

DynTek ProductivIT?

Agent Reference Guide



Pro.duc.tiv.IT

prō, dək'tiv əit.ē: the ability or capacity to produce. a: abundance or richness in output. b: the physical output per unit of productive effort.

c: the effectiveness in utilizing information technology.

Powerful...

... having the resolution to a support call before the user picks up the phone.

Timely...

... reducing the time and money spent on support by up to 80% while increasing the quality of service.

Information...

... knowing the precise details of the configuration, location, and status of every piece of supported hardware and software.



Notice

Copyright? 1999 - 2004 DynTek – All rights reserved

Product Version: 3.30.20.13

Document Date: January 19, 2003

The content of this documentation is furnished for informational use only and is subject to change without notice, and should not be construed as a commitment by DynTek. DynTek assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.

CONTENTS

NOTICE	2
INTRODUCTION	4
TECHNICAL OVERVIEW	5
BUGSOLVER.EXE	6
<i>Policy Handling</i>	7
CRASHWIZARD.EXE	8
CODEWATCH.EXE	8
<i>Data Collectors</i>	8
WEBCONSOLE.EXE	10
BSPPOOLMAN.EXE	11
DEPLOYMENT, INSTALLATION AND CONFIGURATION	13
DEPLOYMENT	13
INSTALLATION	13
<i>Windows Installer</i>	13
<i>System Requirements</i>	14
<i>Installation Process</i>	14
PERIODIC UPDATES	17
CONFIGURATION	17
<i>Environment Variables</i>	17
<i>Configuration Files</i>	19
<i>Registry Items</i>	19
ACTIVATING THE AGENT	22
AGENT TASKBAR	22
COMMAND LINE INTERFACE	23
START – PROGRAMS MENU	24
CRASH MODE	25

Introduction

ProductivIT, currently running on over 50,000 desktops nationwide, is an integrated desktop management solution that provides enhanced help desk support capabilities, detailed asset tracking, suspicious activity reporting, migration analysis and unlimited data mining opportunities. When activated, it instantly reports over 30,000 fields of system-specific data that is accessible through a secure, centralized, web-based portal. The result is substantial cost savings and tangible improvements in every measure of IT performance and service.

The new release, ProductivIT 5.0, is enhanced with new security controls and customer-requested features that revolutionize the way in which PC users receive technical support and capture asset inventory information. Some of the highlights of the new release include:

- ? Improved Customization, Flexibility and Statistics on Support Home Page
- ? Role-based Security Features
- ? Advanced Escalation and Assignment Rules
- ? Web-based Entry of Incidents
- ? Knowledge Base/Self-Service

Professional support groups estimate that 80 percent of the total support time, and the majority of the expense, is spent determining what hardware and software was running on a computer at the time of a request for support or a system failure. ProductivIT's intelligent agent software reduces that time to virtually zero. The lightweight agent (under 4 MB) sits in the background on individual PCs and notebook computers and remains dormant until activated. When a user requests support or the computer registers a software or hardware failure, it instantly takes a snapshot of all the operating parameters of the computer at that time. ProductivIT captures event logs, device information, system information, process information, registry information and DLL/EXE version information. The information is then sent to a central support database where it can be accessed by support professionals in order to determine what the failure was and how to fix it. In addition, manufacturers and software publishers can utilize this information to find out how to optimize their configurations to reduce future problems.

Technical Overview

The ProductivIT agent is an application that is installed on a client machine as a part of the ProductivIT solution. The agent is responsible for collecting diagnostics data from the client machine, displaying a graphical display, and transmitting the information back to a central repository. The agent is invoked either through user activation, command line interface, or through an application crash (GPF or Blue Screen of Death) as a just in time debugger. The agent was tested on Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, and Windows XP. Many facets of the agent can be customized including the look and feel of the screens as well as customizing what actually gets collected and when it gets collected. The agent is actually made up of several components that interact with each other. As can be seen from figure 1 – Bugsolver.exe launches Crashwizard.exe and Codewatch.exe and Codewatch.exe launches Websend.exe.

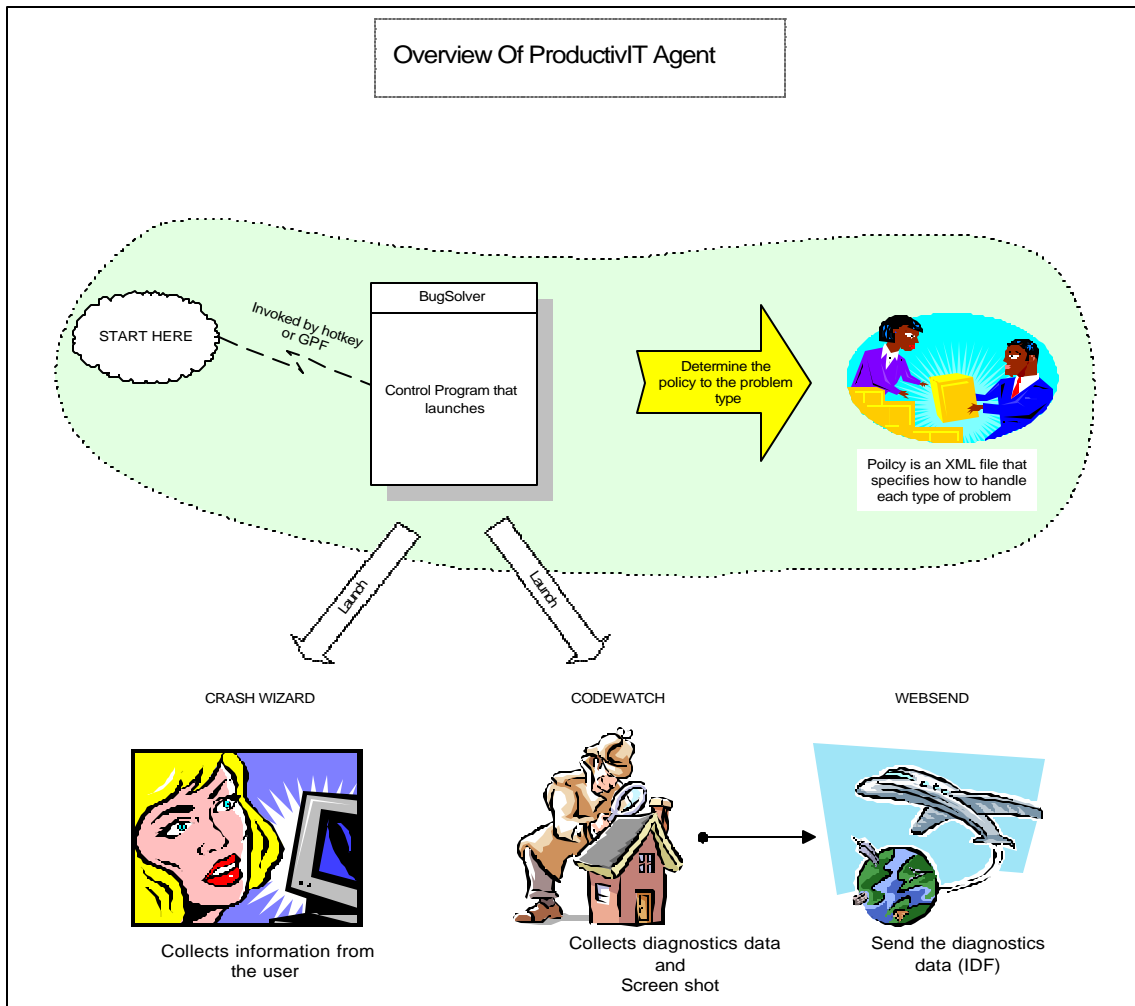


Figure 1

BugSolver.exe

Bugsolver.exe is the controlling application of ProductivIT. The main purpose of BugSolver.exe is to setup the user interface with correct parameters and to determine the level of data collection. Bugsolver.exe will use the information in the policy file to control the operation of the components involved in gathering incident data. BugSolver.exe uses a policy component to determine its subsequent actions.

