A Rapidsoft Systems' White Paper

© June, 2009

Managing Offshore Development Teams

Summary:

Outsourcing software can save you money, help reach market early and compensate for skills that you don't have. It can result in excellent results if done properly and using a professional software development company with which you have cultural match. In this white paper, we present you some helpful tips to manage remote offshore software engineering teams.

© All rights reserved. Rapidsoft Systems, Inc.

Introduction

The effect of Globalization touched all industries and jobs, software development is no different. Now a days software developers working for a company are physically separated with each other and they may be in different time zones within in a country or anywhere in the world. Companies often outsource software development projects to countries like India, China, Russia, etc.. And, in many instances, they also started using open source or freelancing developer teams for all or some of their software development efforts.

The main reason companies adopting globalization of software development is to save money and increase programmer productivity. However if the global software teams and development processes are not managed appropriately, it may ultimately cost more money for the organizations due to issues like poor software quality, missing deadline, etc. In this article, we will explore global software team management issues that arise due to improper use of tools, technologies and without coherent software development process to connect all the virtual team members to participate in the success of the project.

1. Global Software Development Issues

Outsourcing brings several benefits to companies but if not managed properly creates several problems and ultimately the outsource initiative fails to bring the benefits to the organization. Though there are

several problems occur in managing virtual software development teams, in this article we will explore only the issues that arise due to improper use of tools, technologies and without having a coherent software development process to connect the virtual teams to participate in the overall software life cycle management.

Complexity of enterprise software applications has been growing steadily due to rapid technology changes and adoption of those new technologies in developing business applications. Recently IDC conducted a survey in which it found more than 60% of the companies told that the current software code is much more complex than the code written two years back. Growing business applications complexity combined with software teams working from different locations manifest numerous problems in the software development cycle.

Following are two main issues that occurs in global software development:

- Inconsistent development environment
- Open Source & Licensing

1.1. Inconsistent Software environment

Inconsistent software development environment is a big issue in global software development, here environment includes IDE, source code repository, database tools, application server, build and deploy tools, testing tools, etc. Every developer working in the global team must have same version and release of software environment. This is easier said than done, with the advent of open source, a developer can download an IDE and start using it, which might be different from the rest of the other team members. Even if the IDE is same version, open source IDEs like Eclipse has numerous plugins available for the developers to download and use it in their IDE. The plugin used by one developer may not be usable by other developers or it may be of different version. Often times these plugins create "marker code" when checked into the source code control and if retrieved by other developers may corrupt their IDEs. Individual software team members dealing with these types of issues will significantly reduce their productivity and their by increase the overall project budget.

1.2. Open Source & Licensing

Open source community has given great products to software developers and it has been widely used in enterprise software development. But if one developer or a development team using a open source that is not compatible with all other teams, then it creates various issues in the global software development. The Eclipse plugin issue described above is just an example, software developers can use various open source tools, frameworks, libraries that are available over a mouse click away to download and incorporate it into their source code. Since the source is available for the open source software, developers can even modify to fix a bug or change it to work with their own development will cause significant integration related issues. Though the individual source code might have worked during the developer's unit test, may not work when integrated with all other developers code since they might be using the same open source software without any changes to it. Of course the main purpose of the integration test is to find how the distinct software components are working together with all other components in a controlled environment, but issues arising due to mismatched open source software or individual software developers modifying the open source is unwarranted and would significantly increase the software development cost.

Another critical issue that comes with the usage of open source is its licensing. Open source software comes with various categories of licenses. For example, some open source software licenses allow developers to use it in their development environment but it can not be packaged and shipped to end user. So either a developer or a software team using an unauthorized open source will create legal issues and undermine the global software development.

2. Global Software Development Success Factors

Unauthorized use of open source software can be managed among the global software developers by having a consistence set of development environment that includes all the tools necessary for the developers to be productive in their daily work. Following are some of the key factors companies must plan before considering outsourcing their software development to either to their onshore or offshore vendors.

