Agile methodologies vs. traditional lifecycle

Agile methodology and traditional lifecycle refers to the way in which software is developed. However, agile development develops software in a way that is different from the traditional method. Agilephilosophyallows frequent inspection and adaptation of the project while the traditional methodology is a sequential method that splits the project into parts that are supposed to be fulfilled.

However, it lacks adaptability and flexibility in ensuring the requirements of the project are fulfilled (Baker 2006, pp. 34).

In traditional methodologies when a glitch occurs and plans are made, such as changing the software, nears impossibility which means that the software needs to go to the beginning with the development of a new code. This happens as long as there is no further glitch in the development process.

On the other hand, agile methodology has a low risk level when developing the software. This means that it emphasizes the values and principles rather than traditional method of processes. Hence, agile methodology supports working in cycles and at the end of each cycle the priorities of the project are re-evaluated to check whether it conforms to the requirements.

In most cases the Traditional lifecycle and the agile methodologies allows cutting down the total software or picture into puzzle size bits such as coding, designing and testing.

However, when it comes to specific methodology in understanding the breaking down of the project, there are some variations that are evident. In the traditional lifecycle, when a stage is completed it remains like that

because it is hard to manipulate according to time and user needs (Clammer 2007, pp. 56).

This means that the process should start from designing a completely new system. Agile methodology is flexible and allows for change at the end of each stage depending on new ideas that may arise. It enables changes to the project without the entire project been rewritten. Hence, such approach reduces overhead costs and provides a flexible way in which upgrade of programs can be commissioned.

In the case of agile methodology, the project can be launched at the end of each tested stage. This means that it is an opportunity that ensures that bugs are traced and eliminated at the development level and it is further double tested to ensure that the first bug is eliminated.

However, on the view of the traditional methodology, this capability is not provided, but the project is tested at the very end of it. It means that if bugs are found the entire program needs to be re-written (Eberle 2006, pp. 90 - 91).

Another point is the customer satisfaction and object oriented designers and programmers. The modular nature of agile ensures that the right people are employed for the stage for timely release even if it does not match with the entire customer specifications.

While, on the traditional methodologies it supports one main release and any problems such as delays or fulfillment of the customers specifications results into highly dissatisfied customers (Melton 2007, pp. 70).

Both methodologies allows for departmentalization administration. The traditional methodology allows departmentalization at each stage while in the case of agile methodology the coding module of each stage can be delegated to separate players.

Hence, allows many parts or stages to be fulfilled at the same time. However, the level of departmentalization differs; in the case of agile it is more pronounced than in the case of traditional methodology (Eberle 2006, pp. 94).

The two software methodologies have diverse means in the ways that are supposed to fulfill the requirements of software development. Scalability, adaptability and flexibility in addition to customer satisfaction are the main features that set these two methodologies apart.

Bibliography

Baker, F. 2006, Traditional Software Development: Waterfall, McGraw Hill, New York.

Clammer, L. 2007, Software Methodologies: An Introduction, Jakarta, Prentice Hall of Jakarta.

Eberle, J. 2006, Introduction to Software Development, New York Publishers, New York.

Hawthorne, F. 2005, Software Development Methodologies, Oxford University Press, London.

Melton, Z. 2007, Extreme Programming: Agile Software Development, Cambridge University Press, Singapore.