Once the policy has been determined, BugSolver.exe will launch additional applications, CrashWizard.exe (User Interface) and CodeWatch.exe (Data Collection).

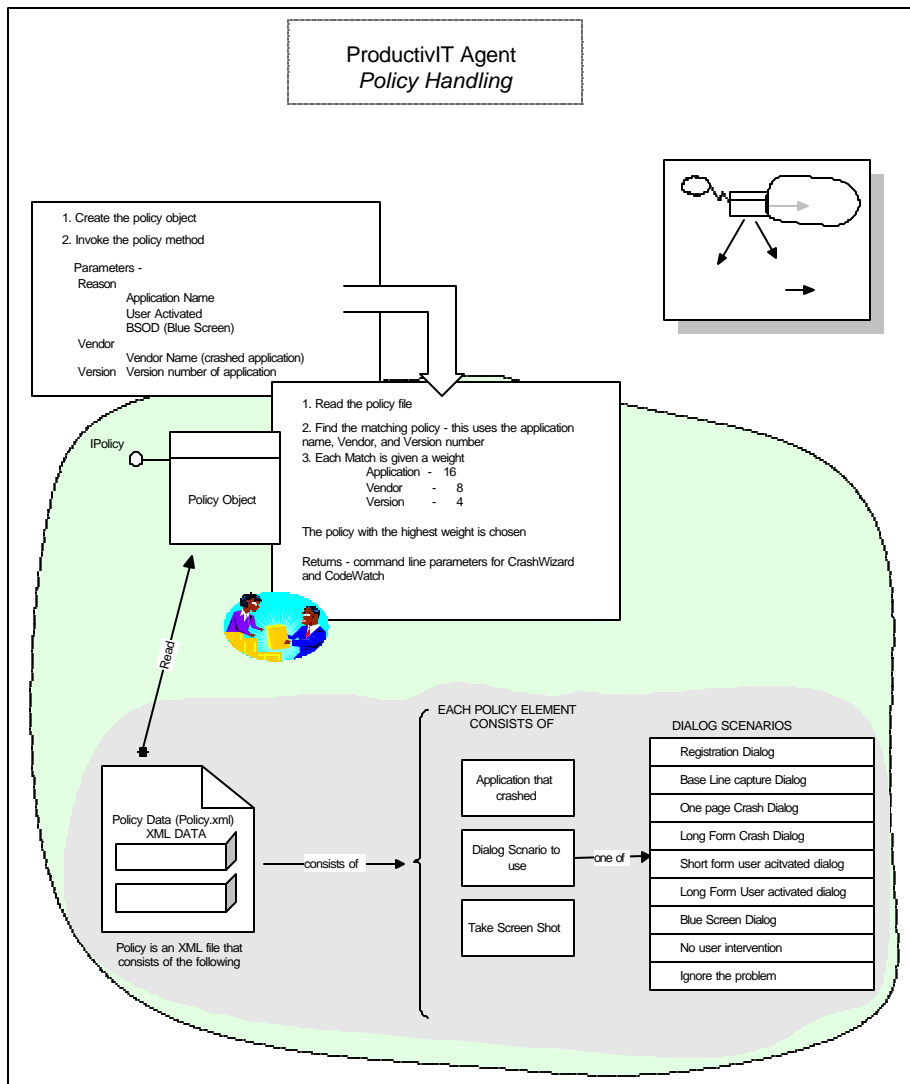


Figure 2

Policy Handling

When BugSolver.exe is invoked it is passed a reason code. This reason code may be one of the following: a crashed application, user activated, registration baseline, command line, or blue screen. This activation code is used to search the policy file to determine subsequent actions.

BugSolver.exe uses an ActiveX component to determine the policy. This component will read the policy file, an XML document, and determine the policy from the reason code. The policy component returns command line parameters that will be used to activate both CrashWizard.exe and CodeWatch.exe. With the command lines in hand, BugSolver launches CrashWizard.exe will know what dialog scenarios to use and CodeWatch.exe will know what to collect. This is pictured in figure 2. You can set policy by Application name, Vendor name, version number, or a user defined option.

