild View Activity

The Rebuild View Activity rebuilds the current view or all views for a Notes database. For large databases, this activity can place a significant load on the server and invokes the indexer task.

Events

For each iteration:

Open DB - Opens the selected database

All Views - Rebuilds all views if RebuildAll is Yes

Select View - Selects a view if View is not blank and RebuildAll is No

Current View - Rebuilds current view if RebuildAll is No

Close DB - Closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Rebuild View		Yes	Yes
DBName	Database name to use In PA Load, DBName can be thread specific	From General	Yes	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RebuildAll	Yes rebuilds all views in the database No rebuilds the current view only	No	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
Server	Lotus Domino server	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes
View	View to open. If the view is in a secondary menu, use the primary menu name followed by a backslash (\) then the secondary menu name for example, Main view\details.		Yes	Yes

The example activity below can be used in both Assistant and Load.

[RBView]

Activity = Rebuild View

RunID = RBView

DbName = TestDiscuss.nsf

RebuildAll = Yes

Reset Result Activity

The Reset Result Activity causes the currently open test session to be closed and a new test session started and to record the events and activities of the ProActive Assistant scenario. This new test session can be included in the existing Result File, or a new Result File can be created. The Reset Result Activity can only be used with ProActive Assistant.

Since the Reset Result Activity manipulates the structure of the Result File and does not execute any Notes functions, it does not cause any entries to be made in the Result File itself. It is only used to manage and structure the Result File for analysis and reporting.

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Reset Result		Yes	No
MailDBName	Name of database from which to mail the Results file. (Only if MailResultNow=Yes)	From General	Yes	No
MailResultNow	Yes will cause the current Result file to be mailed No will not mail current Result file	No	Yes	No
MailResultsTo	Recipient of Results File (Only if MailResultNow=Yes)	From General	Yes	No
MailServer	Name of server from which to mail the Results file	From General	Yes	No

Parameters used in test scenarios (continued):

Keyword	Value	Default	Used in Assist?	Used in Load?
NewResultFile	Yes causes new test session to be started in a separate Result File. No causes the new test session to be appended to the existing Result File. This would be the same as stopping and restarting the test.	No	Yes	No
ResultFile	Path and name of the Result File Default is c:\work\proactiv.csv	From General	Yes	No
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	No

ProActive Assistant Scenario Example:**[NewResult]**

Activity = Reset Result

The above activity will start a new test session within the current Result File.

[NewResultI]

Activity = Reset Result

NewResultFile = Yes

The above activity will start a new test session in a new dated Result File.

[NewResultI]

Activity = Reset Result

NewResultFile = Yes

MailResultNow = Yes

The above activity will mail the current Result File before the new one is started.

User Defined Activity

The User Defined Activity performs a series of commands in a user-defined sequence. Depending on how the activity is defined, different aspects of the environment can be stressed. This can be a very powerful feature, which is typically used to stress custom applications.

User Defined activities are totally free form. They do not have built in events, such as Open Database, as do the predefined activities. This flexibility allows for greater customization, but also requires more attention to the structure of the Notes application being tested.

User Defined can be used to perform any of the following Events. The order depends on the sequence of commands.

- Attach File
- Close Document
- Delete Document
- Edit Document
- Enter Text
- Navigate
- Open Database
- Open Document
- Open View

User Defined - Standard Keywords with General Defaults:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	User Defined		Yes	Yes.
ActivityName	Name of Activity to be displayed in the Activity column of the Result file	n/a	Yes	Yes
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	Yes
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	Yes
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	Yes
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	Yes
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	Yes

As the name implies, User Defined activities are highly customizable by the user. The keywords shown in the table above are typically located at the top of the User Defined activity and apply to the whole activity. The syntax for the keywords is the same as other activities, as shown below.

Keyword = Value

The user-defined activity consists of a series of Commands. Most commands are preceded by the word "Command" as shown below. In some cases commands are followed by one or more parameters in the form Parameter Name = Value. The example below shows an Open Database command followed by related parameters and values.

Command = Open Database
Server = TestServerA1
View = TestView
DBName = TestDatabase.nsf

Commands are executed in the order entered.

User Defined Commands:

Command Name	Parameter Name	Parameter Value	Default	Used in Assist?	Used in Load?
Action	Button	Name or ordinal number of button to push in a form.		Yes	No
Attach File	Attach	File name to attach		Yes	Yes
	Compress	Yes compresses the attachment No does not compress the attachment	No	Yes	Yes
Close All Windows	None			Yes	No
Close Database	EmptyTrash	Yes deletes documents No does not delete documents	No	Yes	Yes
Close Document	Encrypt	Yes encrypts the document No does not encrypt the document	No	Yes	Yes
	Save	Yes saves the document No does not save the document	No	Yes	Yes
	Send	Yes sends the document No does not send the document	No	Yes	Yes
	Sign	Yes signs the document No does not sign the document	No	Yes	Yes
Copy Document (copies document at current cursor position)	CopyToServer	Server containing the database where document is to be copied.		No	Yes

User Defined Commands (continued):

Command Name	Parameter Name	Parameter Value	Default	Used in Assist?	Used in Load?
	CopyToFile	Database on above server where document is to be copied		No	Yes
Delete Document	None	Marks document at current cursor position for deletion. Document will be deleted when database is closed		Yes	Yes
Enter Text	Text{n}	Text parameters		Yes	Yes
Lookup ¹	Key	Value in left column of view specified immediately preceding the Lookup command. Locates document for Lookup.		No	Yes
	Field ³	Field for Lookup		No	Yes
Navigate (only valid in a view)	Down	# of documents to go down in the view Can be a number or a range		Yes	Yes
	NavigateOption	Bottom goes to the bottom of the view		Yes	Yes
		Collapse All Collapses all categories		Yes	No
		Expand All Expands all categories		Yes	No
		Top goes to the top of the view		Yes	Yes
	Up	# of documents to go up in the view Can be a number or a range		Yes	Yes
Open Database ²	DBName ²	Filename of database to open		Yes	Yes
	Server ²	Server name of database to open		Yes	Yes
	View ²	Name of view		Yes	Yes

User Defined Commands (continued):

Command Name	Parameter Name	Parameter Value	Default	Used in Assist?	Used in Load?
Open Document (opens document at current cursor position)	Edit	Yes Opens the document in edit mode No Opens the document in read mode		Yes	Yes
Open Form	Form	Form name		Yes	No
Open View	View	View name		Yes	Yes
Sleep	SleepTime	Seconds or numeric range of seconds		Yes	Yes
	SleepUntil	Next <interval> on <minute>		Yes	Yes
Switch ID	IDFile	File name of Notes ID file		Yes	No
Switch Location	Location	Location name		Yes	No

¹ Command used to simulate a Notes DBLookup.

² Open Database can be used to open a database and leave it open for use by other activities. When this is done, the database will not be reopened (or closed) by these activities. It will be closed with another User Defined Activity, either closing the database or opening another one.

³ The value of this field that was located via Command = Lookup, can be entered into another field in a document by using the <Lookup> text tag in ProActive Load. An example of this syntax would be as follows:

Command	=	Lookup
Field	=	DataField
Key	=	MyDocument
Field	=	NewField = <Lookup>

In this case, the NewField field is in the same document as DataField. After completing the Lookup command, you could move to another document before using the <Lookup> text tag. This data will continue to be available in this scenario until the next Lookup command is executed for this thread.

User Defined – Field Parameters for Assistant:

Type Field	Field Name	Value - ASSISTANT ONLY
Field	FieldName	To enter values into text fields, use the syntax Field=FieldName=Value

The Field parameter shown above must be preceded by “Command = Enter Text.” For entering text in a document, the “Command = Enter Text” must be preceded by a "Command = Open Document" and "Edit = Yes".

ProActive Assistant enters data through the Notes UI. This requires FieldNames specified in the Field parameter to exist on the form being used for the edited document. Data types are handled by the form used by the open document, and don't need to be specified.

