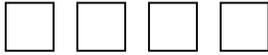


# Multi-Site Software Development – It's Not Just Replication Anymore

An MKS White Paper  
By David J. Martin  
Vice President  
Product Management



Organizations are increasingly moving all or parts of their IT operations abroad. There are many reasons why:

**Access to specialized talent** – If the best designers and developers cannot come to you, move your operations to them; scarcity of talent is also a factor here.

**Reduction of development costs** - Price of talent, ability to do more with less.

**Reduction in time to market** – Software development can be productive around the clock, enabling organizations to achieve 24/7 development operations.

**Proximity to the customer** –Ability to stay close to the customer’s requirements and needs.

**Acquisitions** – Multi-national organizations continue to merge and acquire other organizations with development operations.

Once a software development organization is geographically dispersed, it is critical that the lines of communication be more formal and development processes more rigorous, or an organization will not be able to achieve many of the business advantages and cost savings outlined above. Organizations with global development operations need software configuration management systems. These SCM systems must be designed and built to perform under distributed conditions. In the SCM industry, technology that enables this kind of distributed ability is called Multi-Site. But there are various interpretations of this term among users and among vendors. This paper outlines the various approaches to multi-site, and MKS’s approach to this technology with its SCM solution, MKS Source Integrity Enterprise Edition.

### **What is “Multi-Site”?**

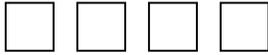
Multi-site software development refers to the ability for geographically distributed software development teams to collaborate on common software projects. More and more, the trend in enterprise software development is for multiple teams to collaborate on common projects from various locations around the continent, or the world. This creates unique challenges, the most important being visibility and timely access to the most up to date project information.

### **What Multi-Site Isn’t**

Multi-site is not repository replication. Repository replication is only one solution to the overall multi-site challenge, but the two terms have become synonymous because replication has been the predominant solution for some years.

This paper identifies and examines the most common problems associated with multi-site development and presents a new and innovative solution called the Federated Server Architecture (FSA). The FSA, developed by MKS, uses advanced and reliable caching technology while building on the widespread advancements in network availability and reliability. The result is a solution that has numerous advantages over the outdated replication paradigm:

- Lower total cost of ownership
- Less administratively burdensome
- Provides developers with real-time project information



## The Three Challenges to Multi-Site Software Development

### 1. Remote Access

Traditional software development occurred in environments where the source code was located on a local server with files available to developers over a LAN. This is still a popular approach for smaller teams that work in a single location. Enterprise software development, however, has led the way in innovative development practices by dispersing their development teams across the country (or world) to take advantage of natural efficiencies.

Increasingly, software development teams want to borrow or share developers and their expertise from teams that are geographically dispersed. In this case, teams become virtual development teams with no relation to geography but only to the project being worked on. This presents enormous efficiencies for the company and helps plug gaps at the team level. However, such a scenario can only succeed if the company has adopted a SCM and change management solution that offers multi-site capabilities. Aside from the efficiencies inherent in this scenario, software development teams are turning to multi-site development for the following reasons:

- Scarce availability of local talent
- Talent is cheaper in another location
- Specific talent in subject matter may reside in another location
- Staff reductions due to budget cutbacks
- Administrative overhead due to managing diverse tool sets for enterprise development teams

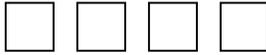
### 2. Slow Network

It is not reasonable to expect all geographically distributed teams to have high bandwidth connections between locations. And until optical networks are ubiquitous, the problem of slow networks is one that will dog companies for a long time to come. It is simply not feasible or efficient for most companies to transfer the bulk data from software projects between locations via their corporate network. Whether bandwidth or latency is the problem, software development can grind to a halt as developers wait minutes or hours for large files to be downloaded from a host server in another location or as they wait for access to project updates.

The solution to this problem, up to this point, has been to replicate source code repositories on a regular (usually daily) basis so that developers can access project information and files from a local server. Problem solved? Not quite. The problem with this solution is that the replication process is complex and totally dependent on the expertise of an administrator that must facilitate the replication and synchronization of project information. Not only is this administratively burdensome, it is also expensive.

### 3. Unreliable Network

Years ago, when replication technology was first introduced, one of the main problems it aimed to overcome was unreliable networks. If geographically distributed software development teams were sharing data over a corporate network there was a risk that the network would become disrupted in some way and the data would be lost. And since the networks were generally operating at slow speeds (see above), the potential time lost to incomplete transfers and the accompanying loss in productivity was a risk that was unacceptable to most organizations. That



