Web Portal Development using Extreme Programming Practices

Pooja Kolte
MCA, PGDCS, SNDT
University Juhu Tara Road,
Santacuz(w),Mumbai-49

Trupti Bhujbale
MCA, PGDCS, SNDT
University Juhu Tara Road,
Santacuz(w),Mumbai-49

Anita Chaware
MCA, PGDCS, SNDT
University Juhu Tara Road,
Santacuz(w),Mumbai-49

ABSTRACT
The following are three main reasons why developing software is difficult. These are: Developing complex software of good quality is extremely tough, developing software is not getting any easier especially in Internet time, and there is an increasing shortage of skilled individuals who can do the work.[1] In face of these pressures, the amount of software development effort required must be optimized. In this paper, we propose an iterative software life cycle using extreme programming practices to resolve software development issues in the smooth manner. The proposed approach speed up process with less effort and produce a more maintainable code for future maintenance and evolution. In this paper we tried to extend the Extreme programming architecture with some definitions for each phase for easy use. Using the enhanced architecture of XP we also did a case study for developing the web portal for student counselling called E-counselling. The same project was also given to another team using traditional waterfall model to compare the results like time duration, cost etc.

Keywords
Agile, continuous integration, pair programming, refactoring, releases, Spike, Unit test, validation test, waterfall model, XP.

1. INTRODUCTION
Extreme Programming (XP) is a software development methodology. It is intended to improve software quality and responsiveness for new requirements of customer. It is a type of agile software development, because of that it advocates frequent “releases” in short development cycles, which is used to improve productivity and introduce checkpoints with the help of that checkpoints new customer requirements can be adopted. The method takes its name from the idea that the beneficial elements of traditional software engineering practices are taken to "extreme" levels.[9] Extreme Programming is a discipline of software development which is based on values of Simplicity, Communication, Feedback, Respect and Courage. It works with the whole team together in the presence of simple practices. Enough feedback is useful to the team to see where they are. Extreme programming is successful because it stresses customer satisfaction or customer wants.
Extreme Programming emphasizes on teamwork. Managers, customers, and developers are all equal partners in a collaborative team. Extreme Programmers constantly communicate with their customers and fellow programmers. The team keeps their design simple and clean. They get feedback by testing their software starting on day one. They deliver the system to the customers as early as possible and implement changes as suggested by customer.

2. HISTORY
Extreme Programming was created by Kent Beck during his work on the Chrysler Comprehensive Compensation System (C3) payroll project. Beck became the C3 project leader in March 1996 and began to refine the development method used in the project. The first Extreme Programming project was started March 6, 1996. [5]
The "practice of test-first development, planning and writing tests before each micro-increment" was used as early as NASA's Project Mercury, in the early 1960s (Larman 2003). A NASA independent test group can write the test procedures, based on formal requirements and logical limits, before the software has been written and integrated with the hardware. In Extreme programming this concept is taken to extreme level by writing automated tests. Automated tests validate the operation of small sections of software coding, rather than only testing the larger features of code. [5]

3. WORKING WITH AGILE
3.1 Why Agile
Classical or Traditional methods of software development such as Waterfall model, Spiral model, etc have many disadvantages like as follows:

- Huge effort during the planning phase.
- Poor requirements conversion in a rapid changing environment.
- Treatment of staff as a factor of production

Agile Software Development Methodology is invented to overcome these problems. It is lightweight, people-based rather than plan-based.

3.2 What is Agile
Agile software development is a group of software development methods which is based on iterative and incremental development process. It promotes adaptive planning, evolutionary development and delivery of product; a time boxed iterative approach and encourages rapid and flexible response to any change.[6]
Several agile methods that have been developed areas follow: [10]

- Scrum
- Dynamic Systems Development Method (DSDM)
- Crystal Methods
- Feature Driven Development

- Lean Development
- Extreme Programming (XP)

3.3 Why Extreme Programming (XP)

There are some reasons [7] for why Extreme Programming (XP) is needed:

- It produce high quality software
- It keep programmers happy
- It keeps Customer happy
- It will be prepared for change
- It speed-up the development