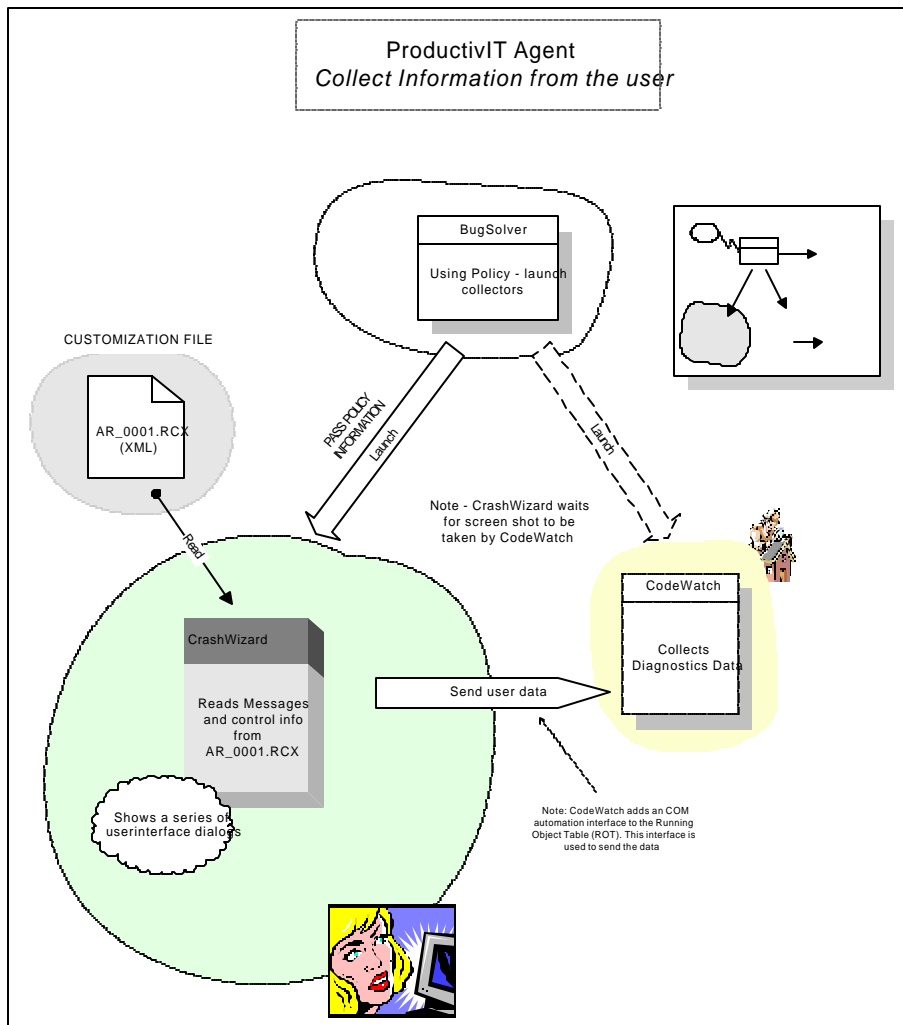


Figure 3

CrashWizard.exe

The crashWizard.exe program is used to display a series of dialog screens to collect some basic information from the user as to the nature of the problem. Through parameters passed from Bugsolver.exe, CrashWizard.exe will determine what user interface scenario to display. In addition, Crashwizard.exe will read data from an XML file to determine what fields to display, the text of the dialogs, what icon to use, and all data related to drop down list boxes. The scenarios that have been built into CrashWizard.exe are as follows to:

1. Registration dialog
2. Baseline capture dialog.
3. One-page crash dialog
4. Long form crash dialog
5. Short form user activated dialog
6. Blue screen dialog

Once CrashWizard.exe has gathered all user input, it sends the data to CodeWatch.exe through an OLE automation interface.

When finished collecting data, CrashWizard.exe will let Bugsolver.exe know that it is finished. Bugsolver.exe will wait for CodeWatch.exe and merge the data together and send to the ProductivIT database for analysis. This is pictured in figure 3.

CodeWatch.exe

This is the central component of the ProductivIT agent. It gathers diagnostics data from the machine and transports into a compressed XML file ready to send to the ProductivIT database. Like the rest of the agent, this component is highly configurable. CodeWatch.exe itself does not directly gather the data but uses a set of ActiveX components called collectors to get the data. The ActiveX components are themselves highly configurable and their configuration information is kept in an XML file. CodeWatch.exe can be configured to collect different data depending on the scenario and policy determined by the policy component. It is beyond the scope of this document to describe the collection architecture in detail, but the following is a list of the types of data collected.

The agent gathers up to 30,000 data points. The types of information included in the collection are:

Data Collectors

- 1) Device Information – All devices attached to a machine will be recorded. This includes device name and type for devices such as physical disk drives, local and network printers, multimedia devices, etc.
- 2) Dll/Exe Information – This collector get all information for all EXE's and all DLL's currently running on a desktop. This includes EXE name, vendor, ID, CPU utilization, memory utilization, associated DLLS and many more details.
- 3) Application and System Event Log data

- 4) Process information – The process information is used to debug applications that have crashed. The collector will retrieve all state information related to the machine, all running processes, associated dll's and exe's, information related to all threads running for a process, all windows related to process with state information, register values, and much more.
- 5) Registry information – The registry collector will collect as much or as little of the registry as you would like.
- 6) System information – System information includes – Computer name, Directory information, Network information, Windows information, Drive Information, Network Protocols, environment variables, device drivers, services and status, IRQ information, complete SMBIOS information, and Direct X information.
- 7) File collection – This collector will collect the entire contents of text files and store the results in the XML file. The default collection is for – System.ini, Win.ini, control.ini, config.sys, autoexec.bat, and ProductivIT log file.
- 8) Event Log – This collector will retrieve all information related to the system and application event logs.
- 9) Customized collectors – Customized collectors can be created or can be called within ProductivIT to include data in the XML file.

The data that is collected is stored in an XML file, with the .RCC extension, on the local drive. CodeWatch.exe will also take a screen shot to capture the state of the display when the problem occurred. In order to speed the transport of the data, the XML file is compressed and the screen shot is stored as a highly compressed JPEG file. Codewatch.exe operation is pictured in figure 4.

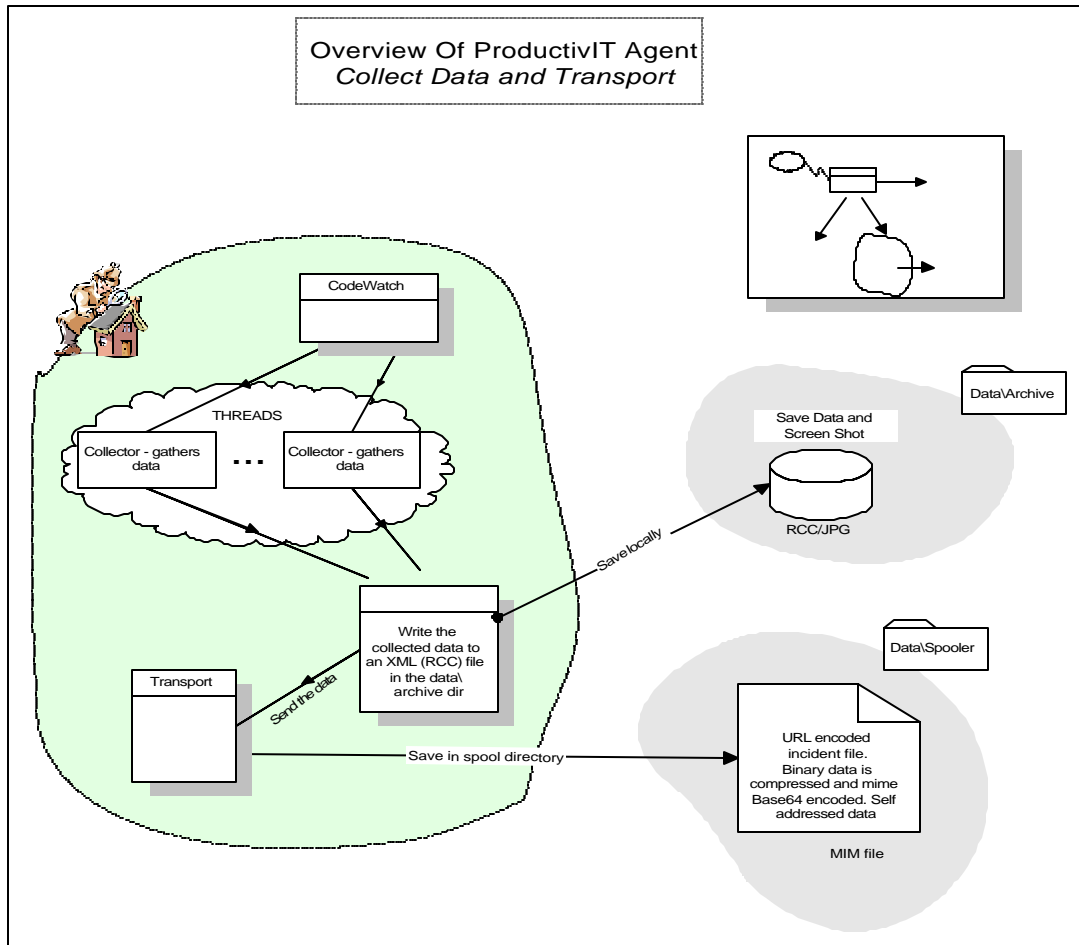


Figure 4.