User Defined – Field Parameters for Load:

Type Field	Field Name	Value - LOAD ONLY
Field	FieldName	<p>To enter values into text fields, use the syntax Field=FieldName=Value</p> <p>Two special text fields for Load.</p> <p>Multivalue with semicolon as separator Field = TextList = “one;two;three;four;five”</p> <p>Empty Field Field = EmptyField =</p> <p>(Where TextList and EmptyField are the actual names of the fields)</p>
FieldN	FieldName	<p>To enter values into numeric fields for Load, use the syntax FieldN=FieldName=Value</p> <p>Valid: FieldN = NUMBER1 = 123.456 FieldN = NUMBER2 = 123,456 FieldN = NUMBER3 = 1</p> <p>Invalid: FieldN = NUMBER1 = 1.2.3</p> <p>Three special number fields for Load</p> <p>Number Random Field = NumberRandom = <Range 10-99>.<Range 1-9></p> <p>Number List with semicolon as separator Field = NumberList = “1;2;3;4;5”</p> <p>Number Null Field = NumberNull =</p> <p>(Where NumberRandom, NumberList, and NumberNull are the actual names of the fields)</p>

User Defined – Field Parameters for Load (continued):

Type Field	Field Name	Value - LOAD ONLY
FieldTD	FieldName	To enter values into date fields, use the syntax FieldTD=FieldName=Value Valid: FieldTD = TIME1 = 5/25/99 FieldTD = TIME2 = 4/1/00 23:59 FieldTD = TIME3 = 5/23/01 10:30 PM Invalid: FieldTD = TIME1 = 24/5/99 10:10 FieldTD = TIME2 = 5/25/99 25:23

Since ProActive Load is not entering data through the Notes UI, there are some special conventions that must be observed when entering data with the Field Parameters.

To create a field with a null value, enter the field parameter without a value.

Example: Field = FieldName =

When entering multiple values into a multivalue field, the values must be separated by semicolons and the entire string must be enclosed in quotes.

Example: Field = FieldName = “one;two;three;four”

Numeric, time, and date values entered into fields with the “Field = FieldName =” syntax will be entered as text. To enter data of a specific non-text data type the FieldN and FieldTD syntax must be used.

The example activity below can be used in both Assistant and Load.

[ReadEdit]

Activity	= User Defined
RunID	= ReadEdit
DBName	= TestData.nsf
Command	= Open Database
View	= \$View
Command	= Navigate
NavigateOption	= Bottom
Command	= Navigate
Up	= 6
Command	= Open Document
SleepTime	= 10-30
Command	= Close Document
Command	= Navigate
Down	= 1
Command	= Open Document
Edit	= Yes
Command	= Enter Text 'Optional line for Load, Required for Assistant
Field	= Subject = Document Edited by Thread No. <thread>
Field	= Field1 = Enter New Data
Command	= Close Document
Save	= Yes
Command	= Close Database

Views Activity

This activity opens a Notes database, locates all of the available views, then cycles through each view. This activity places a relatively light load on the server (depending upon the size of the database and number of views).

Events

Open DB – Opens the selected database

For each iteration:

Select View – Selects one view from all available views

Page Down – displays the next page for each page Down Parameter

Following the final iteration:

Close DB – closes the database

Parameters used in test scenarios:

Keyword	Value	Default	Used in Assist?	Used in Load?
Activity	Views		Yes	No
DBName	Database name to use	From General	Yes	No
NumSets	Number (or numeric range) of times to run a count activity	From General	Yes	No
PageDown	Number (or numeric range) of pages after opening the view		Yes	No
RunID	Value to be placed in the RunID column of the Results file for this activity	From General	Yes	No
RunType	Count runs for NumSets iterations Timed runs for TestDuration minutes	From General	Yes	No
Server	Lotus Domino server name No does not encrypt the document	From General	Yes	No
SleepTime	Number (or numeric range) of seconds to sleep between activity iterations	From General	Yes	No
TestDuration	Number (or numeric range) of minutes to run a timed activity	From General	Yes	No

Text Parameters

Since one of the primary differences between ProActive Assistant and ProActive Load is the use of the Notes User Interface by ProActive Assistant, the use of text parameters is very different between the two products. For this reason, the topic is divided into two separate sections, each dealing with the use of text parameters for the specific products.

In all cases, a number may be appended to the Text keyword to enhance readability of successive text lines in the PROACTIV.INI file (for example, Text1= Text2=, etc.). These optional characters are ignored during the execution of the test scenario.

Text Parameters in ProActive Load

In ProActive Load, Text parameters are only used to specify text to be entered into Notes documents. Optionally, you can append a number to the Text keyword. The value of the Text parameter simulates a user typing at the keyboard. In addition to letters, numbers and spaces, special keys may be represented by using the following sequences:

Special key	Key Sequence	Example
ENTER	<ENTER {n}>	<Enter 3>
TAB	<TAB {n}>	<TAB 4>

Where {n} is an optional repeat count indicating the number of times the key is to be pressed. The two parameters above can only be used in rich text fields for formatting purposes. The <ENTER> parameter creates a carriage return and the <TAB> creates a tab.

Special Text Tags

The following special tags may be placed anywhere where the "Text =" parameter is valid.

<DATE> - Inserts the current date.

<ITERATION> - Inserts the current iteration. .

<ITERATION {+ or - number or range}> - Inserts the current iteration (optionally modified by a number or range of numbers).

For example, <Iteration + 10 - 20> will insert a string representing the current iteration plus a random number between 10 and 20.

<LOOKUP> - Inserts the value from the most recent LOOKUP command that was executed by a User Defined activity. The value of the <Lookup> tag is thread specific. .

<RANGE n – n1> - Inserts a random number between n and n1. .

For example <Range 1- 60> would generate a random number between 1 and 60.

<TIME>- Inserts the current time. .

<THREAD>- Inserts the thread number. For more information, see the sections listed below. .

- Assigning DBName to Specific Threads
- Nested Substitution Within the @Variable Functions
- Generalized Thread Substitution

<p>Note: In ProActive Load, if two text parameters are used together, such as <time> <date>, be sure they are separated by a space.</p>
--

Text Parameters in ProActive Assistant

In ProActive Assistant, Text parameters have several purposes, the primary being to specify text to be entered into Notes documents. In addition, the Text parameters are used to navigate around the Notes User Interface via the pressing of keys on the keyboard. Optionally, you can append a number to the Text keyword to enhance readability of successive text lines in the Scenario file (for example, Text1 Text2, etc.). These optional characters are ignored.

At the end of this section, there is a simple example that demonstrates both the UI navigation and data input.

The Text parameters are used to simulate a user typing at the keyboard. In addition to letters, numbers and spaces, special keys may be represented by using the following sequences:

Text Parameters in ProActive Assistant:

Special key	Key sequence	Example
ENTER	<ENTER {n}>	<Enter 3>
TAB	<TAB {n}>	<TAB 4>
UP ARROW	<UP {n}>	<Up 5>
DOWN ARROW	<DOWN {n}>	<Down 5>
LEFT ARROW	<LEFT {n}>	<Left 10>
RIGHT ARROW	<RIGHT {n}>	<Right 10>
ESCAPE	<ESC {n}>	<Esc>
SPACE BAR	<SPACE {n}>	<SPACE 20> ¹
PAGE UP	<PGUP {n}>	<PGUP 2>
PAGE DOWN	<PGDN {n}>	<PGDN 2>
HOME	<HOME {n}>	<Home>
DEL	<DEL {n}>	<DEL 5>
BACKSPACE	<BS {n}>	<BS 10>
INSERT	<INSERT>	<Insert>

Text Parameters in ProActive Assistant:

Special key	Key sequence	Example
BREAK	<BREAK>	<BREAK>
END	<END>	<END>
CLEAR	<CLEAR>	<CLEAR>
CAPSLOCK	<CAPSLOCK>	<CAPSLOCK>
NUMLOCK	<NUMLOCK>	<NUMLOCK>
NUMERIC KEYPAD	<NUMPADx {n}> where x is 0123456789/-+*.	<Numpad0 10>

¹ Spaces can also be entered by simply inserting spaces in the text.

{n} is an optional repeat count indicating the number of times the key is to be pressed.

Key Modifiers

Any of the above keys can be modified with CTRL, ALT and/or SHIFT to simulate multi-key combinations. For example:

<Ctrl Home>, <Alt F4>, <Ctrl Shift F9>

Special Text Tags

The following special tags may be placed anywhere where the "Text=" parameter is valid.

<**ACTION** *ActionButtonName or Number*> - Presses a button named *ActionButtonName* or the ordinal number (from the left).

For example <Action 3> presses the third button from the left, <Action Send> presses the Send button.

If the Action button contains cascaded selections, as in the "Reply" button in the Notes R5 mail template, the ordinal number of the selection must be used with the name of the button. For example, <Action Reply\2> would be used to choose the "Reply with History" selection, which is the second choice under Reply.

<**ATTACH**>- attaches the file in the Attach = parameter in place of <attach> tag .

<DATE> - Inserts the current date.

<ENDEVENT> - Ends the most recent user defined event.

