

Build vs. Buy: Designing an Effective Software Update Delivery Solution

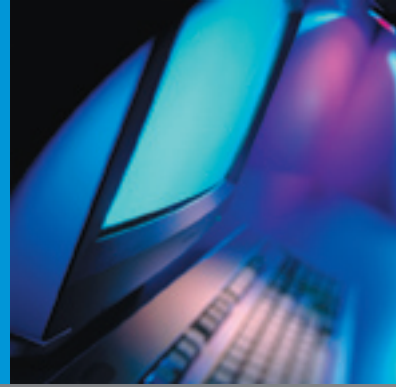


TABLE OF CONTENTS

Introduction: Build It or Buy It	2
Different Software Update Delivery Requirements	3
Product Manager Requirements	3
Consumer End-User Requirements	4
IT Administrator Requirements	5
Implementing a Software Update Delivery Solution	6
Client Component	7
Server Component	7
Server Component for Publishing Updates	8
IT Administrator Component	8
Additional Challenges to Developing an Update delivery Solution	9
It Requires a Diverse Team of Experienced Developers	9
It Requires an Elaborate QA Environment	9
It Requires a Stable System	9
It Requires Continuous Improvements	9
An Alternative to Building Your Own Update Delivery Solution	9
About FLEXnet Connect	9
From the Trusted Name in Software Updating	10
Summary	10

Build vs. Buy: Designing an Effective Software Update Delivery Solution

Introduction: Build It or Buy It

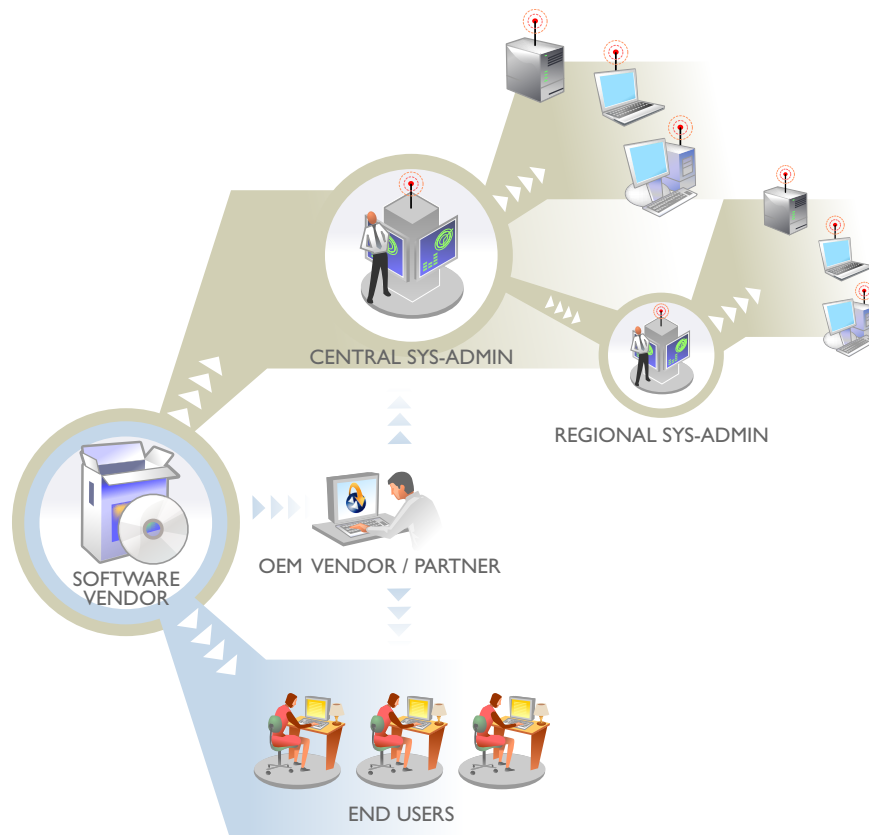
For software producers, the ability to effectively deliver new software and data updates, patches, and bug fixes to every customer is vital to the success of any product line. Keeping customers on the most current software version optimizes user productivity and satisfaction and significantly reduces the vendor's customer support costs.