Websend.exe

WebSend.exe is the program used to transport the Incident Data File (IDF) to a local server or to the ProductivIT database. The data will be sent over the Internet using the transport component. The data is sent using HTTP thus avoiding any problems negotiating a firewall. This too is an ActiveX component that is part of the collected configuration. If an Internet connection is not available at the time of the incident, the data is placed in the spool directory to be sent later when a connection has been established. All this is completely transparent to the user and show in figure 5.

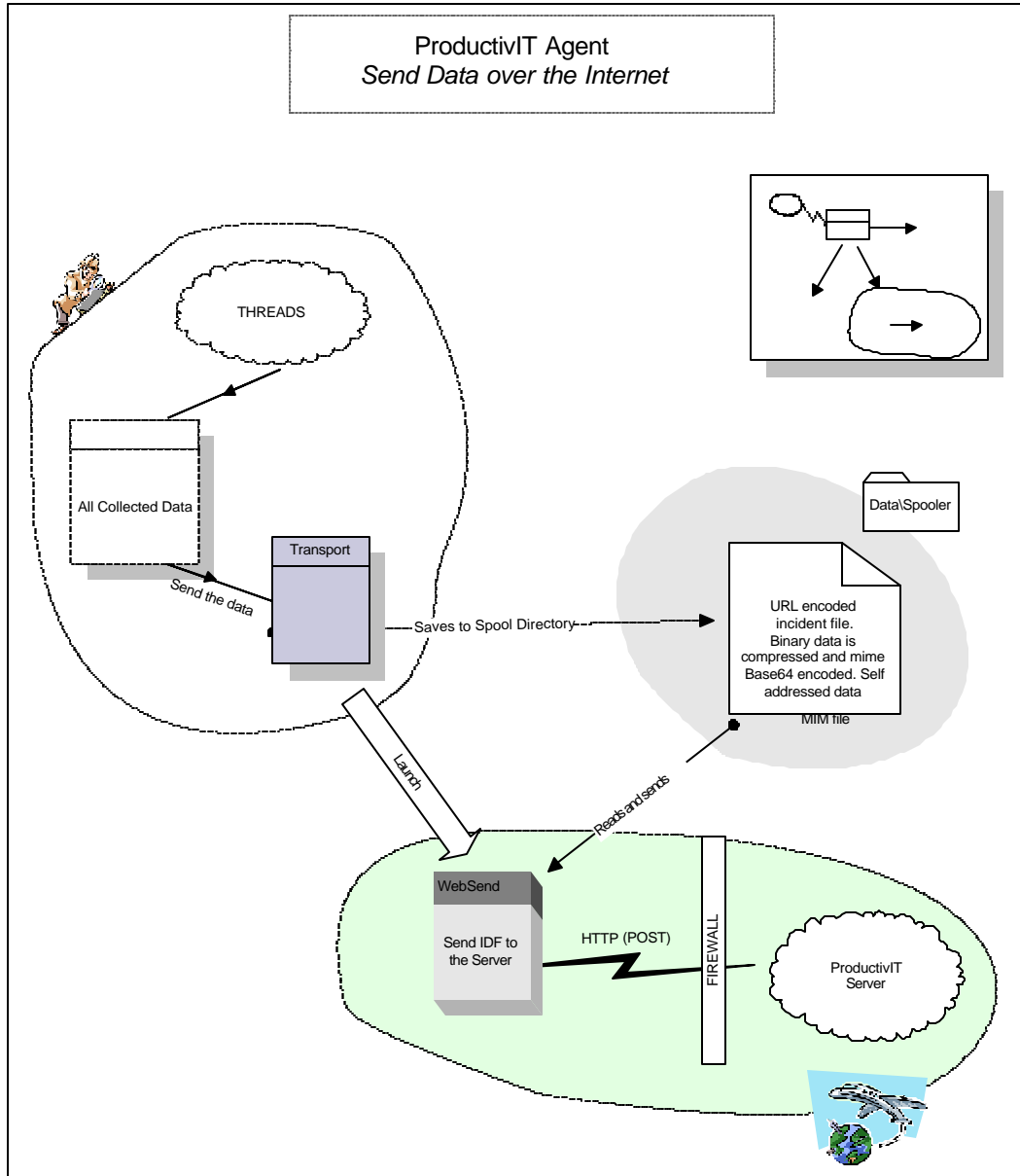


Figure 5

BSPoolman.exe

BSPoolman.exe is a spooler program that wakes up every x minutes to see if any files are waiting to send. The spooler is used to send files that could not be sent earlier because of any Internet connection.