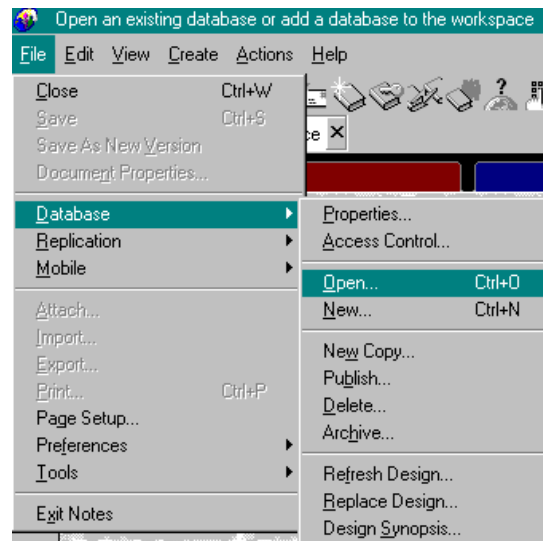
(Within the activity, the number of EndEvents should match the number of StartEvents)

<ITERATION> - Inserts the current iteration.

<ITERATION {+ or - number or range}> - Inserts the current iteration (optionally modified by a number or range of numbers). For example, <Iteration + 10 - 20> will insert a string representing the current iteration plus a random number between 10 and 20.

<MENU *MenuItem*> - Activates the menu *MenuItem*. To activate a submenu, separate the menus with backslashes (\).

For example <Menu File\Database\Open> presents the Open Database dialog box in the standard Notes mail file, as shown in the example below.



PAUSE {milliseconds}> - delays in place of the <pause> tag.

If milliseconds is numeric, delays that number of milliseconds.

If milliseconds is omitted or not numeric, delays MaxWait milliseconds. Use this parameter to slow down the execution of the Scenario.

<STARTEVENT *EventName*> - Starts a user defined event named *EventName*. .

<TIME> - Inserts the current time.

Additional operations that are used for entering text

To press a button on a form, use the navigation keys described above to place the cursor on the desired button, then use <space> (or a single space) to simulate the user pressing the space bar to activate the button.

For example: **Text = <Down 5><Space>** presses the down key five times (to position the cursor over the button and then presses the button with *Space*, which simulates pressing the space bar.

To press the OK button in a dialog box.

In Notes R4 the OK button was usually indicated as **OK**, so entering <Alt O> would usually press the button. In Notes R5 the “O” is not indicated with an underline, so the TAB key must be used to move to the button which would then be pressed by <Space> or <Enter>.

Text = [fieldname] = value

Optionally, a fieldname can be included. ProActive Assistant will go to the named field before inserting the text. If fieldname is omitted, text is inserted in the current field.

This feature can be used with the Compose, Edit, and User Defined Activities in ProActive Assistant only.

Field = [fieldname] = value

In ProActive Assistant, the function works similar to the Text=parameter, but instead of inserting the text, it executes a “FieldSetText.” This can be used, for example, to set a keyword field to one or more named values. For example, the Phone Message form in the Notes Mail template has a checkbox keyword field with several named values (Telephoned, urgent, Please Call, Returned Your Call, Will Call Again). This is easier than using the Text=parameter to simulate the user navigating to each checkbox and pressing the spacebar to select it. This parameter will permit you to specify Field = PhoneReason = Telephoned, Urgent, Please Call, Returned Your call. This will set the appropriate four checkboxes. If fieldname is omitted, the value is set in the current field.

This function does not support the entry of rich text. If you need to enter rich text, use the **Text = [fieldname] = value** function.

FieldSize = [fieldname] = # of bytes (number or range of numbers)

Makes the field length equal to the specified number of bytes. If the field length is greater than the number of bytes, the field is truncated. If the field length is less than the number of bytes, the original field value is repeatedly concatenated to the field until the specified number of bytes is reached.

For example: Text = This is a test.
 FieldSize = Body = 100
 would yield "This is a test.This is a test.This is a test.This is a test.This is a test.This is a test.This is a "

If the fieldname is omitted, the current field is used.

This feature can be used with the Compose, Edit, Mail, and User Defined Activities in both ProActive Assistant and ProActive Load.

Text Parameters Example

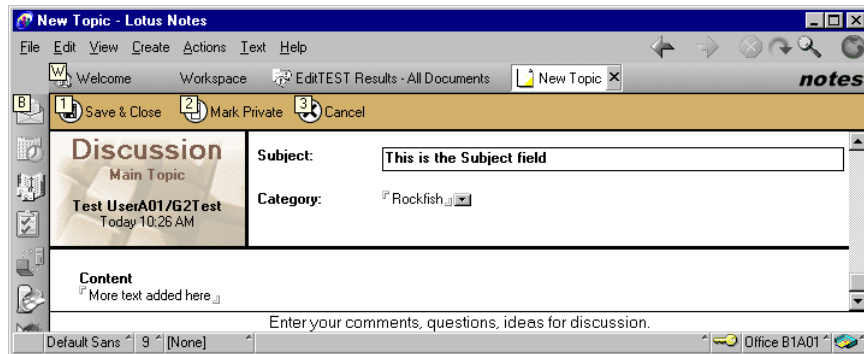
The example shown below uses a document in the standard Notes R5 Discussion database. It assumes that the cursor is in the subject field and using the following text parameters it will navigate among three fields and input some data.

Text1 = This is the Subject field <Tab>

Text2 = Rockfish <Tab>

Text3 = More text added here

Text4 = <Action 2>



When the scenario runs, ProActive Assistant will place the text “This is the Subject field” in the Subject field. The <Tab> moves the cursor to the next field and the text “Rockfish” is entered. The <Tab> will move the cursor to the body field where the Text3 entry will insert the text “More text added here.” The Text4 entry pushes the action button, Mark Private on the action bar. This action button could also be pushed by specifying its name, as shown below.

Text4 = <Action Mark Private>

<Action> is considered to be a “Special Tag” and is covered in a previous part of this Text Parameters section.

Invoke Method Parameters

Invoke Method Parameters, which are only supported in the Compose, Edit, and User Defined Activities of ProActive Assistant, provide another way of executing functions in the Notes UI without relying on duplication of individual keystrokes. These parameters rely on the use of OLE and tend to be more reliable and predictable than the normal keystrokes. The Invoke Method parameters can only be used in a currently open Notes document, and are supported anywhere in a ProActive Assistant scenario where the “Text= “ parameter is supported. Multiple Invoke Method parameters may be mixed with any other parameters that are also supported with the Text Parameters.

The syntax for using Invoke Method is “InvokeMethod = [MethodName],” the list of valid MethodNames is shown in the table below.

Invoke Method Parameters:

Method Name	Description	Methods Supported by Activity		
		Compose	Edit	User Defined ³
Clear ¹	Clears the selected text.	Yes	Yes	Yes
Close	Closes the current document.	No	No	Yes
CollapseAllSections	Collapses all sections in the document.	Yes	Yes	Yes
Copy ¹	Copies the selected text to the clipboard.	Yes	Yes	Yes
Cut ¹	Cuts the selected text and places it on the Clipboard.	Yes	Yes	Yes
DeselectAll	Deselects the entire contents of the current field.	Yes	Yes	Yes
ExpandAllSections	Expands all sections in the document.	Yes	Yes	Yes
GoToBottom	Moves cursor to the last field in the document.	Yes	Yes	Yes

Invoke Method Parameters (continued):

Method Name	Description	Methods Supported by Activity		
		Compose	Edit	User Defined ³
GoToNextField	Moves cursor to next field in the document.	Yes	Yes	Yes
GoToPrevField	Moves cursor to previous field in the document.	Yes	Yes	Yes
GoToTop	Moves cursor to first field in the document.	Yes	Yes	Yes
Paste ²	Pastes Clipboard text into the current field.	Yes	Yes	Yes
Refresh	Refresh the document.	Yes	Yes	Yes
Reload	Refreshes the current document with any changes made to the stored document.	No	No	Yes
Save	Saves the current document.	No	No	Yes
SelectAll	Selects the entire contents of the current field.	Yes	Yes	Yes
Send	Mails the current document to addressees in SendTo field	No	No	Yes

¹ Before these parameters can be used, the data in the current field must be selected with “InvokeMethod=SelectAll” or “Text=<ctrl a>.” “InvokeMethod=SelectAll” is not supported in fields in Notes documents where “Control a” is not supported, e.g., Subject and Categories in documents created with the R5 Discussion template.