Unfortunately, most software producers today have no means of proactively delivering updates to their user base. Many still rely on ineffective updating methods, such as mailing update CDs to every customer or trying to alert users through email that new updates are posted on their Web site. Manually shipping update CDs is slow and often expensive while still requiring end users to do too much work to install the update, and posting new updates to a Web site assumes customers will actually take the time to visit the site, find the correct file, and install it properly. Neither method is reliable, nor do they give software producers a way of knowing which customers have successfully installed the latest updates.

The easiest and most effective way to ensure customers have the latest updates and patches is for software producers to use an electronic software update delivery solution with built-in usage reporting functionality. Such a

solution enables software producers to proactively deliver updates and patches to their user base, as well as receive real-time usage data reports detailing how many users have installed any update, the total number of active users of an application, and more.

Once a software producer decides to use an electronic update delivery solution, one question that usually arises is whether the software producer should purchase the solution or attempt to build their own updating tool. On the surface, building an update delivery solution may seem like a relatively simple task, one that might take a few developers just a month or two to accomplish. But creating an effective update delivery solution is far more complex than it might seem. The following document details some of the requirements a development team must consider when designing their own software update delivery solution. It also discusses several of the technical components developers must program into the solution, and a few of the additional challenges they must overcome in order to create and maintain an effective software updating tool. Finally, this document introduces an alternate for software producers contemplating building their own update delivery solution.



Different Software Update Delivery Requirements

When developing an effective software update delivery solution, you must be constantly aware of the many requirements of both the customer receiving the updates and the software producer Product Manager responsible for update distribution. The figure below depicts the relationship between the software producer and its customers when it comes to update delivery.

The following section examines some of the requirements of the Product Manager and the customer. Because the customer receiving the updates can be different depending on whether you sell software to consumers or enterprises (as shown in the figure above), the customer section is broken up into two parts: the consumer end user and the IT administrator.

Product Manager Requirements

When designing an update delivery solution you must consider the following requirements of the software producer Product Manager tasked with handling the publishing and management of the updates:

- **Product Managers need a proactive, simple way to deliver updates.** Getting customers on the current version of your software improves user satisfaction

and reduces customer support costs, but delivering updates and patches to customers can be a never-ending headache for Product Managers. They need an easy way to proactively notify all their end users of new updates and deliver and install those updates to their customers' systems.

- **Product Managers need a non-programmatic way to publish updates.** Product Managers aren't developers; they shouldn't be required to make source code-level changes, or hand edit raw data files each time they have to deliver a new update. An effective update delivery solution provides Product Managers with an easy-to-use interface that walks them through the update publishing process.
- **The update delivery solution must have role-based security.** Because an software producer might need different people to access their software update delivery solution, it must have role-based security that lets Product Managers define the access rights and privileges of every user in their group. Additionally, the security model must support the definition of groups of users or product groups and restrict permissions based on these groups. The wrong user must never be allowed to access unauthorized data or perform restricted tasks, especially for other product lines they aren't associated with.