3.4 What is Extreme Programming (XP)

Extreme Programming (XP) is a software engineering methodology, the most popular among several agile software development methodologies. XP prescribes a set of day-to-day practices for managers and developers. Extreme programming is based on five values that are:

- **Simplicity**- XP keeps things as simple as possible while meeting the requirements of the project. Relation between communication and simplicity is, simplicity in design and coding should improve the quality of communication.
- **Communication**- Constant and thorough communication between members of the development team as well as customer is needed to success.
- **Feedback**-For the customer satisfaction, customer involvement and feedback for product is required.
- **Respect**- The respect value includes respect for others as well as self-respect. Members respect their own work by always going for high quality and seeking for the best design for the solution at hand through refactoring.
- **Courage**- Developers must have the courage and confidence to bring change and produce quality results for customer satisfaction.

There are four basic activities that XP proposes for software development process:

- **Coding**- In XP coding is considered the only important product of the system development process. XP programmers start to generate codes at the very beginning.
- **Testing**-XP prefers to always check if a function works properly or not by testing it. XP uses automated unit testing.
- **Listening**-Listening is very important in XP. For XP developers the ability and expertise in technical aspects should be accompanied by the ability to be good listeners. This ability is useful for them to understand what customers want and to develop the solutions which match customer's needs.
- **Designing**-Without proper design in the long run system becomes too complex and project could come to a halt. It is then important to create a design structure that organizes the logic in the system so too many dependencies in the system can be avoided.

4. ADVANTAGES AND DISADVANTAGES OF XP

4.1 Advantages

- Pair programming:
  XP team works with the pair of developer that means there are always two developer called pair in XP to develop the code. One can do the coding and other can instruct him and verify that code whether it is correct or not. This code development is done only on one machine by that a particular pair. Rotation is performing between pair to take knowledge of work of coding.
- Robustness:
  XP breaks the task into modules. All that modules can combined together to get end product. Because of the coding for modules there are very less bugs occurred in code. Validation testing determines successful completion of code, implementation testing checks whether the customer requirements implemented properly or not and regular testing reduces the chance of bugs in the coding at development stage.
- Resilience:
  Traditional approach work well when there are static requirements but in actual requirements keep changing, but XP allows the change in requirements of customer.
- Coding is important:
  XP gives importance to the coding rather than unnecessary paperwork and meetings. It works on small piece of code for simplicity in code and allows changes as per the customer requirements.
- Employee satisfaction:
  Extreme programming is a value-driven approach sets fixed work time for employee with less overtime work. Small piece of code and customer feedback helps to work before tight deadline of project, improving the employee satisfaction.
- Cost saving:
  In traditional approach, changes made at the end of project life cycle. In XP changes are allowed during development stage which reduces the cost, because cost will increases as the project moves ahead in different phase of life cycle.
- Test driven development:
  XP gives importance to coding as well as testing. It is called test driven developments which XP checks whether the code or function works properly or not. The testing is done with the help of unit test which means the test of small piece of code with automated test tools.
- Lesser risks:
  XP divides task into small piece of code which reduces the dependencies of architect, project manager and coder which also reduces the chances of risk.

4.2 Disadvantages

- Extreme programming is hard to do:
  Extreme programming is hard to do because it has lots of disciplines and practices which are sometimes not accepted by the team. There is need of customer involvement during development and most of the times customer don't give their valuable time.
- XP is not structured:
  XP is not structured, because of that it is difficult to find defects to the tester by just looking towards screen but in traditional approach the documentation helps to find significant number of defects in code.
- Refactoring can be a waste of time:

XP can modify the code to perform change in the code. But too much of refactoring is not necessary sometimes and its wastes a time.

