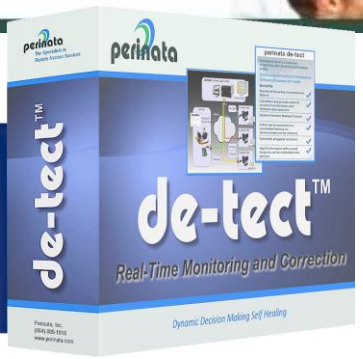
perinata
"The Specialists in Remote Access Services"

de-tect™
perinata de-tect



Perinata de-tect is a real-time monitoring and corrective/self-healing action.

Continuous monitoring of hardware and software for health

Benefits

- Reactive and Proactive corrections to failures
- Can detect and provide action to recover from Hardware or Software discrepencies
- Dynamic Decision Making Process
- Actions can be executed in a co-ordinated fashion on devices/nodes on the network
- Extensible pluggible structure
- High Performance with a small footprint, can be embedded into devices

Perinata de-tect is a better, low-cost way to monitor system health.

IT monitoring tools, CPU performance monitors and Machine Management Systems have all been implemented to provide the IT staff with important information and alerts about system performance.

Still application errors, network traffic errors, switching issues, upgrades and capacity problems continue to plague system performance. Common methodologies for mitigating these problems are first to "over build" the system to handle unexpected loads. Second, IT designers build in redundant systems to handle failure. Third, businesses rely on a large staff of highly trained IT technicians to carefully watch over the system and react to any alerts or failures as soon as they are noticed. Then, they apply their trouble-shooting skills to overcome them and correct the system discrepancy. These are all appropriate and necessary steps. But, can measures be taken on the system to manage some discrepancies that may lead to a larger problem?

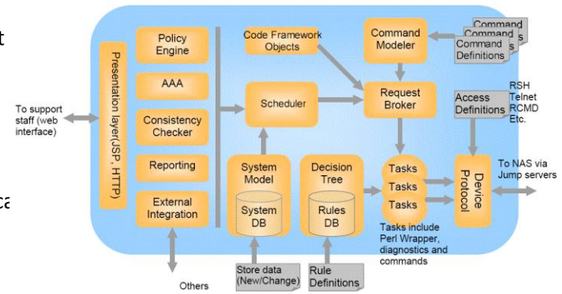
Perinata de-tect subscription service was built to capture the useful system health information and consolidate it to pre-empt problems or catastrophes. It uses consistency checking across the system to validate business process success and automated response to correct known issues without interfering with application performance.

How does perinata de-tect Work?

Perinata De-tect provides a distributed architecture with Server and Agent.

The Detect Server is a J2EE application server based enterprise application. It provides the following management functions:

1. Device connectivity management
2. Remote device access
3. Remote device management
4. Presentation extensibility
5. Policy management for authentica
6. Scheduler for device operations



Logical components

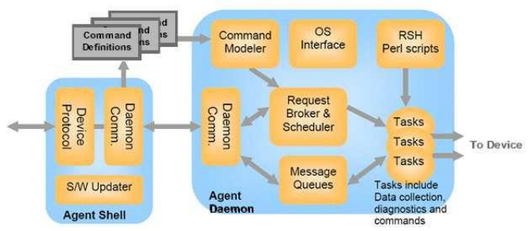
The diagram above gives an overview of the logical components in the De-tect Server.

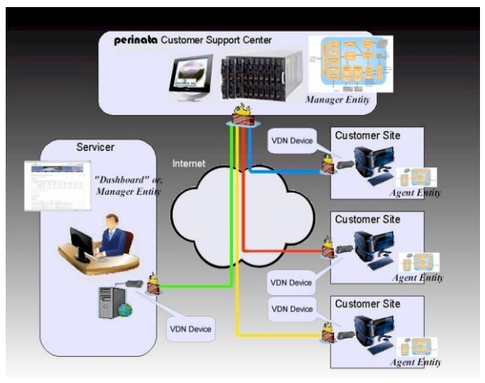
The De-tect Agent is a scalable device management agent that can reside at the target location on a pc and can run on many operating systems. It acts as an extension to the iDEAS™ server to implement the new process management strategy. The iDEAS™ agent can:

1. Reside on a device
2. Provide a device control protocol for management
3. Provide extensible command definition interface
4. Provide wrapper support for perl scripts or other pre-existing programmed calls
5. Provide means for simple decisions to be implemented as Tasks
6. Scheduler for local job

Logical components

This diagram gives an overview of the logical components in the IDEAS Agent.