- **Product Managers should be able to access the solution from anywhere at any time.** Product Managers may need to access update information or publish new updates, patches, or hot fixes at bizarre times, and they can't always be sitting at their office to do it. They should be able to log on to the update delivery solution from anywhere at any time.
- **The update delivery solution needs to coexist with an software producer's update site.** For software producers that also enable end users to download updates from their Web site, the update delivery solution must coexist with this update site. This means that updates posted to your corporate Web site should be the same files used in the update solutions, which ideally would reference them directly. If two update sources don't have the same build, the software producer will be forced to waste time by doubling its QA effort, or worse yet, deliver the incorrect update files.
- **Testing updates before distribution should be easy.** Often Product Managers will want a quick look at the user experience of a new update before sending it to their QA lab or out to customers. An effective update delivery solution should provide testing functionality so Product Managers can preview the end-user interface, and download, apply, and test the update without having the associated application deployed to the other production users. An update delivery solution should also enable an software producer's QA lab to test the update process for new updates before deploying them out to customers.
- **Product Managers need real-time reporting tools.** Product Managers need to know if their updates are being successfully installed by their users. An effective update delivery solution should provide this information, but it should also enable Product Managers to access other types of real-time usage data, including key facts on the total number of active users of an application, the number of users on each software version, how quickly new product releases are used in the marketplace, and more. It should also tell Product Manager how many users clicked on the marketing or support messages they deliver to their customers' desktops.
- **Product Managers must be able to target updates to specific user groups and machines.** Often Product Managers publish updates that aren't meant for their entire user base. That's why they need their update delivery solution to make it easy for them to target specific user groups and, in some cases, particular customers' machines. The update delivery solution should enable Product Managers to enter in targeting conditions so updates are only sent to customers using certain operating systems or using an optional component of the software, for example.
- **Product Managers must be able to send updates only to users entitled to receive them.** Some software producers require customers to be part of a maintenance or subscription plan to receive updates. An update delivery solution should have functionality that automatically validates the entitlements of every user, so unauthorized customers never receive privileged updates. The update delivery solution should be able to authenticate customers either silently or by requiring the user to authenticate by entering data – like a serial number.
- **Product Managers must be able to target updates for multiple products or messages to multiple users.** Often a single update or HTML message will be meant for more than one version of a product, and sometimes an update or message is meant for more than one product. A Product Manager shouldn't have to waste time publishing the update or message separately for each product and version it targets. An efficient update delivery solution should enable them to publish the update once and have it delivered to everyone.
- **Product Managers need a way to stagger the delivery of updates.** An effective update delivery solution should enable Product Managers to distribute updates and messages to a small percentage of their customers before deploying it out to their entire user base. By staggering update delivery, Product Managers can get feedback on an update or message's performance from users and make any necessary changes before sending it to everyone. Staggered delivery also enables software producers to more evenly distribute the update load on their download servers.
- **Product Managers must be able to brand updates and messages to match their product line.** The updating experience presented to the end user should match the look, feel, and branding of the product itself. The update delivery solution should enable the Product Manager to brand the updates and messages so it offers customers a consistent end-user experience.

Consumer End-User Requirements

If you sell software to consumers or end users that can update their own machines, you must consider the following customer requirements when designing an update delivery solution:

- **Make the updating process simple for the end user.** Never assume your end users are technically savvy or will proactively search for new updates and patches for your applications. End users require an easy-to-use UI to notify them of the availability of new updates, and they need those updates to be easy to install. Even requiring end users to save and launch an update themselves is asking too much. The simpler you make the update notification, download, and installation process for your end users, the better. If end users can't do it all with one or two mouse clicks, too many of them will fail to install your updates.
- **The update process should be automatic.** Although end users should be allowed to approve any changes made to their systems, once an update has been accepted by the end user it should happen in the background.
- **Downloading updates should not tie up the end user's bandwidth.** When designing an update solution for consumer software, you must assume that your customers live in a dial-up world with extremely limited bandwidth. If the downloading of your updates takes too long or takes up too much bandwidth, end users will be reluctant to download and install your updates in the future.
- **Only bother the end user when there is an update.** End users want to be notified when new updates are available, but they don't want to be told that "No updates are available" every time they launch your application. An effective update delivery solution should check for updates in the background and only notify the end user when new updates are available. If no updates are available, it should remain silent.
- **The updating experience should be tailored to fit the application.** Not all consumer applications are the same, so they shouldn't all present the exact same updating experience to the end user. Your update delivery solution should make it easy to tailor the particulars of your product's updating experience to fit your application. For example, some applications should prompt the end user to install critical new updates the moment the app is launched, while others are better off notifying users of new updates right after the application is closed. The UI used to notify the user of new updates should have the same look, feel, and branding as the application itself to provide a consistent end-user experience. An effective update delivery solution must make it easy to do all of the above and more.
- **The update text should be in the end user's native language.** Providing an update message in English to a German end user simply doesn't work. An effective update delivery solution should translate update text into the native language of the end user, whether they speak Danish, Dutch, Swedish, or any other language.
- **The updating experience should be tailored to fit the end user.** Never force users to install updates that don't apply to them. An effective update delivery solution must be able to determine which software product and version each customer is running and display only the updates appropriate to them. It is important for an update delivery solution to be able to add conditions against file versions, file timestamps, and registry settings when targeting end users, so you can send updates only to customers using certain operating systems or using an optional component of your software. This will save you bandwidth and avoid sending unnecessary updates to your end users.
- **Your updating server should always be available.** One of the worst sins an update delivery solution can commit is being unreachable when an end user is trying to download an update or patch. That's why an effective update delivery solution must be able to efficiently process hundreds of individual user requests in milliseconds and scale to maintain steady performance as load/demand increases by simply adding extra servers. Your update delivery solution must be able to scale to meet the demands of your entire user base, and have redundancies in place so that if one server goes down, the others can handle the extra load seamlessly.
- **Don't send the end user messages that don't apply to them.** End users expect to receive informative messages about new updates, new product releases, and important technical support instructions, but they don't want to receive information that isn't relevant to the specific product and version they run. Just like you wouldn't send a customer an update for a product they don't own, you shouldn't bother end users with messages that do not apply to them. You must be able to target the messages you send based on product and specific version, the operating system they run, and more.

IT Administrator Requirements

If you sell software to enterprises, governments, hospitals, or any organization managed by IT administrators, you must consider the following requirements when designing an update delivery solution:

- **IT administrators need to be proactively informed of the availability of new updates.** Depending on the organization, a single IT administrator can be responsible for managing as many as several thousands applications. They have no time to regularly search your site for new updates and patches for your products. It is crucial that you are able to notify them when new updates are available for your applications in a way that they will take notice. If you don't, your new updates could get lost in IT's daily shuffle and never deployed.
- **IT administrators need complete control over update review, targeting, and distribution.** An effective solution must provide IT administrators with an easy way to control how they manage your updates, including when and how they access, review, target, and distribute your updates. They need to be able to control when (date and time of day) they check your server for new updates, including the ability to schedule weekly and monthly automated update checks. They must be able to optionally review all your updates' metadata, including a full description, file size, and any relevant installation instructions before they decide to deploy them. They should also have the option to bypass any review or testing and automatically deploy your updates as soon as they receive them. When preparing to deploy your updates, IT administrators should be able to easily target the machines in their organization they wish to receive your files – functionality should include the ability to target specific products and versions and add conditions against file versions, file timestamps, and registry settings. Finally, update distribution should be as simple as possible, so an effective update delivery solution should provide administrators with a choice of deploying using their software distribution system (i.e., SMS, ZENworks, LANDesk, etc.) or using your update delivery solution itself.
- **IT administrators need access to real-time reports on update distribution.** Did the update just deployed install successfully on every target system? If not, which systems still require the update? This is the kind of accurate, real-time reporting data IT administrators require from an update delivery solution, but it isn't necessarily the only data they need. Different customers will require different reports, and an effective update delivery solution should be easily customizable to accommodate different customer reporting requirements.
- **Updates must be dead simple to install and run.** The last thing IT administrators need is to have to waste time and resources figuring out how to install and run your updates. You need to provide them with an easy way to download your updates, and the updates themselves need to be bulletproof, requiring no end-user interaction.
- **IT administrators don't want to be forced to deploy updates.** You can't assume that your customers' IT administrators will want to deploy every update you create. An effective update delivery solution must provide IT administrators with the option of first reviewing and testing your updates before deciding whether or not to deploy.
- **Updates must be easy to manage.** IT administrators strive to reduce the TCO (total cost of ownership) of the applications they manage. The harder your updates are to manage, the higher your application's TCO. Your update delivery solution must simplify the entire updating process, from notification to testing to distribution. If not, you risk alienating customers and driving up your own support costs.
- **IT administrators must be able to easily pass the update to testing.** Many, if not all, of your customers' IT administrators will first test your updates before deploying them out to their operating environment. They may do the testing themselves, or they may pass the update on to regional or local IT teams to conduct their own tests for their particular systems. If they chose to pass on the updates, an effective update delivery solution should make it easy not only to pass on the update on, but for the local administrators to manage them, including easy review, targeting, and distribution.
- **IT administrators must be allowed to use their existing software distribution system.** If an enterprise uses a distribution system such as SMS, LANDesk, ZENworks, or Tivoli to deploy their applications, an effective distribution system must allow them the option of using it to distribute new updates and patches.