- Duplication of code:
  XP has a pair programming that means they works in pair for the development of code which creates duplicate data in code as well as in database and because of that it takes too much of time for code testing.
- Sometimes XP team does not believe on fixed price, fixed scope:
  There is a fixed scope and fixed price in XP, sometimes XP team does not believe on that concept because traditional approach and other programming works with detailed planning

5. ARCHITECTURE OF XP

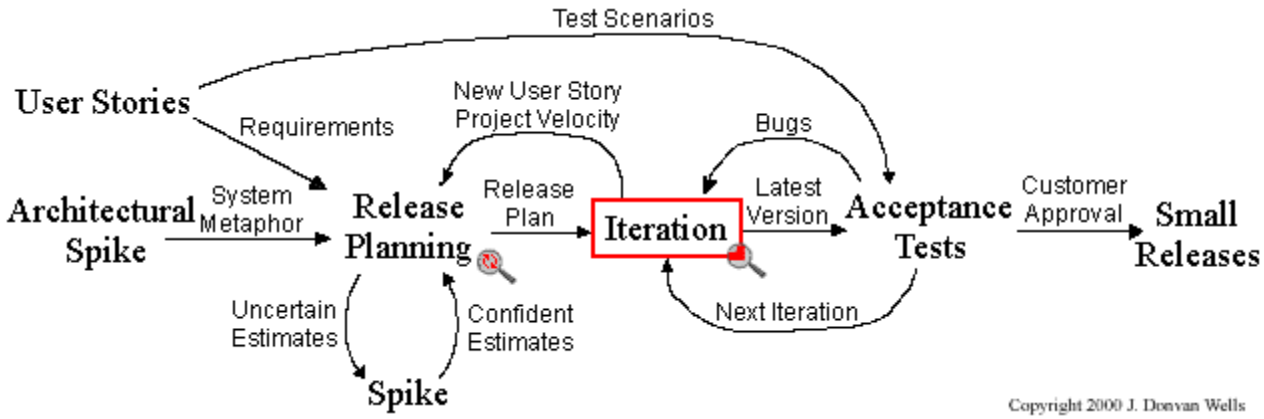

Fig 1: Extreme programming project 2000 Don Wells.

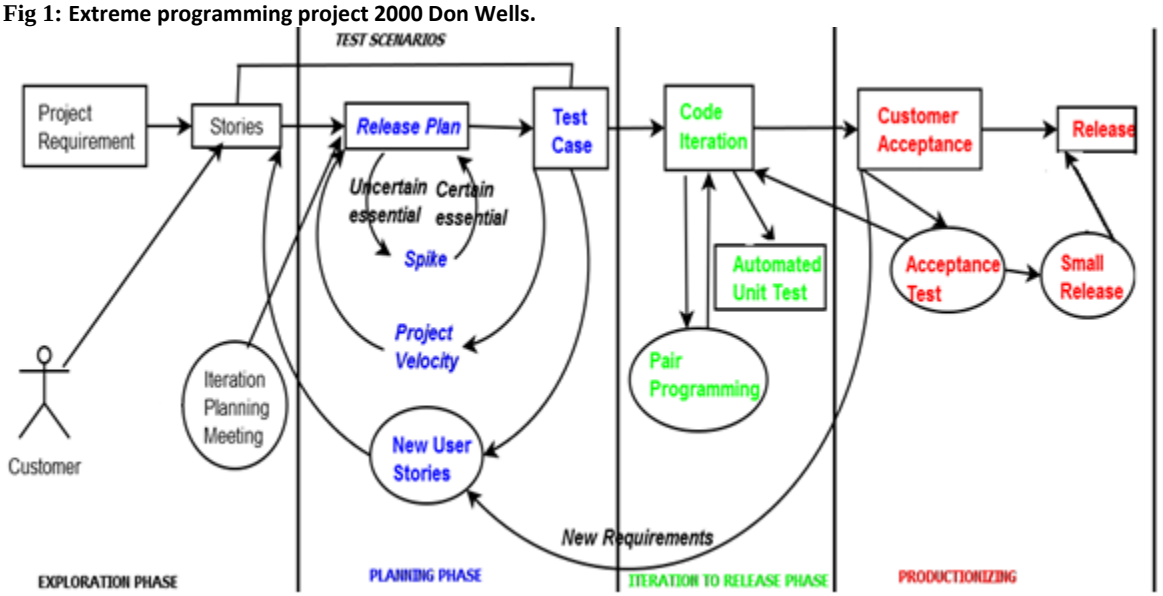

Fig 2: Extended Architecture of Extreme programming with defined processes.