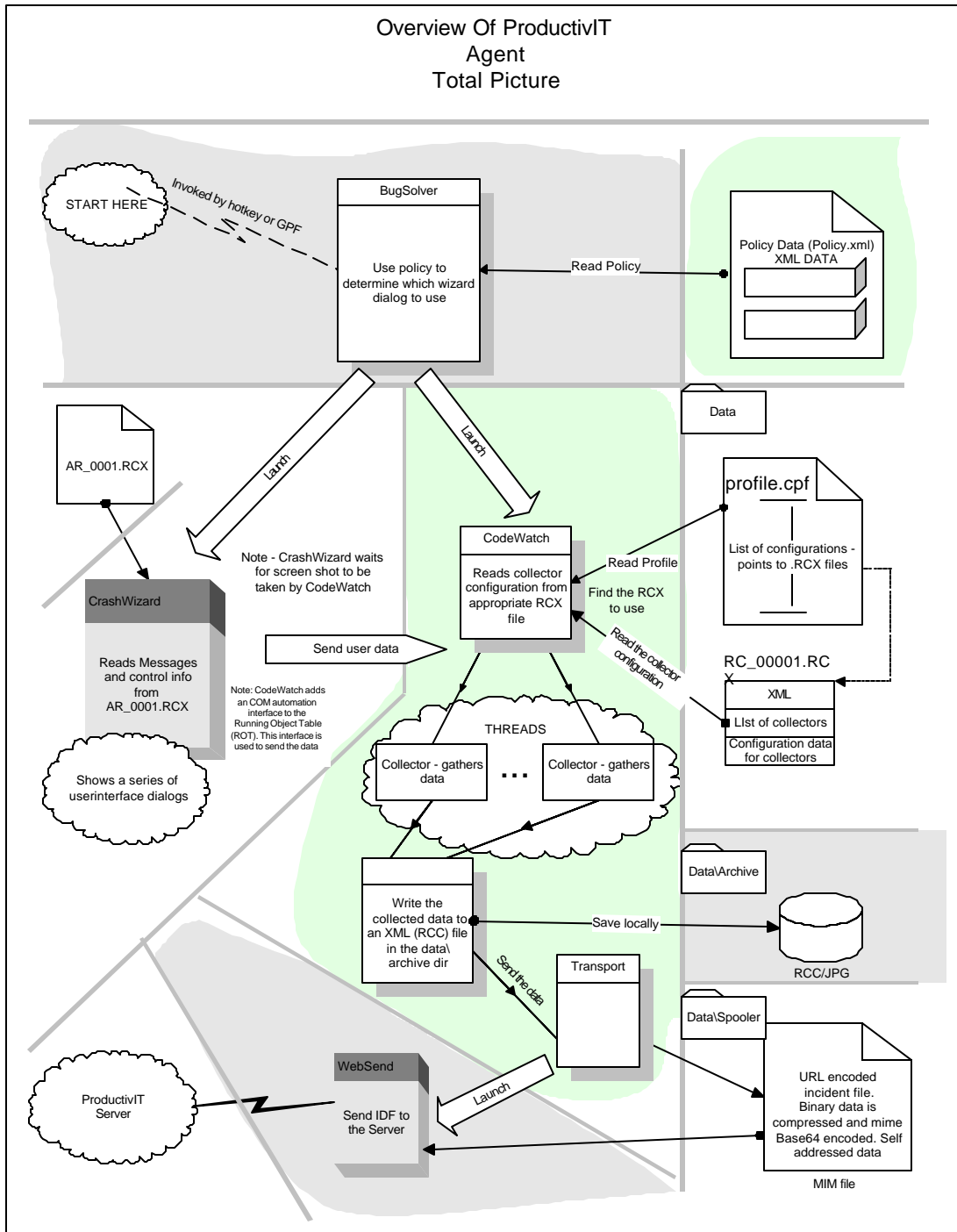


Figure 10

Deployment, Installation and Configuration

Deployment

The ProductivIT agent can be deployed using many different methods because of its small size and Web portal backend. The deployment method you select can be customized to your needs as an organization. The methods currently available include:

- A. Directly download agent through ProductivIT Web Portal or through a web link from any web page.
- B. Through a network Logon script
- C. Through and E-mail web link or attachment.
- D. Distributed with a CD.
- E. Directly download agent from a shared network drive.
- F. Through electronic software distribution packages, like Microsoft SMS, Computer Associate's ESD and Novell's Zen Works.

Installation

Regardless of the method of deployment the installation instructions will be the same for each method. As mentioned previously in this document, the agent is highly customizable and as a result the print screens shown below may actually differ in your installation. Where customization is possible it is noted.

Windows Installer

The Agent installation is performed by Microsoft's "Windows Installer" (**MSI**), this is the Microsoft-supported installation method for software that runs in the Microsoft Windows environment. The installation script itself is stored as an "MSI" file, which contains all of the product components and the "database" that Windows Installer uses to sequence through product installation, repair and un-installation.

For the purpose of overview, an MSI file is a file that contains a database of tables with fields. The Windows Installer reads the database and acts according to certain values in the database. In one sense, the database is akin to a machine-interpreted language, with the Windows Installer as the interpreter – an MSI file is a relatively primitive set of instructions and data.

Although the Windows Installer is a standard component of Windows XP, Windows 2000 and Windows Me, it is not standard with Windows NT, Windows 98 and Windows 95. Therefore, the agent installation process is prepared to install the Windows Installer itself on systems that do not already have it (or have an inferior version). In addition to the MSI installation file, the agent is available in two EXE formats that install the Windows Installer before installing the Agent. The two EXE formats are for Windows 9x and Windows NT, which have slightly different versions of the Microsoft Windows Installer.

System Requirements

- ✎ Windows 95/98/NT/2000/XP
- ✎ 16 Megabytes of RAM (32 recommended)
- ✎ 20 Megabytes of hard disk space
- ✎ Internet access with an e-mail account

Additional Requirements if downloading from the Web Portal

- ✎ Microsoft Internet Explorer 5.0 or later *OR* Netscape Navigator 6.0 or later
- ✎ Cookies must be enabled on your browser.

*****Note: On secure systems you need to have proper access to install software in order to complete installation.**

Installation Process

The installation is a straightforward and simple process that actually takes less than 2 minutes to complete. Below are the screens that appear while installing

STEP1. If you already have the agent installed then you will need to un-install the current version before the installation of the new agent can start. Figure 11 shows the first dialog of the Installation Wizard. The current version number is listed at the bottom of the screen. You can press the cancel button at anytime during these dialog boxes to cancel the installation. To continue to step 2 then press the next button.

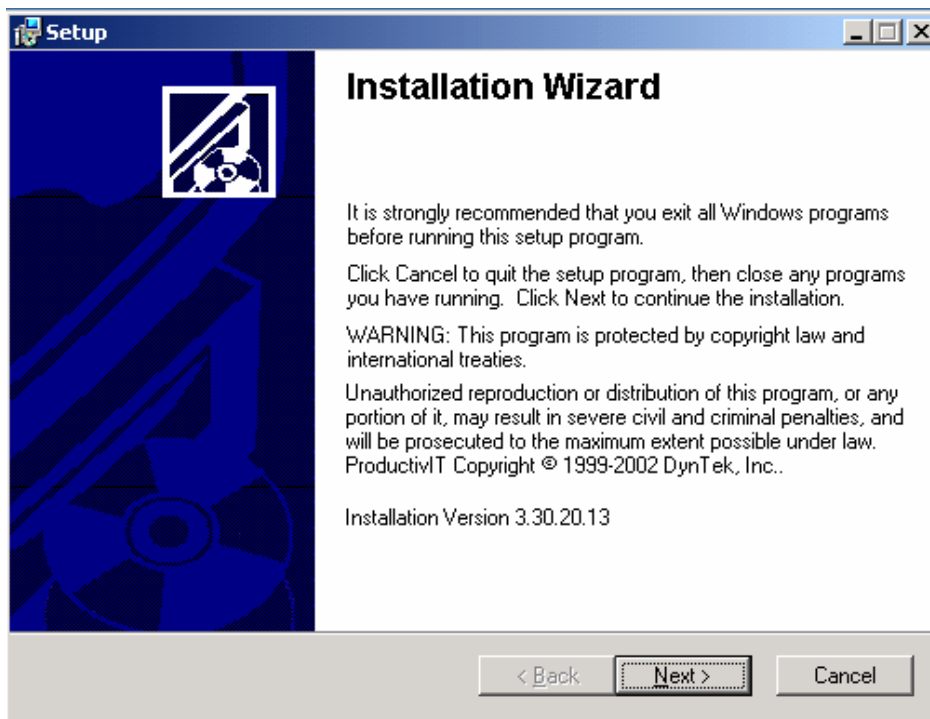


Figure 11

STEP2. The installation wizard will ask for the destination folder to store the application. The default directory will be the Program files\Dyntek\ProductivIT directory. It is possible that your installation was customized and this default directory was changed to a value other than the one noted. In either event, if you wish to change the destination folder at run time it will be done here. Select the Browse button and you will have an explorer dialog box to search through. Pressing the Back button will take you back to Step 1, pressing the Cancel button will cancel the installation, and pressing the Next button will continue the installation.