Implementing a Software Update Delivery Solution

On the surface, implementing an effective update delivery solution may seem like something one or two developers can handle in a few months time, but in reality it requires a far more significant commitment. The following section highlights the components that make up an update delivery solution and some of the programming and code that each component requires.

Client Component

The following bullets describe a few of the functions of an update delivery solution that require code to be resident on the target machine and integrated into the installed application and update.

An update solution must be able to:

- **Proactively and silently check for updates and only display a message when updates are available.** This requires the creation of Internet connectivity software, as well as source code to enable the solution to interact with databases.
- **Check for Internet connectivity and not prompt if the user is not connected.** This requires knowledge of Internet APIs for every platform targeted by the application being updated. The more platforms it targets, the more difficult and time consuming this becomes.
- **Read, store, and use proxy settings, including ID and password for authenticating proxies.** This includes auto-configured, browser-default configured, and script-based configured proxy servers.
- **Determine which version of the application is installed.** The client agent must be able to query the target machine's operating system resources to determine the current version, service pack, and patch level of the application. In the case of hot fixes or smaller updates, this determination may involve the presence and version of individual files or resources on the user's system, rather than querying more obvious version settings.
- **Call servers to find available updates.** This requires the definition and implementation of a remote API and handshaking algorithms. It requires extensive back-end support, including servers that are easily scalable.
- **Only display updates available for installed applications.** Once an update is displayed or installed, it should never be displayed again to that particular end user.
- **Use parameterized conditions to control which updates are targeted to users.** This requires building each condition per update, and, because of privacy issues, the update solution's client agent needs to run these conditions on the target system, rather than sending the information about the end user's system to the server for evaluation.

- **Display the user interface in multiple languages.** Many popular applications have a user base consisting of over 30 languages. Not only does supporting these languages require isolating and separating all the strings in the user interface, it requires manually translating text into each language, which can be extremely time consuming. A translated user interface also requires careful design to ensure that the layout accommodates strings that may be vastly larger or smaller depending on the end user's preferred language.
- **Download updates using checkpoint restart logic.** This ensures the integrity of the update if a connection to the update server is somehow interrupted in mid-download.
- **Download updates using bandwidth throttling.** The file transfer rate during update downloading should be able to increase or decrease based on the target system's available bandwidth, thereby preserving the user's interactive experience.
- **Verify the integrity of the downloaded file to ensure the full file was downloaded.** This can involve digital signatures or encryption technologies to ensure no third-party manipulations of downloaded binaries have occurred.
- **Run on multiple platforms and install OS-specific executables passing optional command-line parameters.** This requires knowledge of each platform an application targets and their proprietary installation formats, including MSI, MSP, EXE, and JAR. Additionally, if your client displays a user interface, it must be ported to each supported platform or developed in platform-neutral technologies.
- **Display a professional user interface.** The UI displayed to the end user needs to be polished and easy to use. This requires expertise in user interface design.