We have tried to model diagram in figure 1 in to four phases as shown in figure 2. Those are exploration phase, planning phase, iteration to release phase, production phase which are further implemented using seven steps as project requirement, user stories, release plan, test case, code iteration, Customer Acceptance, Release.

As per the Theory of extreme programming the following are actor's involved in extreme programming.[4]

- Customer: customer/client is one person who is involved in all phases. During exploration phase customer is involved for writing user stories, again for creating release plan during planning phase. While coding customer is supposed to be present if required by developer, and for small release he is required to pass acceptance test.
- Developers are responsible for planning and iteration to release test.

- Project leader is responsible to conduct daily standup meeting and keeping people moving around.
- Project manager is responsible for creating release plan and solving managerial issues. He is also responsible for creating commitment with customer.

5.1 Exploration phase

During exploration phase project requirements are gathered. Customer is supposed to write user stories. As "customer is always available" is one of rule in extreme programming, Customer is available for iteration planning and release planning meetings. The planning process within extreme programming is called the Planning Game. Planning for per iteration is done once in a week which is done for creating to plans that is iteration plan. Iteration planning meeting is conducted. The plan contains guideline for developers and assigns the tasks of the developers[4]. There is no customer involvement. Iteration Planning is carried in three steps:

- Exploration : With help of class collaboration cards requirements taken from the clients in the form of user stories, are converted into tasks
- Commitment: These tasks are then assigned to a pair of programmer and time required is estimated with help of project velocity.
- Steering: The tasks which are carried out are then compared with the original user story at the end of each iteration.

The purpose of the Exploration phase is to guide the product into delivery.

5.2 Planning phase

All complex issues are broken down to simple design and uncertain and essentials issues are spiked to certain one. User stories are used to write down unit test cases. In this phase release Plans are created. These are helpful for determining what requirements are included for which near-term releases, and when they should be delivered. The customers and developers are both responsible for conducting this meeting. The customer will provide important requirements for the system. These will be written down on user story cards. Developers will commit themselves to the functionality that will be included and the date of the next release. In the steering phase the plan can be changed. If customer acceptance is failed then new test cases are written and if still problem persist, new user stories can be created.

5.3 Iteration to release phase

At developer's site, customer is available while implementations, as functionality requirements from user stories are left off. Project velocity is number of user stories finished per iteration, which is useful for planning next iteration. [4] The programmer gets the task card for one of the tasks. The programme will implement the task with one more partner; such kind of programming is done in practices of Pair programming. The programmers will design the functionality of the task. Programmers start writing automated tests which is called as Unit Testing before coding the functionality. The test is executed after they have coded the task. Refactoring involves removing code which is the symptoms of weaker design.

5.4 Productionizing

User stories finished per iteration need to accept by customer during production phase where user will get working, tested, implementable software component. XP supports continuous process of software development. Productionizing phase involves following steps.

- Continuous integration[4]
  It involves integrating often and using collective ownership[3].Each team player likes to work with the latest version. Changes should not be made to obsolete code causing integration headaches. Set up a dedicated integration computer. Whole team will be responsible for making changes in design or in code.
- Small releases
  The delivery of the software is done via frequent releases of live functionality creating concrete value. Small releases also increase customer confidence about development of the project. Also it is possible for customer to provide new suggestion after working with software in real life.