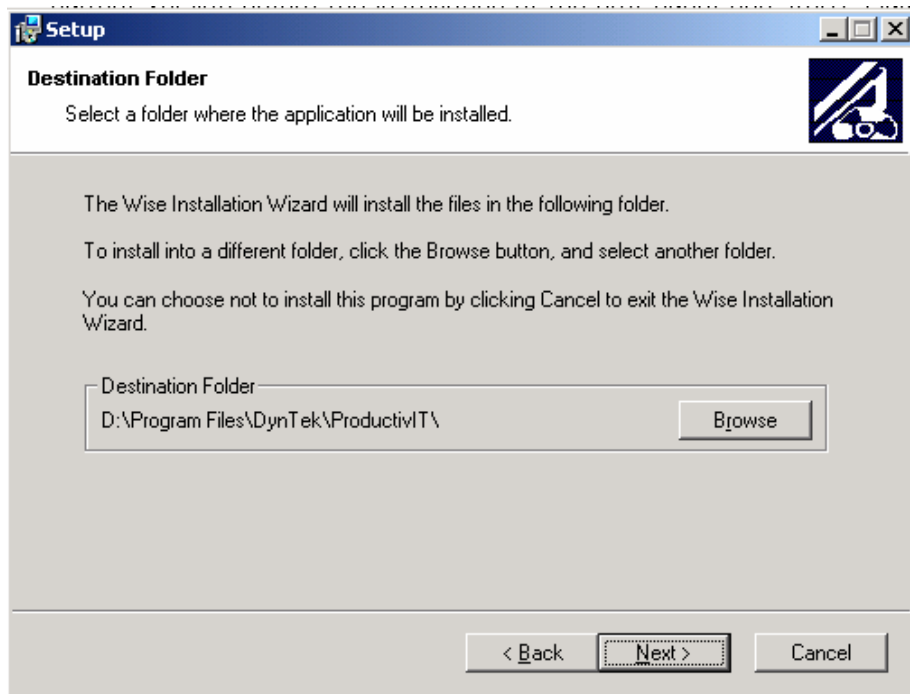


Figure 12

STEP 3. This is your last opportunity to change data or abort the installation routines. Pressing the Back button will take you back to Step 1, pressing the Cancel button will cancel the installation, and pressing the Next button will start the actual installation.

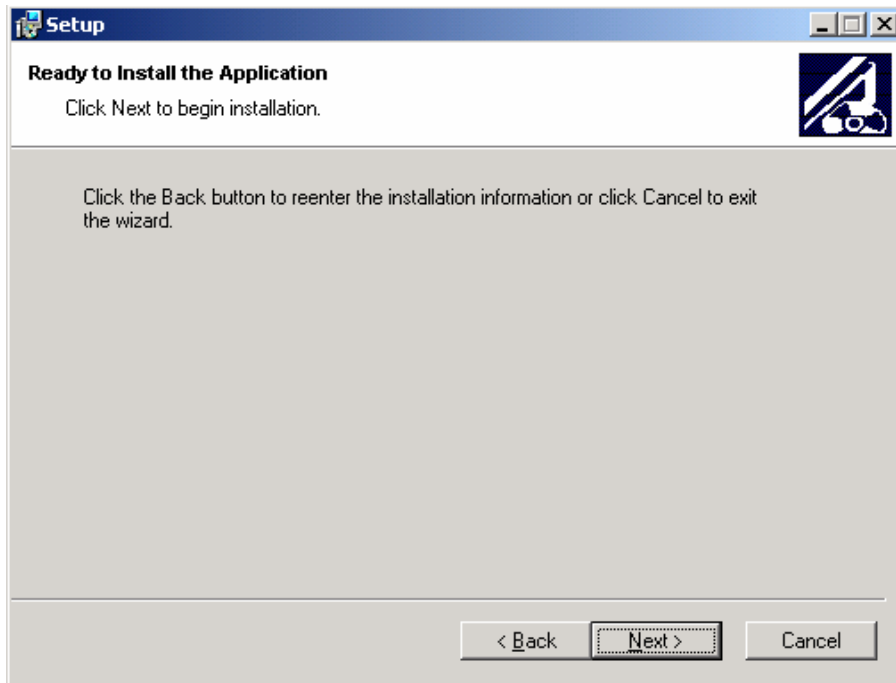


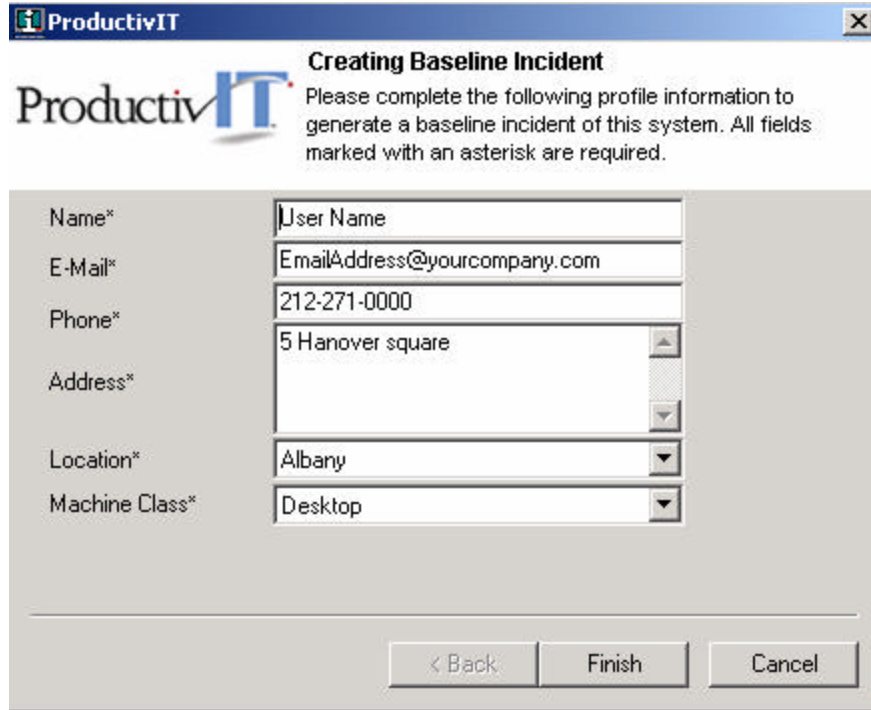
Figure 13

STEP 4. The creation of the baseline incident will appear after completion of the installation and prompt the user to provide or verify personal information, including department or workplace, phone number, etc. Each organization can specify exactly what type of information to request. Within the first few seconds of activation, the agent will capture the complete inventory of software and hardware as well as the configuration of the desktop. This capture generates an Incident Data File (IDF) called the Baseline IDF, which is then sent with the user registration information to the portal or support facility.

The Baseline IDF serves several purposes. It is a confirmation of successful deployment of the desktop. A report of baselines generated for a department or agency can serve as documentation of the deployment's progress. The Baseline establishes the beginning date of the desktop lifecycle and the point of reference for all software and hardware refreshes and seat or lease renewal dates.