² If you wish to use this parameter to replace the data in the current field, the field must be blank or the data selected with “InvokeMethod=SelectAll” or “Text=<ctrl a>.” Otherwise, the pasted data will be prepended to the data currently in the field.

³ When used in a User Defined Activity, “Command = Enter Text” followed by the “Text =” parameter must precede any use of the Invoke Method parameters.

General notes: The methods that are not supported in Compose and Edit are usually due to that event already being built into these Activities and automatically performed by them, e.g., document close. Attempting to also do this with Invoke Method will result in conflicts.

To use Invoke Method, you normally have to have navigated to the appropriate field in the User Interface. In addition to some Methods that are used for navigation, you can use Text= parameters, GoToField=, Field=[fieldname]=, and Text=[fieldname]=.

Appendix B – Sample Scenarios files

This appendix contains two sample scenario files, one for ProActive Assistant and one for ProActive Load. Each of these file contains several similar activities.

Also provided with the ProActive Tools software is a Scenario Library contained in a Lotus Notes database, PATScenarioLibrary.nsf. This database contains sample scenarios for all of the predefined ProActive Load and ProActive Assistant Activities. The Scenario Library also contains usable copies of the samples shown in this appendix.

ProActive Assistant Sample Scenario File

```
*****
' This is a long Ini file that contains samples of all the activities.
' You will probably need to make modifications to this Ini file
' before it will run correctly in the local environment.
*****
' PAASAMPLE.INI - Parameter file for ProActive Assistant
' Valid Activities:
' -----
' Calendar
' Compose
' Copy Database
' Edit
' Find
' Index
' Mail
' Read Documents
' Rebuild View
' Reset Result
' User Defined
' Views
*****
```

[General]

RunID	= Sample	
Path	= c:\notes	
Client	= @notes.ini.keyfilename	' get user name from Notes.ini
Server	= @Notes.ini.mailserver	' user's mail server by default
RunType	= Count	' Valid types are Timed or Count
NumSets	= 1	
CloseNotesWhenDone	= No	' Do not terminate Notes after
completing session		
MaxWait	= 500	' Max time (mseconds) to wait for idle
ResultFile	= c:\Work\ProActiv.CSV	
MailResults	= No	' do not send result file as attachment
MailResultsTo	= Joe Administrator	
MailDBName	= @notes.ini.mailfile	' Send from this database
MailServer	= Local	
DeleteResultFile	= No	' Delete file from disk after
		' successfully sending the result
'StartTime	= 17:30	' Start the session at 5:30 PM
'MaxStartWait	= 5	' If ProActive Assistant was
		' initiated after 5:30 PM, but before
		' 10:30 PM, begin immediately; if after
		' 10:30 PM, wait until 5:30 PM tomorrow
		' Set to Yes to run in testmode
TestMode	= No	
NotesEXEName	= Notes.exe	
DBName	= @notes.ini.mailfile	' Default mail database

' Activity Set Section

[Activities]

Call	= ReadDocs	' read mailfile
SleepTime	= 5	
Call	= ChangeViews	' in NAB
SleepTime	= 5	
Call	= TestMail	' in mailfile


```

SleepTime      = 5
Call           = TestCopy           ' from Assistant Help
SleepTime      = 5
Call           = TestIndex          ' in Assistant help
SleepTime      = 5
Call           = RebuildViews        ' in mailfile
SleepTime      = 5
Call           = Find1               ' in Assistant Help
SleepTime      = 5
Call           = ComposeDocs         ' in mailfile
SleepTime      = 5
Call           = ComposePhone        ' using Phone message form
Do for 5 minutes
SleepTime      = 5
Call           = Calendar            ' Calendar Activity
SleepTime      = 10
Call           = User Reply          ' User defined activity
End

```

' Activity Routines Section

[ReadDocs]

```

' Read documents from this workstation's mailfile for the default number of minutes
  Activity      = Read Documents
  NumberOfDocuments = 5           ' Read 5 documents per iteration
  ReadTime      = 4               ' Sleep 4 seconds between documents
  View          = All by Date     ' use this view

```

[RebuildViews]

```

' Rebuild current view of the default database for the default number of minutes
  Activity      = Rebuild View
  View          = ($All)          ' rebuild this view

```

[ChangeViews]

```

' Cycle through the views of the default database
  Activity      = Views
  DBName        = Names.nsf

```

[TestCopy]

' Copy Sample.nsf on the default server to a local copy. Copy both design
' and documents. Don't copy the access list. Delete the copy after completing the activity.

Activity	= Copy Database	
DBName	= Sample.NSF	' copy the Sample database
CopyToFile	= TestCopy.NSF	' to TestCopy.nsf
CopyToServer	= local	' on local
CopyTitle	= G2 Test Copy Database	' with this title
CopyDesignOnly	= No	' copy the design and documents
CopyAccessList	= No	' but not the ACL
DeleteCopy	= Yes	' Delete the copy when the activity completes
ReplaceCopy	= Yes	' Replace copy if DB exists
NumSets	= 2	' run 2 times
SleepTime	= 10	' sleep 10 seconds between iterations

[Find1]

' Find strings in the Sample database for the default number of minutes

Activity	= Find
DBName	= Sample.nsf
View	= Intro PATools
FindString	= ProActive Assistant
FindString	= Activities

[TestIndex]

' Do full text index of default database for the default number of minutes

Activity	= Index
DBName	= Sample.nsf

[TestMail]

' Mail documents from the workstation's mailfile for the default number of minutes.

' Works with standard Notes mail template

Activity	= Mail	
SendTo	= Test UserA05, Test UserA09; Save=Yes	
SendTo	= Group2A	
Sign	=Yes	' to each SendTo recipient
SendSubject	= Mail from the ProActive Assistant	
Text	= This document was sent to you from the ProActive Assistant	

Text2 = by G2 Associates.<ENTER>"
Text3 = The Mail Activity composed this memo and sent it to<ENTER>
Text4 = recipients specified in the parameter file, PROACTIV.INI.
SleepTime = 10

[ComposeDocs]

' Compose and send documents from the workstation's mailfile for the default number
' of minutes.

Activity = Compose
Form = Memo
Save =Yes
Sign =Yes ' Send, save, & sign
Text1 = Test UserA22<tab> ' Send to field
Text2 = Test UserA21<tab 2> ' cc field
Text3 = ProActive Compose ' Subject field
Text4 = <TAB> ' Tab to Body
Text4 = This memo was composed by the ProActive Assistant<ENTER>
Text5 = Compose Activity. This memo should also be saved, and signed.

[ComposePhone]

' Compose a Phone Message in Mail Template

Activity = Compose
Form = Special\Phone Message
Save =Yes
Sign =Yes
Text1 = Test UserA11<TAB 3> ' To field, tab to the Contact field
Text2 = Someone important<TAB> ' Contact field
Text3 = Important company Inc.<TAB> ' Company field
Text4 = 301-555-1212<TAB 2> ' Phone# field, tab over the fax field
Text5 = <DOWN> <DOWN 7> <TAB> ' one space checks the telephoned
check box
' Down to the Please call check box
' one space checks that check box
' Down 7 times to the Urgent check box
' one space checks that check box
' Tab to the message field
Text6 = This will test the special phone message,<ENTER>
Text7 = which is a form in the mail databases.

[Calendar]

' This activity will create calendar entries for the default number of minutes
' using the user's mail file.

Activity	= Calendar	
DBName	= @Notes.ini.mailfile	
SendInvitation	= Yes	' send invitations
CalendarType	= Appointment	' create an appointment
Description	= Calendar Activity	' short description of appointment
Date	= 1/5/2000	' Date of appointment
Time	= 10:00 AM - 12:00PM	' from 10 to noon
PencilIn	= Yes	' doesn't affect freetime
KeepPrivate	= Yes	' keep it private
Text1	= This is the detailed description field of an appointment.	
CalendarType	= Invitation	' create an invitation
Description	= Test invitation	' short description of invitation
Date	= 1/6/2000	' date of invitation
Time	= 9:00 AM - 10:00 AM	' time of invitation
PencilIn	= No	' affect freetime
Text1	= This is the detailed description field of an invitation.<ENTER>	
RequiredAttendees	= Test UserA02	' send invitations to
OptionalAttendees	= Test UserA22	' these individuals
FindFreeTime	= Yes	' push the FindFreeTime button
CalendarType	= Event	' create an event
Description	= Lotusphere (Event)	' short description of event
Date	= 1/25/2000	' starting date
Duration	= 2 – 6	' random date from 2 to 5 days
CalendarType	= Reminder	' create a reminder
Description	= Test Reminder	' short description of reminder
Date	= 1/13/98	' date of reminder
Time	= 7:00 AM	' time of reminder
CalendarType	= Anniversary	' create an anniversary
Description	= Test Anniversary	' short description of anniversary
Date	= 1/15/2000	' date of anniversary