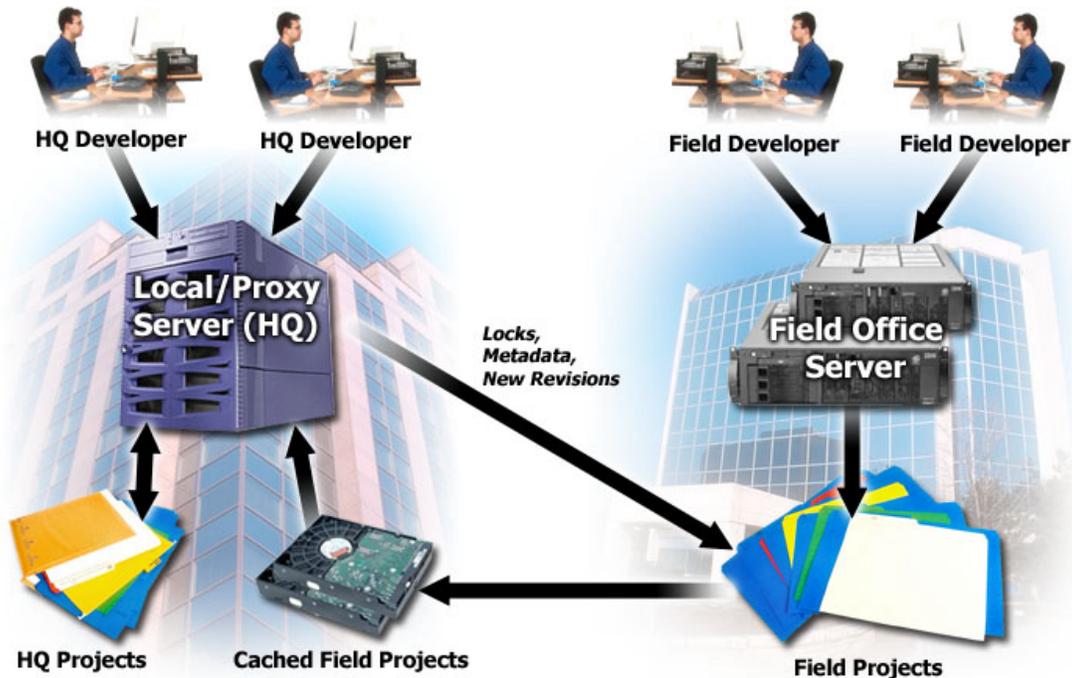
is why it was attractive to make copies of software repositories on local servers and then periodically synchronize changes with the master server via a network connection.

But technology has changed and networks have become vastly more reliable, making replication technology look antiquated because of its hardware-heavy and administration-intensive character. It is true that the risk of network downtime has not been eliminated completely, but it is now regarded by most companies as a manageable risk. It has become more manageable because of redundancy, better recovery and network software and a more mature network infrastructure the world over.

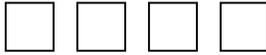
## Today's Solution to the Multi-Site Challenge

### The MKS Federated Server Architecture (FSA)

While repository replication has proven to be a workable solution for many companies, it was developed with yesterday's technology in mind. Slow and unreliable networks were the norm and disk caching was still a relatively expensive proposition, given the price of disk space. True to its reputation for innovation in the SCM market, MKS has developed a new approach to multi-site software development, called the Federated Server Architecture, which eliminates some of the main disadvantages related to replication technology: high cost of ownership, high administrative burden, and lack of real time visibility of project changes. It's another example of how MKS is concentrating on delivering best of breed solutions for the SCM and change management market<sup>1</sup>.



<sup>1</sup> Gartner's latest "Magic Quadrant" (Dec. /01) lists MKS as the leading visionary in the Event-Driven SCM category in terms of "ability to execute", ahead of Rational and Merant. For a full explanation of MKS's best of breed strategy see their white paper entitled "Best of Breed vs. End-to-End: Solutions for Application Development".



## FSA – Addressing the Challenges

### 1. Remote Access

Under the FSA, developers who are using MKS Source Integrity Enterprise Edition for their SCM activities have full visibility of remote source code repositories with their GUI, CLI or Web browser. A local server acts as a proxy for a remote server, which then stores necessary project information in a disk and/or memory cache and communicates with the host server via TCP/IP for updates. During a check out operation, data is checked out from the local server but the record that shows the transaction is written to both the local and host servers, thereby informing all developers, regardless of their location, of the status of the checked out file. Upon check in, data is checked in to both servers simultaneously and the record is, again, updated at both ends. The greatest benefit to this model is that, during concurrent development, project information is communicated and received in real time. This gives project managers the ability to make decisions based on the most up to date information.

Under the replication model, project files are also checked in/out from a local server, but the central server is unaware of the project changes until synchronization occurs a later time. The synchronization process is complex and administratively burdensome, which means that a company must pay an administrator to oversee the entire process, usually on a full time basis. It also means project managers are making critical decisions based on project information that is potentially days old.

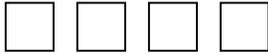
### 2. Slow Network

The great panacea of a ubiquitous optical network is still years off, so individuals and companies will have to make do with existing technology for the time being when transferring data. It is possible, however, to mitigate the problems related to slow networks using the FSA. For those cases where bulk data is being transferred between the virtual and host servers, MKS provides advanced compression functionality that dramatically improves the network performance, even under dial-up conditions.