- Increase individual developer Productivity
- Enhance software quality standards
- Utilize value of Global Development Teams

All of the above aspects are required in any software development project developed either in single location or in multiple locations. However, how companies understands and addresses these issues among the global software development teams is the key for the project success or failure.

3.1. Increase individual developer Productivity

One of the major goals in any software development is to increase the individual developer productivity, global software development is no different. However, the global teams bring in new challenges to the companies that are not seen in teams working in a single location. Following are some of the factors that directly affect programmer's productivity.

- Knowledge of source code
- Source code reuse
- Source code complexity

3.1.1. Knowledge of source code

Developer's knowledge of the source code is the single biggest factor in boosting their productivity. Here knowledge of source code means not just understanding the logical flow of the program but to understand the following:

- How it will work when integrated with other components
- Performance of the code during unit and integration test
- Knowledge of business domain
- Knowledge of API, libraries, build and deploy processes

Individual developer's source code knowledge increases the team's collaborative knowledge and ultimately all the teams working in the project.

3.1.2. Source code reuse

Obviously code reuse means less new code to write and test, so it directly improves the developer productivity. For example using open source increases code reuse, even within a organization there will be several components that can be reused. But the code reuse eluded many software development projects. In many well managed software projects, code reuse is failed to increase the productivity of the developers.

3.1.3. Source code complexity

Source code complexity is directly related to programmer productivity. Generally, offshore vendors get several application maintenance projects more than new software development work. In these type of projects offshore software developers need to understand and maintain the code that are developed by some one else. Several of these projects go to offshore locations without proper documentation and they must understand the code both at the macro and micro levels. If the source code complexity metrics are bad then the offshore developer has to spend more time in understanding the code.

3.2. Enhance software quality standards

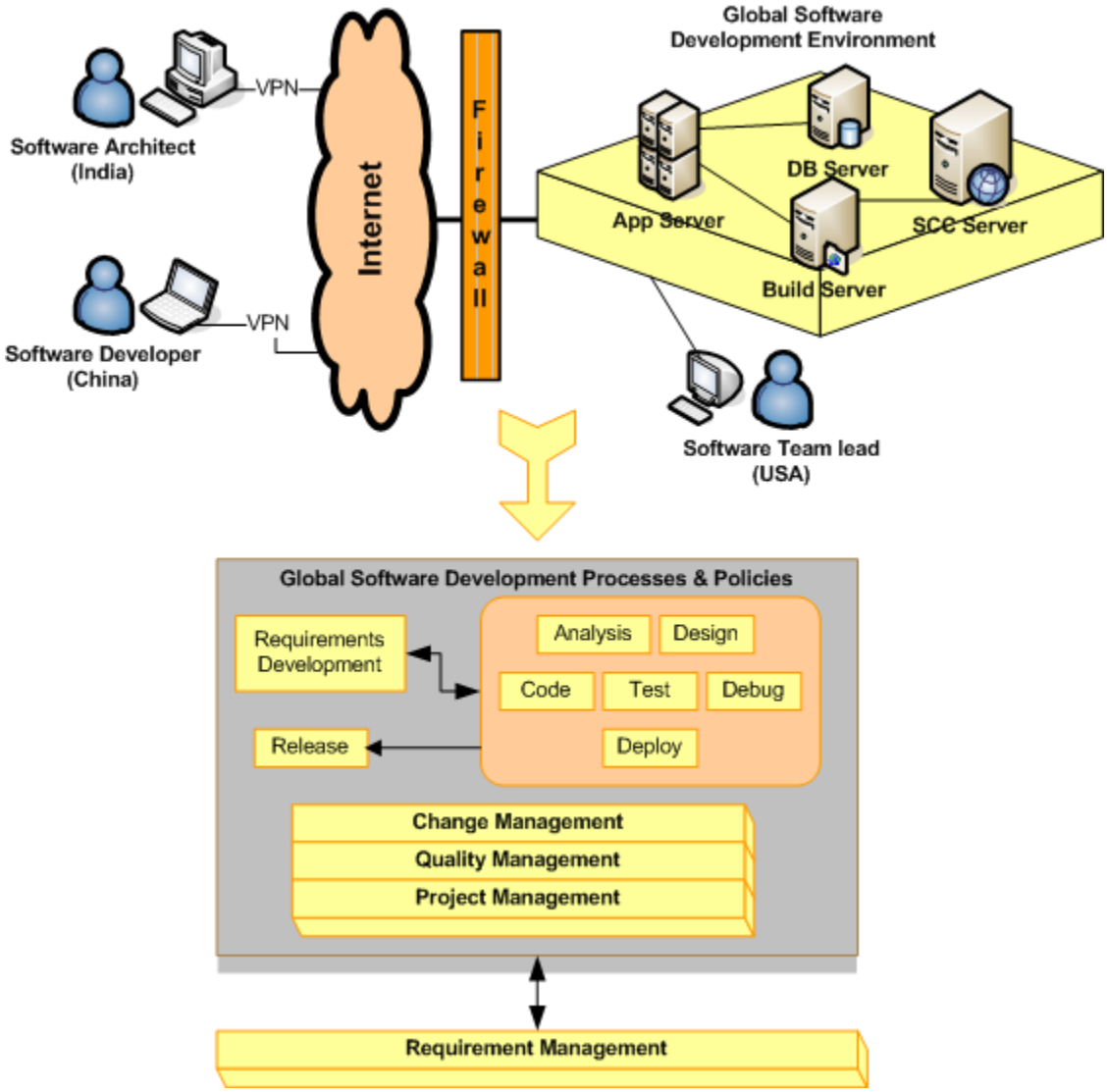
Software quality directly impacts the organization's business value, poor quality increases the cost and reduces the business value of the organization to its customers. Many Indian offshore vendors have six sigma and other quality certifications on software processes. But hiring a quality certified vendor for the software development alone cannot guarantee higher quality in the software delivered by the offshore teams.