[UserReply]

Activity	= User Defined
RunID	= UserReply
Command	= Open Database

Command	= Sleep	
SleepTime	= 5 – 20	' sleep for random time between 5 and ' 20 seconds
Command	= Navigate	
NavigateOption	= Top	' navigate to the Top of the view
Command	= Navigate	' then down 2 documents
Down	= 2	
Command	= Open Document	' open the current document
Command	= Action	' Press the Reply with History button
Button	= Reply\Reply with History	
Command	= Enter Text	' and compose a reply
Text	= This is a reply to a memo composed by the	
Text	= Enter Text Command of the<ENTER>	
Text	= User Defined Activity of ProActive Assistant.	
Command	= Close Form	' close reply form
Save	= Yes	' save it
Send	= Yes	' send it
Sign	= Yes	' sign it
Command	= Navigate	
Down	= 2	' go down 2 documents in the view
Command	= Open Document	' and open the current document
Command	= Sleep	
SleepUntil	= Next 1	' then sleep until the next even minute
Command	= Close Document	' close the document
Command	= Close Database	' close the database

ProActive Load Sample Scenario File

```
' *****
' This is a long Ini file which contains samples of all the activities.
' You will probably need to make modifications to this Ini file
' before it will run correctly in the local environment.
'
' *****
' PALSAMPLE.INI - Parameter file for ProActive Load
'
' Valid Activities:
' -----
' Calendar
' Close Database
' Compose
' Edit
' Find
' Mail
' Open Database
' Read Documents
' Rebuild View
' User Defined
'
```

[General]

```
RunID      = Sample
Path       = c:\notes\data
Client     = @notes.ini.keyfilename      ' get user name from Notes.ini
Serve      = @Notes.ini.mailserver      ' user's mail server by default
RunType    = Count                      ' Valid types are Timed or Count
NumSets    = 1
SleepTime  = 10

ResultOutput = Session, Sleep, Activity, Event ' result items to be captured
ResultFile   = c:\Work\ProActiv.CSV
TimeLineFile = c:\Work\TimeLine.CSV
ErrorFileName = c:\Work\Errorlog.lst

'StartTime   = 14:30
'MaxStartWait = 5
```

Random	= Yes
ListNoteID	= Yes
TestMode	= No
Threads	= 5
ThreadDelay	= 1
FirstThread	= 1
DBName	= mail\tusera<thread>.nsf
Thread1	= Activity1
Thread2-3	= MailCal

' Activity Sets Section

[Activity1]

Do for 10 minutes		
Call	= ComposePhone	' using Phone message form
SleepTime	= 1-10	
Call	= RBView	' Rebuild all views in mailfile
SleepTime	= 10-20	
Call	= EditDocs1	
End		

[MailCal]

Do 2 Times		
Call	= TestMail	' Send Mail
SleepTime	= 1-10	
Call	= Calendar1a	' Invitation
Call	= Calendar1b	' Event
Call	= Calendar1c	' Anniversary
SleepTime	= 10-20	
Call	= MailRead	' Read Mail
End		

[Activities]

' Threads without designated Activity Sections will default to this one

```
Call          = ReadDocs          'read mailfile
SleepTime     = 5
Call          = Find1             'in Sample
SleepTime     = 5
Call          = ComposeDocs       ' in mailfile
SleepTime     = 5
```

' Activity Routines Section

[RBView]

```
Activity      = Rebuild View
RunID         = RBView
NumSets       = 2                'RunType already specified in [General]
RebuildAll    = Yes
'-----
```

[ReadDocs]

' Read documents from this workstation's mailfile for the default number of minutes

```
Activity      = Read Documents
RunID         = ReadDocs
NumSets       = 3
NumberOfDocuments = 5           ' Read 5 documents per iteration
ReadTime      = 4-20            ' Sleep 4 seconds between documents
View          = $All
'-----
```

[Find1]

' Find strings in the default database for the default number of minutes

```
Activity      = Find
RunID         = FindHelp
NumSets       = 3
DBName        = Sample.nsf      ' Database must be full text indexed
FindString    = User            ' find these three strings in
succession
FindString    = Database
FindString    = Activity
'-----
```


[TestMail]

'Mail documents from the mailfile for each thread

'Works with standard Notes mail template

Activity = Mail
RunID = MailSend
NumSets = 4
Send = Yes
Sign = Yes
SendTo = Test UserA10; Save = Yes ' send a separate document
SendTo = Group2A ' to each SendTo recipient
SendTo = Names<select 5> ' select 5 random names from NAB

SendSubject = Mail from the ProActive Load
Text1 = This document was sent to you from the ProActive Load
Text2 = by G2 Associates.<ENTER>
Text3 = The Mail Activity composed this memo and sent it to<ENTER>
Text4 = recipients specified in the parameter file, PROACTIV.INI.
'-----

[ComposeDocs]

'Compose and send documents from mailfile for each thread

Activity = Compose
RunID = CompDoc
NumSets = 3
Form = Memo
Save = Yes
Sign = Yes
Sendto = Test UserA39 ' Send to field
SendSubject = Test ProActive Compose Activity ' Subject field
Text1 = This memo was composed by the ProActive Load"
Text2 = Compose Activity. This memo should also be saved and signed.
'-----

[ComposePhone]

'Compose a Phone Message in Mail Template

Activity = Compose
RunID = CompPhone
NumSets = 2

Form	= Special\Phone Message	
Save	= Yes	
Sign	= Yes	
SendTo	= Test UserA49	' To field
Field	= PhoneCaller = Someone important	' Contact field
Field	= CompanyName = Important company Inc.	' Company field
Field	= PhoneNumber = (301)555-1212	' Phone# field
Field	= PhoneReason = T	
Field	= Subject = Phone Message from Someone Important	
Text1	= This message will test the special phone message,	
Text2	= which is a form in the mail databases.	

'-----

[Calendar1a]

Activity	= Calendar
RunID	= CalendarA
CalendarType	= Invitation
Description	= Test invitation
Date	= 7/7/01
Time	= 9:00 AM - 10:00 AM
PencilIn	= No
FundFreeTime	= Yes
SendInvitation	= Yes
Text1	= This is the detailed description field of an invitation.
RequiredAttendees	= Test UserA09, Test UserA56, Test UserA99
OptionalAttendees	= Group2B

'-----

[Calendar1b]

Activity	= Calendar
RunID	= CalendarB
CalendarType	= Event
Description	= Lotusphere
Date	= 9/26/00
Duration	= 2 – 6
Text1	= Attend Lotusphere in Germany

'-----

[Calendar1c]

Activity = Calendar
RunID = CalendarC
CalendarType = Anniversary
Description = Test Anniversary
Date = 8/15/01
Text1 = This my anniversary

[MailRead]

Activity = User Defined
RunID = MailRead
Numsets = 3
Command = Open Database
View = \$All
Command = Navigate
NavigateOption = Bottom
Command = Navigate
Up = 6
Command = Open Document
Command = Sleep
SleepTime = 10-30
Command = Close Document
Command = Navigate
Down = 1
Command = Open Document
Command = Sleep
SleepTime = 10-30
Command = Close Document
Command = Sleep
SleepTime = 4
Command = Navigate
Down = 1
Command = Open Document
Command = Sleep
SleepTime = 10-30
Command = Close Document
Command = Navigate

Down = 1
 Command = Open Document
 Command = Sleep
 SleepTime = 10-30
 Command = Close Document
 Command = Navigate
 Down = 1
 Command = Open Document
 Command = Sleep
 SleepTime = 10-30
 Command = Close Document
 Command = Navigate
 Down = 1
 Command = Close Database

[EditDocs1]

Activity = Edit
 RunID = Edit1
 SleepTime = 20
 DBName = discuss.nsf
 View = \$All
 NumberOfDocuments = 3
 Deletedocument= Yes
 CopyDocument= Yes
 SendTo = Test UserA02
 Save = Yes
 Send = Yes
 Sign = Yes
 Field = Subject = Test of Edit Document
 Field = Field1 = Just Testing.
 Field = Field2 = Testing Again.
 Text1 = This document was sent to you from ProActive Load
 Text2 = by G2 Associates. <ENTER 2>
 Text3 = The Edit Activity added this info to the Body field
 Text4 = of the note.
 Attach = ProActiv.ini
 Compress = Yes

Appendix C – Error Messages/Troubleshooting

This appendix contains the error messages for both ProActive Assistant and ProActive Load

ProActive Assistant Error Messages

[Activities] Section not found in INI file - terminating run

Possible Cause: Activities section missing, not spelled correctly, or not recognized as a section.