6. ECONOMICS OF SOFTWARE DEVELOPMENT [2]

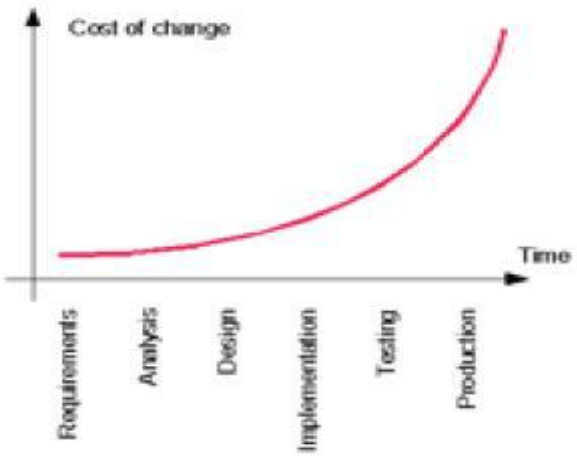

Fig 3: Cost of change for traditional SDLC model [2]

As shown in above diagram cost of fixing bugs is goes on increasing as software devolvement process continues to next phase, but, in Extreme Programming the cost of change curve is seems to be flat. The customer is always available hence the feedback loop is reduced to extreme level. Coding on unit test also helps for serving the purpose. Here the feedback loop is effectively reduced to minutes instead of days, weeks, or even months which is typical in traditional processes.

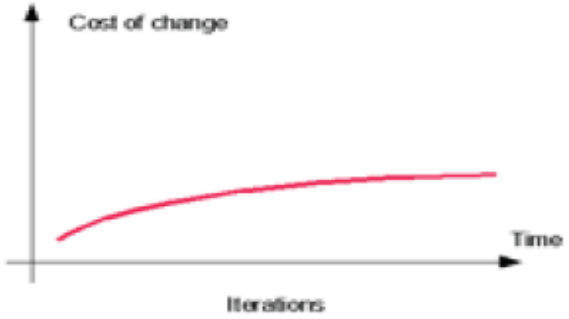

Fig 4: Cost of change for Extreme programming model.[2]

According to Sergey Konovalov and Stefan Misslinger in their Extreme programming paper[2]above graphs are used to show the economic growth for development of software. As a result the chances are good, that the cost of change will not get out of hand. Hence using extreme programming for project development will always save the cost so that later

changes in requirements will not affect the cost for time per iteration.

7. THE EXPERIMENT

The experiment is to implement extreme programming for developing a web portal is given hereafter including students previous exposure to XP, student background, duration of the experiment, E-Counselling project scope, group formation, tools and XP mentor activities.

7.1 Student's background

The experiment was conducted on class-level with 6 students as a developer, 2 as a project leader from senior class, which are involved in this experiment. Student background related to the experiment had subjects such as ASP.Net, DBMS, WT and Software engineering. The developer team had no previous exposure to team, hence first two weak training given to them.

7.2 Duration of the experiment

The experiment was conducted within a time frame of a single semester. Within software engineering course, students need to present assignments for the project. To complete the experiments students worked for 12 weeks.

7.3 E-Counseling project scope

This project was simple but work is done on real problem, because this project is only for the semester exam. The project work as done on simple problems such as the counselling for academic students, admission process, and college names list. To understand concept of software engineering this project has been taken. This project scope was limited to providing the following basic services to students: Students related services like Student registration, student login and registration to take admission, select college list and Admin related services like Admin login, add colleges, update college data, delete college data, and allot colleges to the registered student after filling form.

7.4 Group formation

A team of 6 students participating in the experiment, therefore those 6 students was distributed in 3 groups with the pair of 2 students in each group. Among the 3 groups tasks are allotted to each group per iteration. "2 by 6" Pair programming practice is implemented. Project Head is responsible for allocation of task and rotation of the pairs.

7.5 Tools

To develop a project of E-Counselling the tools were used that: ASP.Net and SQL server.

7.6 Results

This section includes the results with following points: partial adoption of XP, on-site customer, planning game (writing user stories and iteration planning meeting).

7.7 Role of extreme programming

In this experiment, only the XP practices pertinent to small scale projects were focused on "sub-practice".

The sub-practices included those contributing to rapid feedback and learning process namely, planning game, pair programming, collective code ownership, unit testing, simple design, refactoring and use of coding standards.

- On-Site Customer

  Due to real-world constraints, there was no real customer. Students had three hours per week contact with the simulated customer and 1-2 hours with the XP mentor. To enhance communication between developer and customer, one web site was established to post user stories and suggested project releases and deadlines.