Operation	Single Site		Multi-Sites	
	~100Mb/s latency =5us	~450kb/s latency=40ms	~450kb/s latency=40ms	
<b>Project View P1</b>	10.4s	241.0s	90.9s	2.6X Faster!
<b>Project View P2</b>	3.3s	119.5s	9.5s	12X Faster!
<b>Sandbox Creation P1</b>	150.2s	1112.45s	381.2s	3X Faster!
<b>Sandbox Creation P2</b>	65.5s	246.7s	132.2s	1.8X Faster!

**Fig 1.** Illustrates the performance improvement provided by the proxy cache . Results depend on the projects, computers and network settings.

The most important factor, however, is that the user recognizes no difference in performance whether they are working on a project based in a remote or local office. Using the FSA, the local, virtual server contains all relevant project info in its cache. Further, synchronization is an ongoing operation that is invisible to the user and requires no interference by an administrator. It also means that project information (i.e. locked files, changed files) is available to developers and project managers almost immediately, regardless of their location. The key point is that an



object is read only once over the slow network before being stored in the proxy cache and made locally accessible to all permitted users. In the non-multi-site case, the object would be read over the slow network separately for each user requesting it.

### **3. Unreliable Network**

Thanks to advances made in network technology (hardware and software) corporate networks are more reliable than ever before. Outages still occur, however, and it is important that developers have the ability to continue their work during network downtime. Developers who are using MKS Source Integrity Enterprise Edition for their SCM activities are covered in these situations. MKS's famous Sandbox™ technology allows developers to work on project files that have been checked out to their local workstation. They can edit and manipulate these files just as if they were still attached to the corporate network. When the network connection is re-established, they simply have to check in their edited files to their local, virtual server and the synchronization process with the host server will be performed automatically.

By the end of 2002, in a subsequent release of MKS Source Integrity Enterprise Edition, it will be possible for developers to check in/out project files to their virtual server even if the network connection is down. Again, when the connection is re-established, the two servers will automatically resynchronize in a process that is completely transparent to the user and not dependent on an administrator.

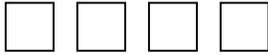
## **Benefits of the Federated Server Architecture**

### **Lower Total Cost of Ownership and Lower Administrative Burden**

The most attractive benefits of MKS's Federated Server Architecture are its lower total cost of ownership and lower administrative burden compared to the replication model. In a replication scenario where teams at different sites are working on the same stream, objects in the repository must be assigned ownership. An object is owned by a specific server and can be modified only through that server. Managing that ownership is a burden in and of itself. The ownership model in turn makes it cumbersome for users at different sites to collaborate because they must work on separate branches owned by their respective servers. Until a synchronization occurs, users at the other site have no visibility to recent local updates. After a synchronization occurs, a merge process is required to bring the work from the branches together. A second synchronization is then required to bring the merged results back to the other site. Further, if both sites want to continue working on the object, either an additional merge or an additional branch creation is required.

In the FSA model, none of this is required because users at both sites work against the same branch in the same repository. Currently, the replication model requires specially trained (usually full time) administrators to facilitate the synchronization process on a regular basis. With FSA, the software does all the work and requires modest administration aside from the normal duties that an SCM or configuration manager would perform. In an enterprise environment, with numerous distributed development teams, the savings that result from decreased administrative complexity multiplies as more teams are added on.

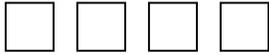
### **Developers and Managers Receive Real-Time Project Information**



In a multi-site environment communication between sites is critical because successful concurrent software development depends on knowing what all developers are doing all the time. Developers must know if another developer has locked a certain file to make edits and managers must be aware of how his resources are being utilized. Enterprise software development demands this level of project visibility and communication because deadlines are becoming shorter and achievement of on-time goals depends largely on the agility of the development team.

Under the replication model, a software development team does not receive real-time project information because they must wait for the synchronization process to occur before everyone is made aware of how the project has changed since the last synchronization. Important decisions by managers can be delayed and developers run the risk of duplicating effort if they are unaware that another developer worked on the same file. The end result is potential confusion, delays in the project, missed deadlines and, ultimately, higher costs.

The Federated Server Architecture gives project managers and developers agility through providing real-time project information, regardless of their location. Managers can make decisions based on the most up to date information and developers can move on to other responsibilities if a desired file is already locked by another developer. In this environment, greater levels of efficiency are achieved, which means less time is wasted and more money is saved.



MKS Inc. and design, MKS Code Integrity, MKS Engineer Integrity, MKS Impact Integrity, MKS Integrity Manager, MKS Source Integrity, MKS Toolkit, CodeRover, Discover, Implementer, NuTCRACKER, SDM and Software Manager are trademarks or registered trademarks of MKS Inc. All other trademarks acknowledged. © 2002. All rights reserved.

IS0302WP