Action: Check spelling and format of [Activities]. Make sure no other characters are present on the same line.

Cannot attach Result File '<ResultFile>' - Result file not sent

Possible Cause: Unspecified problem attaching the result file.

Action: Send the result file after the session terminates.

Cannot determine Version of Lotus Notes (Cannot locate Create or Tools menu) - terminating run

Possible Cause: Environmental problem starting Lotus Notes.

Action: Attempt to start Lotus Notes on the workstation.

Cannot find Result File '<ResultFile>' Result file not sent

Possible Cause: Unspecified problem in trying to locate the result file.

Action: Locate and send the result file after the session terminates.

Cannot find Memo form - Result file not sent

Possible Cause: Database does not have a form named "Memo."

Action: Send the result file after the session terminates. Use a database that has the standard Lotus Notes Memo form.

Cannot find document with string '<Find String>' - continue with next activity?

Possible Cause: No document exists in the selected view of the database with the string specified in the **FindString** parameter.

Action: Check spelling of the **FindString** parameter. Check for the existence of a document within the appropriate database and view. Press **Yes** to continue with the next activity, **No** to terminate the session.

Cannot find menu '<Menu>' - continue with next activity?

Possible Cause: Value of **Activity** parameter is misspelled.

Action: Check spelling of **Activity** parameter. Press **Yes** to continue with the next activity, **No** to terminate the session.

Cannot find <String> - doc not deleted

Possible Cause: Unknown

Action: Delete the edited document from the database.

Cannot Locate Activity '<activity>' - continue with next activity?

Possible Cause: Value of **Activity** parameter is misspelled.

Action: Check spelling of **Activity** parameter. Press **Yes** to continue with the next activity, **No** to terminate the session.

Cannot Locate <INI filename> - terminating run

Possible Cause: File does not exist in correct directory.

Action: Check spelling and location of the .INI file. The file must exist in the current directory or in a directory in the PATH, DPATH, or G2PATH environment variables.

Cannot Locate Lotus Notes - terminating run

Possible Cause: Lotus Notes executable not accessible to workstation, path to Lotus Notes executable not found or **NotesEXENAME** parameter is misspelled.

Action: Check the **NotesEXENAME** parameter in the .INI file for correct name. If the name is not fully qualified, check the PATH environment variable for a directory that contains the executable.

Cannot open File <Mail DBName> on <Mail Server> Result file not sent

Possible Cause: Lotus Notes database could not be opened. Server possibly down.

Action: Use correct database. Check server operations.

Cannot start Lotus Notes - terminating run

Possible Cause: Environmental problem starting Lotus Notes.

Action: Attempt to start Lotus Notes on the workstation.

Could not find '<view1>' in View Menu"

Possible Cause: View is not visible in the current database's **View** menu

Action: None required. The activity continues using the current view.

Did not find <Find String>

Possible Cause: String does not exist within the current view of the current database.

Action: None required. The activity continues with the next string.

<Lotus Notes error>

Possible Cause: Any error coming from the Lotus Notes application. For example, **<Database not found>**.

Action: Refer to the error message for information.

Lotus Notes - Workspace Cannot be located - terminating run

Possible Cause: Environmental problem starting Lotus Notes.

Action: Attempt to start Lotus Notes on the workstation.

No views were found for database '<database>' - continue with next activity?

Possible Cause: The database does not have any views available in the Lotus Notes **View** menu.

Action: Use a different database, or change the design of the database to include views in the **View** menu. Press **Yes** to continue with the next activity, **No** to terminate the session.

Start time parameter(s) incorrect: - terminating run

Possible Cause: Incorrect format of **StartTime** parameter.

Action: Check format of **StartTime** parameter (hh:mm:ss). Ensure that hh is between 00 and 23, mm and ss are between 00 and 59.

Unable to send document - Dialog box did not appear

Possible Cause: Timing problem in attempting to bring up Lotus Notes **Send** dialog box.

Action: Add a text parameter pause to slow Assistant down.
Text = <Pause 5000>

Unrecognized parameters in INI file: <Error Message> - terminating run

Possible Cause: Parameter is misspelled.

Action: Check the .INI file at the Line # indicated in the error message. Check for correct spelling of the parameter name.

Unrecognized parameter(s) in [Activities] Section: <Error Message> - terminating run

Possible Cause: Misspelled parameter or section name.

Action: Check the .INI file at the Line # indicated in the error message. Check for correct spelling of the parameter name. Check for correct spelling and format of section names (for example, **[TextRead]**). Make sure that no other characters are on the same line as section names.

ProActive Load Error Messages & Troubleshooting

Startup Errors

When ProActive Load starts, it does an initial validation of the test scenario. If errors are found, the user is prompted with a dialogue box indicating that the errors have been logged. If the user clicks **yes**, another dialogue is displayed showing the actual errors that were found, including the line number and section. Load will not run until these errors are corrected.

For ProActive Load to run, the Notes executables must be in the system path of the machine where Load is to be run. When Notes is installed, it does not automatically put the Notes executable in the system path, so this must be done manually. If this is not done, the following error message will appear when you attempt to start the execution of a scenario file.

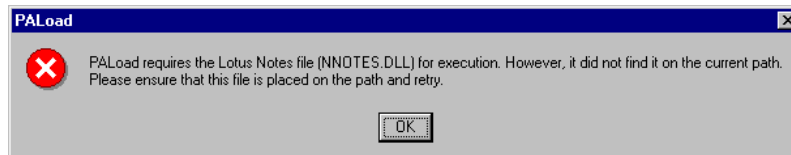


Figure C-1: PALoad Path Error

To modify the system path in NT, select **Settings** and then **Control Panel** from the Start menu button. Then select **Systems** and then the **Environment** tab. Click on the Path variable and then make the changes on the Value line. Press Set, Apply, and then OK. The system path should now be set properly.

Run-Time Errors

Run-time errors encountered while running a scenario are logged to a file. The ErrorFileName parameter assigns the name and location of the error log (see Appendix A, General Section Parameters). The total number of errors is displayed on the Main Tab, see *Figure C-2* below.

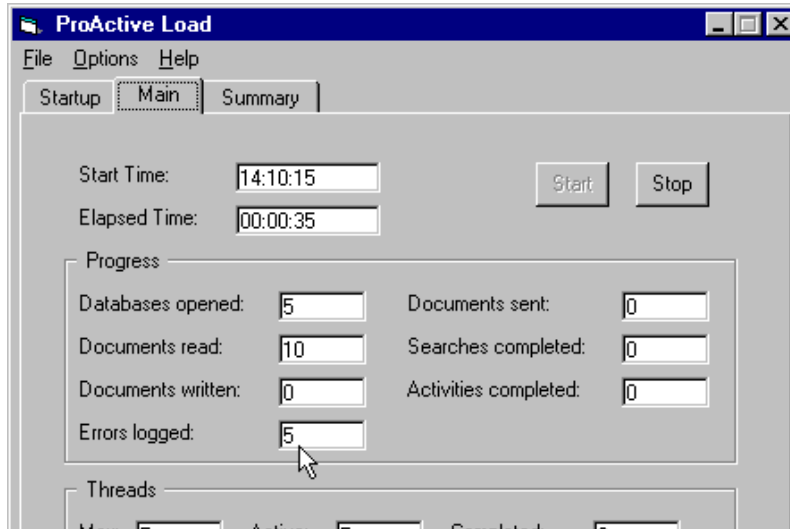


Figure C-2: Errors Logged During Run

You can click the Stop button at any time to end the scenario. You can also view the errors while the scenario is running by going to the Error Log and opening it with a text editor.

If you do not stop the execution of the scenario and it has incurred errors, you will be able to view them by either opening the Error Log with a text editor, or pressing the View Errors button on the Summary Panel as shown in *figure C-3* below.