perinata

"The Specialists in Remote Access Services"

The De-tect server components are described below:

1. **Command Modeler**: The command modeler provides a format for defining new device and command definitions.
2. **Request Broker:** Request broker is a task manager. It is used for starting, monitoring and terminating task instances.
3. **Tasks:** Tasks are basic unit of operations defined in De-tect. *The task can execute <u>any</u> operation defined on <u>any</u> device in the system.*
4. **Device Protocol:** The module that provides means to send and receive concise commands to the De-tect Agent in the device.
5. **Decision Tree:** The module that implements the rules for deciding the next step when an error occurs.
6. **System Model:** The set of objects that provide abstraction for the presentation layer.
7. **Code Framework Objects:** These objects implement code frame objects like Event, event correlator, generic command, device and workflow for the server.
8. **Scheduler:** The scheduler is responsible for scheduling one time and repetitive tasks actions like backup monitoring etc.
9. **Presentation layer:** This layer consists of set of JSPs and HTML that implement the user interface for the server.
10. **Policy engine:** The policy engine records and disseminates policy for the system. Parameters can be set to control behavior of the server.
11. **AAA:** this module provides authentication, authorization and accounting for users defined in the system. Fine grain user authorizations can be provided if so desired.
12. **Consistency Checker:** This module keeps track of the configurations at each of the system.
13. **Reporting:** This module provides means for generating reports from the server.
14. **External Integration:** This module exposes select interfaces of the Framework APIs for integration with external applications like Remedy etc.

The Agent consists of two separate executables:

1. **Agent Shell**: This is invoked from command like to enable telnet integration. This communicates the protocol inputs with the Agent daemon. This also provides means to upload new Operation definitions and software updates from the server.
2. **Agent Daemon**: This is the core Agent. This continuously runs in the device and executes commands and returns results to the Agent client.

The logical components are described below:

1. **Device Protocol:** This is the client counterpart for the server Device protocol module.
2. **Daemon Communication:** This module provides communication with the Daemon. This communication may be in form of binary packets.
3. **Command Modeler:** This module provides a way to define and load new commands into agent. This can be typically implemented as a dynamic library loader.
4. **OS interface:** This module provides integration with the Operating system for automatic restart upon crash etc.
5. **Scheduler**: The scheduler is responsible for scheduling one time and repetitive tasks actions like backup monitoring etc.
6. **Request Broker:** Request broker is a task manager. It is used for starting, monitoring and terminating task instances.
7. **S/W updater:** This module is used for updating the agent software automatically.
8. **Message queues:** When network limitations do not allow "push" messaging, diagnostic results or results of scheduled jobs cannot be immediately conveyed to the server. The results and events generated are temporarily stored in the message queue awaiting the next contact, or "pull" from server.
9. **Tasks:** Tasks are the basic unit of operations defined in De-tect framework. A task can be to restart and monitor a backup job. It can be used for executing a diagnostic command etc. The task can execute any operation defined on any device in the system.
10. **RSH, Perl Scripts:** these are collection of scripts to provide some avenues for tweaking the operation of commands if commands are chosen to be implemented via perl scripts.

Applicability

The following table explains how the guidelines specified in previous chapter can be met by de-tect.

Guideline	Support in de-tect	
Distributed Architecture	Yes	De-tect Framework is an inherently distributed agent that can be deployed in every site or device.
Extensibility: New Error codes	Yes	Device Manager and Rules DB provide means to add new error codes for process management.
Extensibility: New commands	Yes	Device Modeler in server and Command plug-in architecture in agent provide means to add new commands on the fly.
Extensibility: New Rules	Yes	Rules DB provides means to add new rules for process management
Changes in access mechanism	Yes	Device Protocol provides flexible access mechanisms
Extensibility: User interface	Yes	All management functions shall be available as EJB in application server. Any flexible user interface can be developed using calls to the EJB and JSP.
Extensibility: Hooks for integrating with other enterprise applications	Yes	J2EE inherently provides hooks for integrating into a variety of enterprise applications
Scalability	Yes	The De-tect architecture can scale to many 1000s of managed devices. This is because complex operation can be carried out by the agents without server becoming a bottleneck.
Non-Exclusive	Yes	If the access mechanism is Telnet. Existing perl scripts can be used for process management without affecting the operation if De-tect Agent or Server. The perl scripts and the De-tect framework can co-exist and provide smooth transition.

Phone: (864) 885-1918

email: info@perinata.com

website: www.perinata.com