Server Component

The following bullets describe a few of the functions an update solution's server must perform to process update requests from the client.

It must be able to:

- **Retrieve a list of available updates from database on client request.** Different versions of an application will routinely contact the server for updates. The server must recognize them and deliver a complete list of available updates.

- **Send information on updates to the client.**
Updates are more than just files. They should have accompanying metadata (update description, file size, installation instructions, etc.) that should be transmitted and displayed to the end user.
- **Log requests for updates and updates distributed.**
This enables software producers to ensure all end users downloaded the update. Accomplishing this requires that transaction logs are stored into a database for reporting.
- **Easily scale as user base increases by adding additional servers.** If the user base doubles, an update delivery solution must be able to successfully handle the additional load by simply adding twice the servers.
- **Be able to process millions of requests per day.**
Depending on the application, end users might be contacting the server multiple times each day. Also, some updates require multiple server transactions. Each server must be able to seamlessly handle an extremely high load.

Server Component for Publishing Updates

The following bullets describe a few of the functions that the Product Manager's area of the update solution must perform.

It must be able to:

- **Enable Product Managers to easily add and update the list of available updates.** The user interface presented to Product Managers to manage their updates should be simple to use so that no coding is required to publish an update, but robust enough to enable Product Manager to effectively manage each update.
- **Create a logical relationship between installed product and available updates.** Product Managers must be able to assign a specific installation sequence for multiple updates.
- **Display a list of updates per product.** software producers usually have more than one product to update, so the update delivery solution must make it easy to view and manage updates on a per product basis.
- **Allow for easy editing of the update URL.** Product Managers need an easy way to enter and edit the URL where each update is stored.

- **View update and usage reporting.** Product Managers need easy ways to review accurate, real-time data about the adoption rates of their updates and the usage rates of their applications.
- **Create custom UI elements.** Different applications often require different update data to be entered into and captured by the update delivery solution.
- **Control access to software producer staff by product line and function.** Different Product Managers and internal staff need access to the solution for different reasons. The solution should have role-based security to ensure they only can only do authorized tasks on authorized products.
- **Allow segmentation of access by product line.** If an software producer has more than one product or product line, it may need to segment the Product Manager access to ensure that one Product Manager does not accidentally view or edit the update configuration settings for products that they do not manage. This will also simplify the number of updates that each Product Manager needs to view and manage.
- **Test the updates before publishing.** Product Managers need a way to ensure new updates will successfully download and install on the right end users' systems before publishing them to their user base.

IT Administrator Component

The following bullets describe a few of the functions that a customer's IT administrator's area of the update solution must perform.

In addition to needing to perform all the functions listed in the section above for Product Managers, the IT administrator component of the update solution must also be able to:

- **Synchronize with software producer's update server.**
IT administrators need an easy way to synchronize with the software producer's update server to pull down new updates and patches. Administrators need an easy way to schedule when this synchronization occurs.
- **Distribute updates to either end users or to other IT administrators.** If they are distributing the update directly to end users, they should be allowed to either use their own software distribution system or the update delivery solution itself. If they are passing it down to other IT administrators for testing and

distribution, those administrators also need a UI that performs all the same functionality.

- **Present a professional, polished UI.** If an update solution presents a software producer's Product Managers with an unimpressive UI that is difficult to use, that's bad. If it does the same to customers, that's unacceptable, and could cost a software producer significant revenue.

Additional Challenges to Developing an Update Delivery Solution

The following section details a few of the challenges software producers face when attempting to build their own software update delivery solution.

It Requires a Diverse Team of Experienced Developers

As suggested in the previous section, it is extremely difficult for a single developer to author and maintain an effective update delivery solution. Developing an update delivery solution requires substantial development expertise in creating client, server, and database code. While a developer may have sufficient experience in one or even two of these core competencies, they almost never have it in all three. It is therefore unrealistic to task one developer with the creation of an update delivery solution. It takes a dedicated team of developers with specialized skills, and it takes time – both of which cost money.