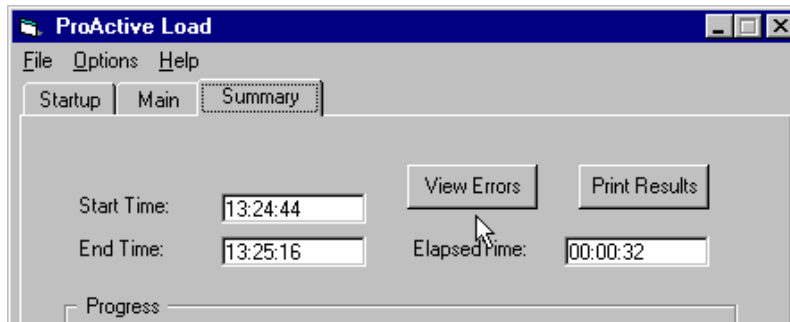


Figure C-3: Summary Panel

When processing has completed and you are asked if you want to view the result, and you select “NO” and have incurred errors, you will still be prompted to view the errors as shown in *figure C-4* below.

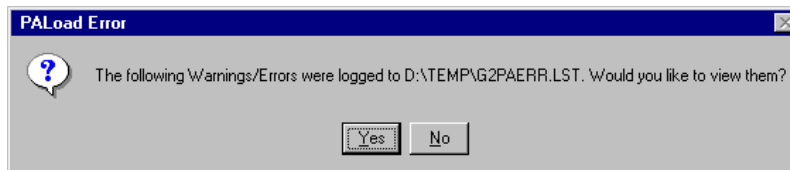


Figure C-4: PLoad Error

When you press “Yes” or “View Errors” the error log will be displayed in the dialogue box shown in *figure C-5* below.

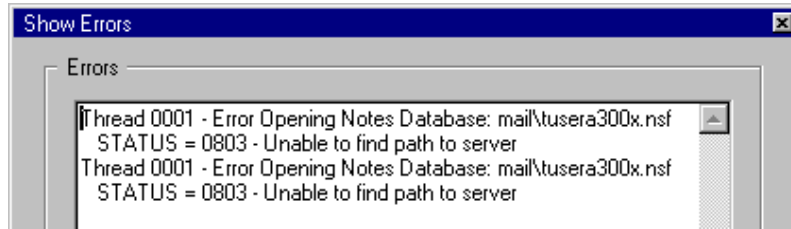


Figure C-5: Show Errors

Use of NT Remote Server Counters

The Load front-end may hang when running more than one Load workstation with the remote NT server counters turned on and the screen refresh rate set to less than 10 seconds. When the front-end hangs, the backend continues to run properly (you do lose the TimeLine data since it stops when the UI hangs). The problem does not appear when the screen refresh rate is left at the default value of 10 seconds. The problem can be avoided altogether if only one workstation is configured to capture the remote NT server parameters.

Too Many Client Sessions

When no view is specified in the Read Activity, the number of user sessions opened on the server is double the number of threads specified in the Load scenario. This problem can be avoided by always specifying the view to be used in the Read Activity.

Appendix D – File Layouts

Table of Limits

The number of parameters in an INI file allows is only restricted by the amount of available memory on the workstation.

Number of lines in .INI file:	Unlimited
Number of Sections:	Unlimited
Number of FindString parameters (Find Activity):	Unlimited
Number of SendTo parameters (Mail Activity):	Unlimited
Number of Form parameters (Compose Activity):	Unlimited
Number of Lotus Notes views:	Unlimited
Number of nested Do groups:	Unlimited
Number of threads per workstation (ProActive Load):	Limited by the workstation

Result File Layout

ProActive Tools writes one record for each "event" within a ProActive Load Simulator session. Each record contains the following fields:

Run ID	Run ID of the test session
Date	Date of the event
Time	Time of the event
Activity	Activity (e.g., Read Documents)
Event	Event (e.g., Open Database)
Client	Client
Server	Server
Iteration	Iteration number
Response time	Time to complete the event in seconds (e.g., 11.234)
Completion Code	Indicates success or failure of the event 0 - OK 1 - Server Not Responding 2 - Lotus Notes Error Message 3 - Unknown Lotus Notes Message
Comments	Additional information about the event

Timeline File Layout

G2's ProActive Load writes one record for each minute run within a ProActive Load Simulator session. The server for remote server values is defined in the General Section by the NTServerName parameter as shown below.

NTServerName = G2Server01

The TimeLine file is saved to the location and filename specified in the TimeLineFile parameter as shown below.

TimeLineFile =C:\2besaved\AnyFileName.csv

Timeline Structure	
Run ID	Run ID of the ProActive Load session
Date	Date of the event
Time	Time of the event
DBs Opened	Databases opened
Docs Read	Documents read
Docs Written	Documents written
Docs Sent	Documents Sent
Searches	Searches completed
Activities	Activities completed
Errors	Errors recorded during test run
CPU%	Percent CPU utilization

Timeline Structure (continued):	
Memory Commit	Memory committed
Pg/Sec	Pages per second
Remote CPU%	Remote server percent CPU utilization
Remote Memory Commit	Remote server memory committed
Remote Pg/Sec	Remote server pages per second

Appendix G – ProActive Tools

Glossary

Activity	Each Load or Assistant simulator session executes activities according to the scenario specified in the scenario file for that workstation. An individual activity is comprised of a series of events that must occur to complete one iteration of the activity. Examples of activities would be mailing, editing, reading a document, etc. Activities are also referred to as Activity Routines. The different types of activities available are described in Chapters 2 & 4.
Activity Set	A collection of Activities that are called to be executed in a specific sequence. ProActive Load allows for multiple activity sets in the PROACTIV.INI file. Activity sets consist of called Activities, SleepTimes between called Activities and loops. ProActive Assistant includes only one default Activity Set named Activities. Available activities types are described in Chapters 2 & 4.
CSV File	Comma Separated Variable file. The results recorded by ProActive Load and ProActive Assistant are stored in a file with the extension CSV, for comma separated variable format. The default results file is named PROACTIV.CSV, but any file name may be used.
Event	The smallest unit of work measured by the Simulator. Each activity is made up of multiple events. Each event is logged as a single record in the results file. Events are described in Chapter 3.
Iterations	The number of repetitions in an activity.
Monitor	The ability of the Assistant simulator to measure response times of events and send alerts if response times exceed defined thresholds.

ReadTime	The amount of time the Simulator waits after opening a document to simulate a user reading the document.
Report Wizard	The part of the Reporter that allows custom designed reports to be created.
Reporter	This component of ProActive Tools analyzes the results files produced by the individual ProActive Assistant runs. The Reporter runs on a Windows 95/98 or Windows NT workstation and generates standard and custom reports. The Reporter can also export files to other applications.
RunID	The RunId is a user-assigned value to identify a specific simulation run or activity, and is stamped on each record in the results.
Session	A single execution of the ProActive Load or ProActive Assistant on a single workstation or series of workstations.
Scenario	A series of activities that make up a simulation test for a single workstation.
Scenario File	Users define and run simulations through the ProActive parameter file (by default, PROACTIV.INI). This file is a simple ASCII file that controls the execution of G2's ProActive Assistant and Load. The Scenario file allows Notes performance engineers to tailor the execution of the ProActive Tools for specific environment and testing requirements.
Scenario Wizards	The Lotus Notes databases that act as a front end user interface to help create Scenario scripts that run Simulator tests and is used primarily as a learning tool.

SleepTime	The amount of time the Simulator waits between activities or iterations.
Test	The combination of sessions run by one or more workstations at the same time.
Test Scenario	A collection of scenario files that have been developed to solve a specific testing problem. These may have been designed to run ProActive Assistant and ProActive Load on several workstations.
Threads	One individual user simulated by ProActive Load. The maximum number of threads run by ProActive Load is limited by the power of the workstation, the complexity of the scenario, and the ability of the network to carry the generated load. A strong workstation with dual processors and lots of memory will support more threads than a light weight machine. When using a strong workstation, be sure that you do not overwhelm your network.