3.3. Utilize value of Global Development Teams

The main reason organizations use global software teams from offshore vendors or from their own captive centers is to reduce cost, increase time to market, use global skills, etc. But if individual team's productivity is low, then organizations cannot achieve any of the above goals. The challenge is to create global distributed software environment so that any developer from any location can understand, develop, unit test the code with minimal efforts their by providing maximum value to the entire project. When organizations works with offshore vendors, it is common for the offshore vendor to move around software professionals between projects based on their customer's needs. So the distributed software platform must have features for individual developers to communicate and collaborate problems with the rest of the team members. Any software environment must have these capabilities for the developers to be productive, but this is critical in software teams working in different timezones. Imagine one team check in the source code in the evening and goes home, and a different team from a different timezone uses and finds regression issues, how will they fix and continue their work?

4. Global Software Development Environment

Based on the issues and features discussed organizations must have a development environment to utilize the global software resources to its full potential. Organizations must either buy, build, open source, or the combination of all the three to create a development environment that addresses all the issues discussed earlier.



The diagram shows software teams working in India, China, and US, the development environment is physically located in US. The global development environment shows a simplistic view of different servers connected together, other servers might exist in large projects. Each developer has an IDE that is connected to the development environment located in US that gives seamless global software development process and policies to the entire project team.

Following are some of the tools required to succeed in managing virtual software teams.

4.1. UML Modeling Tool

UML helps software professionals to visualize and understand the flow of software code in graphical notation without seeing the actual source code. Various elements of UML like class diagram, sequence diagram, state diagrams, etc gives developers to understand the static and dynamic aspects of the software code. Using this tool a developer can forward engineer UML models into source or reverse engineer the code into UML models. This is a single most important feature that helps the global

developers to understand and communicate the software code they are working on with other team members.

