Monitoring 2,000 To 25,000 Servers

With Argent

Executive Summary

Windows Magazine said it best in its independent review of Argent:

> "With a small resource footprint and an intuitive GUI, Argent scales to handle from the smallest to the largest of networks"

In essence, Argent is engineered to be easy to install, easy to learn and easy to use.

Argent isn't too daunting for small customers (compared with other large enterprise software products), yet it can easily scale for customers with 10,000 or more servers and devices.

This white paper focuses on how Argent scales to the largest of networks -- with case studies on how our largest customers tackle important real-world issues including scalability, failover, load-balancing and security.

Argent doesn't have a "cookie cutter" architecture for all customers – Argent provides powerful and flexible architectural components that can be mixed-and-matched to meet the unique requirements of all large enterprises.

Architecture

Understanding how Argent scales requires some basic information on Argent's architecture.

All Argent products consist of two core components:

Scheduling Engine Responsible for supervising, and scheduling tasks

Monitoring Engine Responsible for performing the monitoring (getting performance counters, etc.)

The heart of Argent lies within a **central database**, which holds all reporting data, monitoring configurations and lists of servers and devices. The **Mother Engine** is the core engine connected directly to this database, and can perform monitoring of its own since it also has a scheduling and monitoring engine built-in.

As part of our agent-optional architecture, remote agents can be deployed known as **Daughter Engines**. These Daughter Engines also have a scheduling and monitoring built-in, and periodically contact the Mother Engine for new servers and configurations, updated schedules, and offloads the data it has gathered for the Mother Engine to store into the central database.

The relationship between Mother Engines and Daughter Engines is clear – a single Mother Engine can have multiple Daughter Engines – e.g. multiple remote agents.

The power behind this concept lies with Daughter Engines having an innate failover and load balancing capability – if the link between the Mother Engine and the Daughter Engine is down, since the Daughter has a copy of the complete schedule of tasks to monitor, it can function perfectly well on its own. This means no critical data is ever lost. Argent's **Backup Console** component can also be deployed to ensure alerts continue to be sent even if the Mother Engine is unreachable.

On top of all this, Argent provides load-balancing for Mother Engines, in the form of an advanced architecture known as **Non-Stop Motors**. Non-Stop Motors are essentially a pool of Mother Engines in an active or active/passive cluster (for disaster recovery purposes) – the Daughter Engine concept remains the same, with the only difference being that Daughter Engines can now connect to any of the Non-Stop Motors, instead of just one Mother Engine.

How One of the World's Top Law Firm Does It

Server Count	15,000+
Server Distribution	Highly distributed worldwide
Load Balancing Requirements	High
Failover Requirements	Medium-High
Architecture	Multiple Independent Mother Engines
Argent Administration	Centralized

One of the world's top five law firms has been using Argent for the past 10 years, and they have dozens of offices spread out the world.

The law firm uses three independent Mother Engines that don't talk to each other – one in North America, one in Europe and one in Asia. In essence, they have three completely separate monitoring environments.

Monitoring of satellite offices in each continent are handled by 120 Daughter Engines. With such a large number of servers distributed in many locations – **their main concern, mostly politically-charged, is separation of regions.**

In other words, if the North American site goes down, the European IT Director doesn't want the outage affecting the European servers in the slightest.

Spreading out the Mother Engines also provides obvious load balancing. Instead of a single Mother handling 120 Daughters Engines, the load is split across three Mothers.

Failover is achieved by having Daughter Engines do all the work – if the link is severed, the Daughter Engine continue monitoring its own network segment. If the Daughter Engine itself goes down, an Argent **Backup Daughter Engine** is assigned and Argent automatically sends all tasks to the backup until the primary comes back online.

While the *servers* are distributed, the *management* of Argent falls to a single team of four staff members in Asia who, depend on Argent to keep the business running.

For them, management is simple – there are a standard set of template Rules of what to monitor – and they import the standard definitions to all Mother Engines.

The team enjoys the clear distinction between regions – it keeps the server list at each region relevant and clean.

For instance, when they access the Asian Mother Engine, they know all Rules, Alerts and Monitoring Groups are *only* related to Asian servers

Access to Argent is audited internally by a CRC-protected database table that Argent tracks, as well as user-based security restrictions that can be configured to control what a user can do, and whether they have read/write access to that area.

How One of Europe's Largest Pharmaceutical Companies Does It

Server Count	10,000+
Server Distribution	Concentrated in two data centers
Load Balancing Requirements	High
Failover Requirements	High
Architecture	Non-Stop Motors
Argent Administration	Distributed across different teams (Unix, Windows, Databases, etc.)

One of Europe's largest pharmaceutical companies is largely Unix-centric. Of course, this poses no issues as Argent fully supports all of the major Linux/Unix flavors, iSeries, and Windows, with respective failover and load-balancing agents available for each platform.

The servers are concentrated in two data centers, both are connected via an ultra-high-speed internal network connection. All servers are under the same domain and all subnets can be reachable on most servers.

Argent truly runs this customer's business. And since everything is essentially in one place, their requirements are broken down as follows:

- Reporting data must always be available
- Monitoring cannot skip a beat
- Access to operational data must be heavily restricted

After consultation with Argent, this company elected to use Non-Stop Motors.

To recap, Non-Stop Motors is Mother/Daughter architecture on steroids – instead of a single Mother, there are now multiple Mothers that work in a pool, similar to clustering.

If any Non-Stop Motor goes down, the others will seamlessly pick up the tasks. Tasks are also automatically load balanced across the available Non-Stop Motors.

The database, being the central repository of information, becomes the most vulnerable component. To combat this, the customer has SQL clustering, mirroring, and daily backups to ensure no data is ever lost.

The database also plays a pivotal role in the synchronization between Non-Stop Motors – it acts as the medium of communication so that Non-Stop Motor *X* understands that a task was completed by Non-Stop Motor *Y.*

Non-Stop motors scale by simply adding additional Non-Stop motors. Again, no reboots or planned outage periods are required – Argent simply snaps-in a new motor.

However, it is worth noting that more doesn't always mean better. Adding more motors than required means unnecessary load on the database due to the overhead that comes with synchronizing yet-another motor.

This is where Argent's consulting comes into play – every environment is different and the amount of monitoring work required for each customer varies greatly. Server count gives us a general idea, but the actual monitoring load depends on customer policies and practices.

While the servers are concentrated, the company has different teams heading off different applications and operating systems. For example, they have a Unix Team, a Windows Team, a Citrix Team, a Network Team, a Database Team and an Exchange Team.

All of these teams have no business looking at servers that aren't under their sphere of influence, and politically, teams don't *want* other teams to see or touch "their" servers.

The solution is Argent Global Manager – Argent's web-based GUI that provides "views" that restrict what Rules, Relators, servers, and Monitoring Groups a logged-in user has access to. With complete integration with Active Directory, this company simply leverages off existing Active Directory security groups (say, the "SQL Enterprise Users Group") and assigns different Argent Views to different security groups.