- Planning Game

  The E-Counselling requirements were discussed with the XP group. Because of the time boxing students agreed for 2 releases. Release 1 included 6 stories such as designing the home page of portal, Student Registration form, student login, form filling, Admin login and College allotment, this work was done by first group. Release 2 included 4 stories such as select college, add college data, update college data and delete college data, this work was done by second group. It should be mentioned that the planning game practices was implemented with full success with XP group. There was also a project head which monitors all the groups.

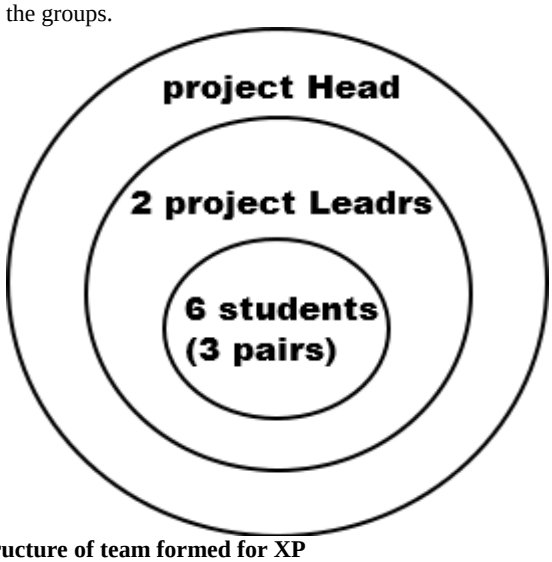

Fig 5: Structure of team formed for XP

Above figure shows that there were total 6 students those were the developers for E-Counselling project. Those 6 students were from junior class. Other 2 students from senior class studying software engineering were the project leaders. The project leader supports the developers and check whether the project is on right track or not. Validation testing, implementation testing and regular testing in extreme programming were done on regular time intervals. There was one project head also which monitors all the students that mean developers as well as project leaders.

8. CONCLUSION

During this case study for extreme programming we can conclude that XP is faster development model than traditional development model. Extreme programming also reduces the cost for maintenance when compared with the traditional waterfall model for the same project which was done by the other team at the same time duration. It also supports test driven approach which help us to deploy the web portal just after the completion whereas for waterfall model testing was done after coding which has taken more time as customer where having some issues after coding . In XP customer is always bound to development process hence it is best practice in which developer can achieve maximum customer satisfaction. The use of pair programming improves quality and speed of development. XP forces to all team members to be expertise in all technology. Hence if customer agrees to be bound for whole development process XP is best practice to implement for project. The only drawback which we had seen during the whole project is proper training for working in agile environment with new technology should be given to the team members.

9. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards development of the model for extreme programming and agile development models. Our special thanks to project head, and development team students for contributing their work.

10. REFERENCES

[1] Booch, G. (2001, March). Developing the Future. Communications of the ACM. Retrieved July 4, 2001, from the World Wide Web: http://portal.acm.org.

[2] Sergey Konovalov and Stefan Misslinger Extreme Programming May 23, 2006

[3] Don Wells Extreme programming.[online] http://www.extremeprogramming.org/map/project.html

[4] K.Beck,Extereme Programming Explained : Embrace change, Addison Wesley Longman, Reading, Mass, 2000

[5] Ll.Williums et al. , " strengthening the case for pair programming ," IEEE software ,Vol. 17, no. 4, July – Aug 2000

[6] Tobias Bergemann Agile software development [online]http://en.wikipedia.org/w/index.php?title=Agile_software_development&oldid=515608970

[7] Michael Kircher Siemens AG, Corporate Technology, eXtreme Programming in Open-Source and Distributed Environments

[8] Enterprise Software Blog http://enterpriseblog.net/a

[9] "Extreme Programming", Computerworld <online>,Computerworld-appdev-92

[10] Extreme Programming Wilfrid Hutagalung <online>,http://www.umsl.edu/~sauterv/analysis/f06Papers/Hutagalung/

[11] Ronald E. Jeffries[online] http://xprogramming.com