Index ([click page numbers to link](#))

@

[@Variable Function, 58](#)

A

Activities, [85, 191](#)

Calendar, [121](#)

Compose, [54, 124](#)

Copy Database, [127](#)

definition, [193](#)

Edit, [129](#)

Find, [133](#)

Index, [135](#)

Mail, [54, 137, 144](#)

Read Documents, [140](#)

Rebuild View, [142](#)

User Defined, [146](#)

Views, [155](#)

Activity Routines Section, [169](#)

description, [19, 54](#)

example, [54](#)

structure, [56](#)

Activity Set

definition, [193](#)

Activity Sets Section

description, [18, 49, 118](#)

Do Groups, [52](#)

example, [50](#)

threads, [49](#)

Activity Types

predefined, [55](#)

Alerts

Monitor Section, [115](#)

Assigning

test names, [82](#)

C

Calendar, [121](#)

example, [123](#)

Calendar Activity, [121](#)

Call=, [51](#)

Comments

entering into Scenario Files, [59](#)

Compose, [124](#)

Compose Activity, [124](#)

example, [126](#)

Copy Database, [127](#)

Copy Database Activity, [127](#)

example, [128](#)

CSV Files. *See* Output Files

definition, [193](#)

Custom Reports

developing, [94](#)

executing, [99](#)

saving, [98](#)

selecting report format, [97](#)

selecting values, [95](#)

D

DBName

threads, [47](#)

Debugging the Test Scenario, [68](#)

DefaultActivity, [49](#)

Delayed Start, [45](#)

Deleting Test Files from

Reporter Database, [100](#)

Do Groups

description, [52](#)

example, [52](#)

Documentation, [25](#)

E

Edit, [107](#), [129](#), [151](#)
Edit Activity, [129](#)
 example, [132](#)
Error Messages
 ProActive Assistant, [181](#)
 ProActive Load, [185](#)
ErrorFileName, [44](#)
Events, [38](#)
 definition, [193](#)
 description, [19](#), [56](#)

F

File Layouts, [189](#)
 Result File, [190](#)
 Table of Limits, [189](#)
 Timeline File, [191](#)
Find, [133](#)
Find Activity, [133](#)
 example, [134](#)
FindFreeTime, [122](#)
FirstThread, [48](#)

G

General Section, [113](#)
 description, [18](#), [43](#)
 example, [44](#), [114](#), [120](#)
 parameters, [110](#), [118](#)
 threads
 Activity Sets, [49](#)
 DBName, [47](#)
Generalized Thread Substitution, [60](#)
Glossary, [193](#)

H

Help - Documentation and Sample
 Files, [25](#)

I

Index, [135](#)
Index Activity, [135](#)
 example, [136](#)
Installation
 ProActive Tools, [24](#)
 Scenario Wizards, [26](#)
Invoke Method Parameters, [164](#)
Iterations
 definition, [193](#)

L

ListNoteId, [44](#)
ListNoteID, [110](#)
Lotus Notes/Domino testing
 considerations, [16](#)

M

Mail, [111](#), [137](#), [144](#)
Mail Activity, [137](#), [144](#)
 example, [139](#)
MaxStartWait, [45](#)
Monitor Section, [115](#)
 definition, [193](#)
 example, [116](#)

N

New Features in Release 5.2, [14](#)
Notes Activities, [19](#)
NT Remote Server Counter, [188](#)
NumSets, [44](#), [47](#), [111](#), [112](#), [168](#)

O

Out of the Box Scenarios, [29](#), [31](#)
 ProActive Assistant, [28](#)
 ProActive Load, [30](#)
 ProActive Reporter, [35](#)
 Running, [27](#)

- Output Files
 - assign path/name, [112](#)
 - Result File Layout, [190](#)
 - Timeline File Layout, [191](#)

P

- Predefined Activity Types
 - listing of, [55](#)
- ProActive Assistant
 - “Out of the Box” Scenario, [28](#)
 - description, [11](#)
 - error messages, [181](#)
 - Sample Scenario, [167](#)
 - system requirements, [21](#)
 - text parameters, [158](#)
- ProActive Load
 - “Out of the Box” Scenario, [30](#)
 - DBNames, [47](#)
 - description, [12](#)
 - error messages, [185](#)
 - generalized Thread Substitution, [60](#)
 - Sample Scenario, [174](#)
 - system requirements, [22](#)
 - text parameters, [156](#)
 - threads, [47](#)
- ProActive Reporter, [77](#), [87](#)
 - ”Out of the Box” Scenario, [35](#)
 - assigning Test Names, [82](#)
 - definition, [194](#)
 - deleting tests, [100](#)
 - description, [13](#)
 - importing result files, [80](#)
 - overview, [78](#)
 - Report Layout, [86](#)
 - Report Selection, [85](#)
 - Report Wizard, [94](#)
 - selecting test results, [83](#)
 - system requirements, [22](#)
- ProActive Tools
 - description, [10](#)
 - installation, [21](#)

- installation Options, [24](#)
- overview, [10](#)
- understanding, [62](#)
- uses for, [17](#)

- ProActive Tools.pdf, [25](#)
- ProActiveTools.com, [7](#)
- Purpose of the “Out of the Box” Scenarios, [27](#)

R

- Random, [44](#), [112](#)
- Read Documents, [140](#)
- Read Documents Activity, [140](#)
 - example, [141](#)
- ReadTime
 - definition, [194](#)
- Rebuild View, [142](#)
- Rebuild View Activity, [142](#)
 - example, [143](#)
- Report Wizard, [94](#), [95](#)
 - definition, [194](#)
- Reporter. *See* ProActive Reporter
- Reset Result Activity
 - example, [145](#)
- Result File, [20](#), [73](#), [112](#)
 - importing, [80](#)
 - layout, [190](#)
- RunId
 - definition, [194](#)
- Running the Test Scenario, [71](#)
- RunType=
 - Count, [47](#)
 - Timed, [47](#)

S

- Sample Scenarios, [167](#)
- Sample.nsf, [25](#)
- Scenario
 - definition, [194](#)
 - test structure, [18](#)

- Scenario File
 - definition, [194](#)
 - entering comments, [59](#)
 - exporting, [108](#)
 - overview, [42](#)
 - parameters, [109](#)
 - structure, [41](#)
- Scenario Wizards, [103](#)
 - building a scenario file, [105](#)
 - building an activity routine, [107](#)
 - definition, [194](#)
 - description, [20](#)
 - installation, [26](#)
 - overview, [104](#)
 - using, [68](#)
- SendTo Options, [57](#)
- Session
 - definition, [194](#)
- SleepTime, [44](#), [112](#), [173](#)
 - definition, [195](#)
 - description, [51](#)
- SleepUntil, [51](#), [119](#), [173](#)
 - description, [51](#)
 - example, [51](#)
- Special Text Tags, [157](#), [159](#)
 - ProActive Assistant
 - Action, [159](#)
 - Attach, [160](#)
 - Date, [160](#)
 - EndEvent, [160](#)
 - Iteration, [160](#)
 - Menu, [160](#)
 - Pause, [161](#)
 - StartEvent, [161](#)
 - Time, [161](#)
 - ProActive Load
 - Date, [157](#)
 - Iteration, [157](#)
 - Lookup, [157](#)
 - Range, [157](#)
 - Thread, [157](#)

- Time, [157](#)
- StartTime, [45](#)
- Structure
 - Activity Routine, [56](#)
 - Scenario Files, [41](#)
 - Test Scenario, [18](#)
- System Requirements
 - ProActive Assistant, [21](#)
 - ProActive Load, [22](#)
 - ProActive Reporter, [22](#)

T

- Test
 - definition, [195](#)
- Test Duration, [71](#)
 - parameter, [47](#)
- Test Mode, [113](#)
 - ProActive Assistant, [68](#)
 - ProActive Load, [69](#)
- Test Profile
 - development, [63](#)
 - sample form, [64](#)
- Test Scenario
 - building & running, [61](#)
 - debugging, [68](#)
 - definition, [195](#)
 - running the, [71](#)
- Testing
 - General Information, [15](#)
 - Lotus Notes/Domino considerations, [16](#)
- Text Parameters, [156](#)
- Text Tags - see special text tags, [157](#)
- Threads, [44](#), [113](#)
 - assigning to
 - Activity Sets, [49](#)
 - DBName, [47](#)
 - definition, [195](#)
- Timed Versus Count, [47](#)
- TimeLine File, [20](#), [73](#)
 - layout, [191](#)

Tips and Techniques, [74](#)
Troubleshooting
 ProActive Assistant, [181](#)
 ProActive Load, [185](#)

U

User Defined, [146](#)
 commands, [148](#)
 field parameters, [153](#)
User Defined Activity, [146](#)
 example, [154](#)

V

Views, [155](#)
Views Activity, [155](#)

W

Workstation Overload, [72](#)
www.ProActiveTools.com, [7](#)

Copyright

This is the ProActive Tools User Manual documentation for Release 5.2 of G2's ProActive Tools.
Copyright 1996 - 2000

G2 Associates, Inc.
6723 Whittier Avenue
McLean, VA 22101
(703) 288-2944
(800) 884-5876

November 1, 2000

All rights reserved. First edition printed 1996. Printed in the United States.

Lotus and Lotus Notes are registered trademarks and Notes and Domino are trademarks of Lotus Development Corp.

ProActive Assistant, ProActive Load, ProActive Reporter and ProActive Tools are trademarks of G2 Associates, Inc.

Any duplication or distribution without prior written consent from G2 Associates, Inc. is strictly prohibited.