The end user information collected at the time of Baseline capture is used to register the deployed seat with the IT department to begin the support process. Key information regarding appropriate service levels can be captured along with other Baseline data or can be associated with the user or machine ID at the central site through a reference table.

The information on this and the reset of the agent screens is completely customizable and you should reference the Customization guide for more information.



ProductivIT x

Creating Baseline Incident

Please complete the following profile information to generate a baseline incident of this system. All fields marked with an asterisk are required.

Name*	<input type="text" value="User Name"/>
E-Mail*	<input type="text" value="EmailAddress@yourcompany.com"/>
Phone*	<input type="text" value="212-271-0000"/>
Address*	<input type="text" value="5 Hanover square"/>
Location*	<input type="text" value="Albany"/>
Machine Class*	<input type="text" value="Desktop"/>

Figure 14

Periodic Updates

Each time a user activates ProductivIT to make a request or to report a problem, a complete data capture occurs at the desktop. This data is sent with the request as an IDF to the IDF repository. In an automated comparison of the new IDF to the Baseline, the help desk staff or field technician will be immediately alerted to changes in the makeup or configuration of the desktop. These changes are often at the root of the reported problem.

Periodic updates to the repository can also be scheduled to maintain an ongoing asset history. These can be set to occur on a specific date or can be scheduled on a weekly or monthly basis. The IDFs generated during these updates are added to the IDF repository where they can be accessed in a variety of asset reports.

Configuration

There are a number of configuration options both at installation time and at the time of creating the installation file. Below are some examples of configuration parameters and what and where the product can be changed.

Environment Variables

The environment variables in the following table are initialized when creating the installation.

Deployment, Installation and Configuration

BSBUILD_VARIATION	<p>The three-letter code that identifies the build variation.</p> <p>No Default.</p>
BSBUILD_COMPONENT_VERSION_SPECIAL	<p>A number that identifies the build variation.</p> <p>No Default.</p>
BSBUILD_INSTALL_MANUFACTURER_NAME	<p>The name of the manufacturer. This value is associated with the product and displays as the name of “Publisher” in the Control Panel “Add Software” configuration applet.</p> <p>Default: Dyntek, Inc.</p>
BSBUILD_INSTALL_URL_INFO	<p>The URL of the company information page. It is the link URL for the “Publisher” in the Control Panel.</p> <p>Default: http://www.Dyntek.com/</p>
BSBUILD_INSTALL_URL_HELP	<p>The URL that contains product support information. This URL is displayed with this string and also is a hotlink to this URL.</p> <p>Default: http://www.ProductivIT.com/</p>
BSBUILD_INSTALL_PHONE_NUMBER_HELP	<p>The telephone number to call for product support. Displayed in the Control Panel “Add Software” configuration applet.</p> <p>Default: +1 (212) 271-8550</p>
BSBUILD_INSTALL_DIR	<p>The program files installation directory.</p> <p>Default: Dyntek\ProductivIT</p>
BSBUILD_INSTALL_SHORTCUT_DIR	<p>The name of the “Start Menu” folder that contains the program shortcuts.</p> <p>Default: Dyntek ProductivIT</p>
BSBUILD_INSTALL_SUPPORT_ID	<p>The hexadecimal string support ID.</p> <p>No default</p>
BSBUILD_INSTALL_COPYRIGHT HOLDER	<p>The name of the copyright holder. This value displays in the installation wizard dialog.</p> <p>Default: Dyntek, Inc.</p>

Configuration Files

Policy.xml: This is per-customer information and the schema of this XML file describes special behavior with regard to different types of application crashes and special behavior for standard incident events (e.g. Support Request, Program Test, VxD Crash, &c). It includes a number that indicates the first panel that CrashWizard should display on startup. It is read by the BugSolver.exe application.

AR_00001.rcx: This is per-customer information and the schema of this XML file generally describes the contents of the CrashWizard dialogs.

RC_00001.rcx: This is per-customer information and the schema of this XML file describes what information is to be collected by the various CodeWatch collectors and which collectors should be launched to collect the information. Included in the information in these files is the list of registry items and environment variables to be collected.

***.ico and *.bmp files:** This is per-customer information of icons and bitmaps that the CrashWizard component currently reads these types of files for customization.

Registry Items

These items are stored in HKLM/Software/AstraTek/BugSolver:

“Controls” Key

Control has a sub-key for each of the ActiveX controls implemented by ProductivIT and each sub-key contains two values:

Name	Description
CLSID	Presumably the GUID of the ActiveX control.
OSs	A comma-separated list of operating systems and version.

The sub keys on one installation are:

- ? Device Control
- ? Directory Control
- ? Dll_Exe_Collector Control
- ? Event Log Control
- ? ProcessGatherer Control
- ? Registry Control
- ? SysInfoCollector Control
- ? Transport Control

“CW_Info” Key

This key contains several values and one sub-key, all described below:

Name	Description
Active	0 for yes and 1 for no
BuildNumber	The product build number of the product.

Deployment, Installation and Configuration

CW_Date	A string that describes the date of the build.
InstallComponents	NOT USED
LogDir	This refers to the directory that contains the log file.
OldAeDebugAuto	This contains a copy of the system AeDebug/Auto registry value before it was overwritten during installation.
OldAeDebugDebugger	This contains a copy of the system AeDebug/Debugger registry value before it was overwritten during installation.
ProductCode	This contains the product code GUID for the product
ProductVersion	This contains the version number string for the product.
SPVersion	NOT USED

The "Protocols" sub key contains protocol names used by the Transport component to send the incident files to the appropriate place

"Debugger" Key

This key contains two values:

Name	Description
BugSolver	System value used to tell ProductivIT how to invoke BugSolver.
Debugger	System value used to tell ProductivIT how to invoke Debugger.

"Ver 1.0" Key

This key is an extension of the values in the "CW_Info" key described above. Its values are:

Name	Description
AutoRas	Always turned off
BugSolverTested	No Longer used
CrashWizard	The fully qualified path name of the CrashWizard program.
DataDir	This refers to the directory that contains the data files.
InstallNo	This monotonically increasing value indicates the installation version number. It is passed to the server when querying for a product update.
IsCrashWizardOn	Debugging entry
LogDir	This is a duplicate of the value under the "CW_Info" key.
LogFile	This is the fully-qualified path name of the log file
NOBS	Debugging entry
ProductName	This is the product name.
Profile	This is the fully qualified path name of the profile.cpf file.
ReasonProfile	This is the fully qualified path name of the ar_00001.rcx file.
SentFailedNotice	This value contains the message text for a web send failure notice.
ServerName	This value contains the name of the web server.
SilentMode	Debugging entry
SupportID	This is the customer support ID.