4.2. IDE Design Tools

Typical enterprise software projects use many different technologies like web services (SOAP, REST), web 2.0 components, etc. Some IDEs provide model driven development for some of the technologies. For example if you're using JBuilder, then it has design tools for EJB3.0, JPA, etc which helps you to design your components and the JBuilder generates source code for you to either modify or integrate with other components. If your organization decided to use the auto source code generation feature then everyone in the global team must use this feature. If the policy is not enforced on all the team members then code becomes complex and unmanageable. No IDE can provide all the features required for all the team members like software business analysts, architects, designers, developers, testers, etc. Some component's source can be completely auto-generated with no modifications, some needs some modifications, and the rest must be developed by the developers. Organizations must perform technical due diligence and come up with a strategy to identify features that needs to be auto-generated vs hand-coded and they must enforce the policy among all the team members.

4.3. Collaboration Tool

Developers working in global teams often communicate with each other to perform tasks like share source code snippet, conduct code review, perform test cases, design diagrams, share exception stack traces, etc, on real-time. They may also remotely debug to solve a specific issue. Organizations must have tools to support this features to achieve the full benefits of global team development.

4.4. Project Management Tool

Generally in a typical global software projects, the core technical team will be responsible for assigning high-level tasks to team leads, and team leads create and assign individual tasks to their team members. Organizations must have tools to create, assign and manage tasks to the global team leads and to individual developers. Often, requirement, responsibilities, and priorities changes in a project, core team responsible for managing the project must have capabilities to reassign and move the tasks among the team members to achieve the project milestones. At any given time, it should be possible for the project management to get the real-time status of the project's progress.

4.5. Code Analyzer & Profiling Tool

Static code analyzer tool helps to find issues that arises in distributed teams, multiple developers working in multiple locations write code that may functionally be right but may not confirm to the coding standard or it may have other quality and maintenance related issues like poor performance, unused code, un-maintainable code etc. Though these issues arise in developers working in single location it is more prevalent in global software teams. Static code analyzer and code coverage tools identify these issues and developers can fix it before checking the code into the source code repository. Similarly, profiling tool identifies dynamic nature of the software code and reports to the developers to identify and fix the issues.

Any large-scale software projects will have all the above mentioned tools and may have several other tools specific to their project requirements. Most of these tools are stand-alone products and works well in projects where all the team members are located in one place and they all work at the same timezones. Nevertheless, it fails to achieve the goals of distributed teams that work in different locations

and in different timezones due to various reasons mentioned in this article. The key challenge for the organizations in a global software projects is to integrate all the tools with the IDE and create a global software development environment as shown in the figure for every developer to design, code, test, debug, build, and deploy the code to development or test servers.

Conclusions

Organizations are increasingly using global software teams located in different countries and they face with the challenges of communicating, coordinating with the developers and maintaining high programmer productivity among the team members. As described in this article organizations should create and use a global software development environment so that all the project members like project managers, business analysts, architects, DBAs, developers, testers, and infrastructure team members can fully collaborate and perform their daily tasks seamlessly. Without the integrated global development environment, organizations cannot achieve their goals from their virtual software teams.

By following the above common sense approach to software outsourcing, you can truly benefit from the lower cost of offshore outsourcing. The main points are using only professionally run companies that demonstrate a level of professionalism and are willing to provide access to their engineering teams. Besides, your ability to openly communicate can make or break a project therefore having a local project manager for your project with whom you can deal with on daily or weekly basis is very important.

We hope the above article helps if you are looking to outsource any software development. And, if you would like to talk to us - you can visit us at http://www.rapidsoftsystems.com/. We promise to give you no obligation help whether you use us or not for your next software project.

For more information and specific questions, please contact us at:

Rapidsoft Systems, Inc,
Mailing Address: 7 Diamond Court, Princeton Junction,
New Jersey 08550, USA

(Princeton) New Jersey, (San Jose) California, Delhi/ Gurgaon (India), Mumbai (India), Chennai (India)

Web: www.rapidsoftsystems.com

Phones: **1-609-439 4775 / 1-609-439-9060 (US East Coast, NJ Office)**
1-408-829-6284/ 1-408-890-2509 (US West Coast, San Jose Office)
Fax: 1-831-855-9743

Email: info@rapidsoftsystems.com