MANAGEMENT OF TIME UNCERTAINTY IN AGILE

Rashmi Popli and Priyanka Malhotra and Naresh Chauhan

Assistant Professor,Department of Computer Engineering, YMCAUST FARIDABAD

ABSTRACT

Agile software development represents a major departure from traditional methods of software engineering. It had huge impact on how software is developed worldwide. Agile software development solutions are targeted at enhancing work at project level. But it may encounter some uncertainties in its working. One of the key measures of the resilience of a project is its ability to reach completion, on time and on budget, regardless of the turbulent and uncertain environment it may operate within. Uncertainty of time is the problem which can lead to other uncertainties too. In uncertainty of time the main issue is that the how much delay will be caused by the uncertain environment and if the project manager comes to know about this delay before, then he can ask for that extra time from customer. So this paper tries to know about that extra time and calculate it.

KEYWORDS

Agile Software, Slack, Optimistic time, Pessimistic time, Probability of Delay

1. INTRODUCTION

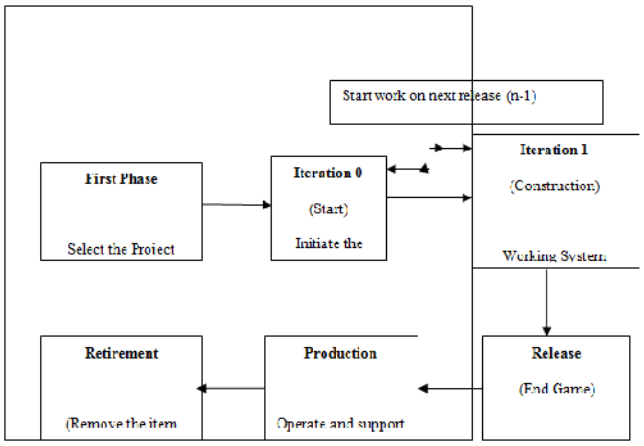
In the last few years Agile methodologies appeared as a reaction to traditional software development methodologies. Agile methodologies for software development take a novel, lightweight approach to most aspects of designing and producing applications. Agile software development is an iterative development method. Its basic concept is people – centred and it acknowledges that requirements can change. A key characteristic of any agile approach is its explicit focus on time estimation and business value for the clients. The goal of time estimation is typically to develop potentially shippable product. The accurate estimations of time is critical for both developer and customer. Ignorance of estimation methods may cause serious effects like exceeding the budget, poor quality and not right product. The key factor which is causing the problem in estimation is time uncertainty, so there is a need of some mechanism for minimizing uncertainty of time.

In Section II the life cycle of agile is described. Section III describes what uncertainty is. In Section IV related work in this field is discussed, section V proposes scenario calculation of slack time and uncertainty in time in agile. In Section VI shows evaluation and results of proposed algorithm, section VII concludes the paper.

2. AGILE LIFE CYCLE

Agile SDLC contains the six phases, pre-project planning, start, construction, release, production and retirement. These phases are described as in detail:

DOI : 10.5121/ijsea.2014.5408



Agile Life Cycle

2.1 Pre-Project Planning:

This phase is the first phase of ASD life cycle. The various activities performed in this phase are:

- **Define the goal of project**: First the goals of the project and market aspects are defined. It explores how the new functionality improves the organization's presence in the market, how it will impact profit of organization. This phase also helps in identifying the potential stakeholders and their goals.
- **Select best strategy for the project**: When the strategy is chosen for the project several issues are considered, like the present team is capable to handle the project or has to increase the size of team. Whether there is need of relocate the team or not and which software life cycle paradigm - traditional/waterfall, iterative, or agile – will be good for the project.
- **Feasibility Analysis**: During this phase feasibility study of the project is done. In feasibility study only four issues are needed to consider economic feasibility, technical feasibility, operational feasibility, and political feasibility [40]. Feasibility analysis efforts should also produce a list of potential risks and key milestone points during the project.

All the activities of the pre-project planning should be as agile as possible because in this phase collaboration with stakeholders is necessary who are knowledgeable enough and motivated enough to consider the potential project and invest in just enough effort to decide whether to consider funding the effort further.

2.2 Iteration 0: Project Initiation

The second phase or the first week of an agile project is referred to as Iteration 0 or project initiation. The various activities performed during this phase are:

- **Garnering initial support and funding for the project**: In this phase the project output or final product, its cost and approximated time is estimated. At this level it should be clear that what is being produced, how much it will cost and how much time it will take to complete.