Deployment, Installation and Configuration

Name	Description
UserMail	User Email Address
WebSite	Contains the web site to send the incidents

User Registry Items

These items are stored in HKCU/Software/BugSolver.com:

GUID Key

Unique key for this desktop

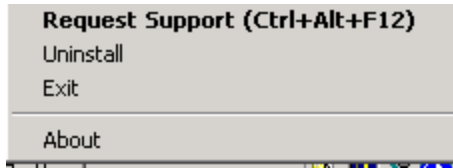
"BugSolver" Key

Contains per-user information

Name	Description
Address	The per-User "address" field.
Department	The per-User "Department" field.
Location	The per-User "Location" field.
MailAddress	The per-User "MailAddress" field.
PhoneNo	The per-User "PhoneNo" field.
SerialNo	The per-User "SerialNo" field.
User Name	The per-User "User Name" field.

Activating the Agent

Agent Taskbar



The ProductivIT Agent Taskbar Applet is the always-resident portion of the ProductivIT Agent. The user interface (which can be disabled) consists of a taskbar icon menu and a global "hotkey." Since the Agent Taskbar Applet is the only Agent application that is always running when a user is logged into a system, it also serves to perform background tasks on a regular schedule. Conversely, **if the Agent Taskbar Applet is not running, then no background tasks are performed.**

In typical configurations the Agent installation program starts the Agent Taskbar Applet as the last step of installation and the Agent Taskbar Applet starts whenever a user logs into Windows. When running, the Agent Taskbar Applet displays its icon on the taskbar below the taskbar clock. Right-clicking the icon presents a menu allowing the user to perform one of the above-listed functions. A typical configuration would include launching programs to perform these functions:

- ? **Request Support** – generate a ProductivIT "request support" incident. (This function is also typically assigned to the hotkey.) (Provided by launching BugSolver.exe with appropriate parameters.)
- ? **Uninstall** – uninstall the Agent software (Administrator only). (Provided by launching the Windows Installer with appropriate parameters).
- ? **Send Baseline** – Send a "baseline" incident. (Provided by launching BugSolver.exe with appropriate parameters.)
- ? **View Unsent Messages** – View messages pending in the send queue. (Provided by launching BSSpoolMan.exe.)
- ? **Exit** – exit the program, removing the taskbar icon and ending scheduled activity.
- ? **About** – a standard "about" box.

The menu is per-customer configurable. The text in the menu and whether or not a menu item is present is configurable.

In a typical configuration, double-clicking the icon does the same as the "Request Support" menu item: it generates a ProductivIT "request support" incident.

The background tasks that the Agent Taskbar Applet performs are:

Activating the Agent

- ? **Launch Programs**
- ? **Receive Files from Server**
- ? **Launch (Received) Files**

The background tasks are performed periodically, controlled by registry environment variables that specify the time interval for each task. **The Agent Taskbar Applet must be running in order for the scheduled tasks to be performed.**

A typical configuration will schedule the BSSpoolMan application to periodically send files. Some configurations may schedule the BugSolver application to periodically send a system snapshot (e.g. for monitoring assets).

The Agent Taskbar Applet also has a single command-line function that tells it to terminate all ProductivIT tasks in anticipation of product un-installation or repair.

When the Agent Taskbar Applet detects that the user has pressed the hotkey combination, it performs the "Hotkey" action. If the hotkey value in the registry is 0 (zero), the hotkey function is disabled. (This component does not provide a method of changing the hotkey value, but only reads the value defined in the registry.)

When run without arguments, the Agent Taskbar Applet operates normally. A single, optional, parameter is permitted, which specifies an "action" code that must correspond to one of the actions configured in the registry. If such an action is specified, the applet performs the action and exits, without any other processing normally associated with the Agent Taskbar Applet.

For example:

```
APPLET.EXE Action1
```

With this example, the application performs the "Action1" action and exits.

Command Line Interface

There are a number of command line interface parameters that you can invoke with ProductivIT. Below is a list of the options available

For bugsolver.exe

```
BugSolver [{-/?}] [{-/}{a|A}] [{-/}{b|B}] [{-/}{c|C} s] [{-/}{e|E} n] [{-/}{h|H}] [{-/}{i|I}] [{-/}L] [{-/}] [{-/}{m|M}] s [{-/}{n|N}] [{-/}{p|P} n] [{-/}{r|R} s] [{-/}{s|S}] [{-/}T] [{-/}t] [{-/}{u|U}] [{-/}{v|V}] [{-/}{x|X}] [<StatusFile>]
```

Parameter	Title	Description
?	HELP	Display a page listing all the available parameters for invoking BugSolver
{a A}	Set Permission	
{b B}	Blue Screen	Test a Blue Screen Crash Wizard interface
{c C}	Company Name	Include the Company Name in the Call
{e E}	Event ID	Pass an Event ID

Activating the Agent

{h H}	User Activated	Request For Support
{i I}	Install BugSolver	Register all components for ProductivIT
L	licensed Incident	Include the licensed by company with interface
l	licensed Incident2	Include the licensed by company with no interface
{m M}	Software Name	
{p P}	Process ID	Pass a Process ID
{r R}	Reason	Invoke an incident with a specific reason code
{s S}	Silent	Invoke an incident without invoking the user interface
T	Scheduled1	Scheduled Baseline with interface
T	Scheduled2	Scheduled Baseline with no interface
{u U}	Uninstall	Un register all components
{v V}:	Verbose	
{x X}:	No Crash	No Crash detection

Examples of usage Issue:

Example	Usage
bugsolver.exe -p 0 -e 0	Issue baseline with 0 process id and 0 event id
bugsolver.exe -T	Issue scheduled incident with interface
bugsolver.exe -t	Issue scheduled incident without interface
bugsolver.exe -L -m "abc.exe" -c "LaborSoft" -r "Software Problem"	Call ProductivIT from program "abc.exe" licensed by company "LaborSoft" with reason "Software Problem", show interface
bugsolver.exe -l -m "abc.exe" -c "LaborSoft" -r "Software Problem"	Call ProductivIT from program "abc.exe" licensed by company "LaborSoft" with reason "Software Problem", hide interface

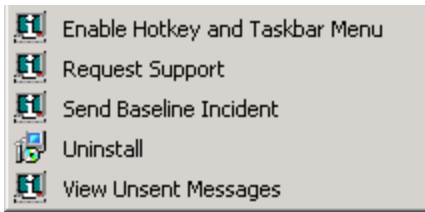
For Test_Bs.exe

Test_BS [{-|/}{t|T}] [{-|/}{h|H}] [{-|/}{x|X}[e|d]] [{-|/}u{a|A}] [{-|/}U{a|A}] [{-|/}{s|S}]

Parameter	Title	Description
{t T}	Hotkey/Baseline	Invoke the hotkey and issue baseline if no baseline has been sent previously
{h H}	Hotkey/Baseline	Invoke the hotkey and issue baseline if no baseline has been sent previously
{x X}	Kill it	Close the about dialog box
u{a A}:	update	Update the registration in interactive mode
{s S}	Show	Do not show about dialog box boxes and make a GPF

Start – Programs Menu

You can activate the agent from the start programs menu, which is identical to the agent taskbar mentioned earlier. The standard start menu includes the following items:



Crash Mode

If desired, ProductivIT will invoke at the time of a crash or system event crash. The product will act as the just in time debugger to send incident data to the portal.