**Collaboration and communication**

In order for a team to succeed to a high standard, members must be able to effectively collaborate. Communication is a key element to this success. It allows members to share their own opinions and knowledge and keep others up to date, this can help to prevent many common project issues such as over lapping work and misconceptions leading to a poor quality product.

Alistair Cockburn on of the creators of the agile approach believes the most effective method of communication is face to face where a physical medium can be used to illustrate ideas. [1]

<http://agilemodeling.com/essays/communication.htm> [1]

Discussion

Holding meetings is an effective way to have team members communicate amongst one another and also with clients. Holding these meetings periodically is a good way to ensure that work is being completed at a reasonable rate, and also allows an opportunity for members to ask for second opinions. In these meetings objects such as whiteboards, charts and paper should be used to increase the effectiveness of communication. [1] Research on communication in agile systems development reflected by 155 studies showed that teams reported communication was more efficient due to physical proximity. [2]

<http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1253&context=bise> [2]

Using discussion tools such as Facebook and Skype with have inbuilt group chat and video call features are especially important for teams to communicate. Using these can help to overcome time zone differences and allows instant communication without the need to physically meet. [2] While these tools can be effective it is important not to use too many simultaneously.s

Collaborative tools such as google drive, wikis and shared codebases are extremely helpful in project work. It allows others to see real time updates live and also prevents the occurrences of overlapping in work. Using these tools has been proven to reduce the need for constant communication in agile development teams. [2]

Seeking feedback from clients and customers is an important part of software engineering and it relies on good communication. Without performing this the product may be developed in a manner that in unsatisfactory to the client.

Retrospectives after significant milestones is good way in which a group can reflect on their current methods and determine areas in which they performed well and which could be done better. By completing retrospective activities such as a project feedback form project groups can learn from their mistakes and discover ways to work more efficiently. [5]

<http://www.peachpit.com/articles/article.aspx?p=1801991&seqNum=4> [5]

**Managing change**

Making changes in a project is an essential and vital part of progressing and overcoming obstacles. It is typical for unforeseen problems such as a change of costs, client’s needs and technological issues to occur during a project. Other reasons for changes could be upgrading technology or personnel to improve efficiency, or external social, political or environmental pressures. Regardless of the size of a change it must be managed effectively.

Before making a change it is important to first analyse the issue then devise a plan to solve these issues. When changes have been made it is important to communicate the changes to involved stakeholders and to promote acceptance of the change. On-going development is necessary to ensure the benefits are seen. Documenting changes is important so they can be referred to in the future.

Identifying need for change:

The first practice involved in change management is the understanding of the need for a change. This involves analysing the current issues and then identifying solutions to the issues. These solutions must be appropriately analysed to determine the implications the change would have on stakeholders. According to these implications a risk management plan will also be created. Before progressing assessment will be made on whether the change should progress or not.

Developing a change plan:

Developing a change plan can begin following analysis. This involves identifying the milestones and resources required to implement the change. The plan will detail the responsibilities of each team member involved and identify the tasks they are required to complete, in what timeframe, and a rationale of why and how it can be achieved.

Communicating changes:

Communicating the change to stakeholders is important as they are the ones directly affected by the changes. Communicating the need and benefits for a change helps improves stakeholders understanding and can lead to less resistance to change. Stakeholders need to have the ability to contribute their own opinions and ideas as it has been shown that taking an open-mind approach leads to better implementation. Keeping stakeholders informed and engaged promotes adoption for the change.

Enforcing progression of change:

Once changes have been implemented it is important they are enforced by making sure each member is fulfilling their responsibilities according to the change plan. In this step progress is tracked and the risk management plan is utilised when necessary.

Documenting the change:

Documenting the details of the change is important as they may be required in the future. Making a change could cause errors and affect the functionality of software and might not be identified for a long period of time, being able to review exactly what was changed can help solve these issues. Furthermore it is possible to forget a rationale for making a change. Having change documentation can help improve efficiency on future tasks by reflecting on changes that were successful and which were not.

**Quality Assurance:**

Quality assurance is set of tasks which ensure the quality of software engineering processes which lead to an increase in the quality of the actual product.

Standards and rules:

In order to ensure that work is being completed in a correct manner to do a certain standard some rules should be put in place.

Reviews:

Throughout a development process it is important to include reviews at certain stages to ensure that work is being completed according to the original plan. Keeping track of progress is important as certain areas may be overlooked.

Testing:

One of the most important practices involved in the assurance of quality is testing. Testing is required to determine the functionality of all features of developed software. There are many forms of testing, each which with aim to discover defects and malfunctions of code. Some of these are not easily discoverable and therefore a method of automated software testing should be used to increase the chances of their discovery.

Change management:

It is important to follow a well formed procedure when making changes to ensure that they are necessary and will actually provide a solution before they are made. Changes are always necessary in a project, therefore managing they efficiently will help to improve the over all quality.

Continual learning:

As members of a team may be forced to work with software or concepts they are not familiar with it is important to ensure that they develop their knowledge in those areas. By allocating time to acquiring knowledge, the skills of team members can be developed and this is likely to improve the quality of the product.

Risk management plan:

There should be a plan in place which can be followed when unforeseen events occur. The idea of this plan is to reduce any impact on the progression of the project. Solutions to multiple scenarios must be put in place early to mitigate any issues. Examples of this would be computer failure, team member illness and security breaches .

https://www.mitre.org/publications/systems-engineering-guide/acquisition-systems-engineering/risk-management/risk-management-approach-and-plan

highered.mheducation.com/sites/dl/free/0073375977/673802/chapter16.doc

<http://softwaretestingfundamentals.com/software-quality-assurance/>

http://softwaretestingfundamentals.com/sqa-vs-sqc/