It Requires an Elaborate QA Environment

Another reason why creating an effective update delivery solution is difficult is that from a QA standpoint, it requires more than just basic functionality testing. You also have to conduct performance and scalability tests to ensure that the update solution can handle millions of end-user requests in a day's time and effectively scale by adding additional servers. You also need to conduct redundancy tests to ensure that if one server goes down, there will be no change in the end-user experience. Performance, scalability, and redundancy tests are specialized, requiring that you set up and maintain a dedicated QA environment, and many organizations do not have the experienced manpower and equipment to conduct them.

It Requires a Stable System

Developing the updating software and professional, polished UIs is not enough if the system isn't stable. If it loses contact with end users mid-download or periodically crashes, the solution is a failure. An update

delivery solution is similar to the unmanned shuttles and probes NASA sends to Mars. No matter how state of the art the machines may be, they are useless if they can't communicate with NASA. Stability in an update delivery solution is an absolute necessity.

It Requires Continuous Improvements

Technology and end-user needs are constantly evolving, and an effective update delivery solution needs to keep pace. So creating an updating tool isn't just a one shot deal – something developers can build once and then go back to developing your applications full time. It requires continuous improvements, and that means a substantial and long-term commitment from your development team. Plus, each new upgrade to the solution requires a new round of QA tests described above, further tying up hardware and personnel.

An Alternative to Building Your Own Update Delivery Solution

Instead of committing substantial developer resources to creating a potentially flawed update delivery solution, many software producers instead use FLEXnet Connect to update their installed products.

About FLEXnet Connect

FLEXnet Connect makes it easy for software producers to electronically deliver updates, patches, and bug fixes to any user in any operating environment. Whether you want to deliver updates directly to home consumers running Windows or to enterprises with Linux systems managed by IT administrators, FLEXnet Connect is the only solution you need.

FLEXnet Connect also enables you to deliver targeted HTML messages, including news about new product releases and upcoming events, directly to users' desktops. Plus, if your relationship with your users explicitly permits, FLEXnet Connect can provide you with invaluable data about your installed products, so you always know who is using what version of your software, the number of users who have installed a given update, and more.

FLEXnet Connect can update any application running on any platform, including Windows, Mac OS X, Solaris, Linux, AIX, and any flavor of UNIX. It has a secure architecture that scales to hundreds of millions of end users, making it ideal for software producers of every size, customer base, and budget. Plus, FLEXnet Connect is extremely easy to implement into your applications, so you can be up and running in no time.

From the Trusted Name in Software Updating

FLEXnet Connect is from Aceso Software, the company that develops the InstallShield and InstallAnywhere installation-authoring solutions. Since 1987, the name InstallShield has been synonymous with quality software installations and updating. Because end users are familiar with the InstallShield installation and updating experience, they are more willing to trust it and accept updates that follow its industry-standard format. It helps reduce customers' reluctance to adopt new updates and patches.

Summary

More and more software producers are using an electronic software update delivery solution to optimize the performance and stability of the software they sell and reduce their customer support costs. When deciding whether to purchase such a solution or create their own, software producers should be aware that creating an effective update delivery solution is extremely complex, requiring a substantial and long-term developer commitment. For software producers looking to save money by purchasing a proven, effective third-party update delivery solution, there's FLEXnet Connect from Aceso Software.

You can learn more about FLEXnet Connect, and optionally download an evaluation of the solution, by visiting <http://www.aceso.com/fnc>.



Acresso Software Inc.
900 National Parkway, Suite 125
Schaumburg, IL 60173
USA

Schaumburg (Global Headquarters):
+1 800-809-5659

United Kingdom (Europe,
Middle East Headquarters):
+44 870-871-1111
+44 870-873-6300

Japan (Asia, Pacific Headquarters):
+81 3-5774-6253

www.acresso.com