- **Requirement gathering**: At this stage full participation of the stakeholders is needed to gather and model the requirements. At this stage the requirement gathering and modeling is done. The main goal is to understand the problem and solution domain. Not much time is wasted to create documentation. The details of these requirements are modeled on a just in time basis in model storming sessions during the development cycles.
- **Starting to build the team according to project**: Whenever the development of project starts, the team building also start in parallel. Key team members are identified. At this point there should have at least one or two senior developers, the project coach/manager, and one or more stakeholder representatives in the team.
- **Building an initial architecture for the system:** At the starting of the project there should be at least a general idea of how the system is going to build and the project is based on which application. The developers of the project discuss and decide a potential architecture for the system. This initial architecture evolves over time. The goal is to identify an architectural strategy, not about to write a huge amount of documentation.

2.3 Construction Iterations

During construction iterations, high-quality working software is delivered incrementally, which meets the changing needs of the user or customer. This can be done by the following steps:

- **Communication between customers and the developers**: Communication between the customer and developer is necessary for reducing risk by using rapid feedback cycles and via closer collaboration.
- **Implementing functionality in a priority order**: In ASD change in the requirements is allowed to meet the exact needs of the customers. The stakeholders are given complete control over the scope, budget, and schedule – they get what they want and spend as much money as required and for as long as they're willing to do so.
- **Analyzing and designing:** Every individual requirement is analyzed before the implementation of that requirement. A test-driven design (TDD) approach is selected for development. The individual testing is performed for every developed requirement.
- **Ensuring quality**: Quality of the product is ensured by selecting the best design and testing the code time to time.
- **Continuous delivery of the working software**: At the end of each development cycle or iteration there must have a partial, working system to show customer. Pre-production testing can be done like system/ integration testing.
- **Confirmatory and investigative testing**: In agile process a significant amount of testing is required throughout construction. Confirmatory testing is the agile equivalent of "testing against the specification" because it confirms that the software which is going to be built will work according to the requirement of our stakeholders. Investigative testing is done by test professionals who are good at finding defects which the developers have missed. These defects might be due to integration problems or sometimes they may be because of requirements which are not considered or simply have not implemented yet.

2.4 Release Iterations: The "End Game"

During the release iteration phase, also known as the "end game", the system is transit into production. The various activities of this stage are:

- Final testing of the system: Final system testing and acceptance testing are performed at this point, although the majority of testing has been done during construction iterations. In this case beta testing can be performed for the system in the presence of end users.
- Rework: There is no value of testing the system if it is not planned to act on the defects that has been found. One cannot address all defects, but should expect to fix some of them.
- Finalization of the system and user documentation: Some documentation may have been written during construction iterations, the documentation is treated like any other requirement: it is also estimated, prioritized, and created only if stakeholders are willing to invest in it.
- Training: Training is provided to end users, operations staff, and support staff to work effectively with working system.
- Deploy the system: System is deployed after this cycle.

2.5 Production

The goal of the Production Phase is to keep systems useful and productive even after the product has been deployed to the user community. This process will differ from organization to organization and perhaps even from system to system, but the fundamental goal remains the same: keep the system running and help users to use it.

2.6 Retirement

The goal of the Retirement Phase is the removal of a system release from production, and occasionally even the complete system itself. This activity also known as system decommissioning or system sun-setting. Retirement of systems is a serious issue faced by many organizations today as legacy systems are removed and replaced by new systems.

3. UNCERTAINTY

Meaning of Agile is “moving. In releasing a particular plan or **user story**, it is needed to fix a set of release dates and then determine how much functionality can be achieved by those dates. Also this can be done by deciding the functionality first and then deriving the release date. In either case the functionality value is accessed against the cost and time to develop the system, In previous used methods cost and time both get neglected in assessing the functionality which lead to uncertainty in both time and cost. Also the size of the user story is not certain. These all factors leads to poor estimation in agile project and hence time uncertainty. In this paper there is an attempt to find solution to problem of these uncertainties and calculating the percentage of uncertainty.

4. RELATED WORK

Malik Hneif, Siew Hockow presented a review of Agile methodologies in software development. This review starts with a brief background about different approaches in software development. It includes difficulties in software development as development involves more critical and dynamic industrial projects and new difficulties emerged according to the growth of companies like evolving requirements, customer involvement, deadlines and miscommunications.

McDaid, D. Greer, F. Keenan, P.Prior, P. Taylor, G. Coleman[8] proposed a set of key practices which includes the practice, termed "Slack", of only signing up to for what the team is confident of achieving. Within this approach it is always possible to add more stories, time permitting, thus delivering more than was actually promised. This practice acknowledges that there is a significant amount of uncertainty in the estimated time to complete releases.

Rashmi Popli, Anita and Dr. Naresh Chauhan[5] proposed a common life cycle approach that is applicable for different kinds of teams. This approach describes a mapping function for mapping of traditional methods to agile method.

Siobhan Keaveney and Kieran Conboy[1], gave the study of the applicability of current estimation techniques to more agile development approaches by focusing on four case studies of agile method use across different organizations. The study revealed that estimation inaccuracy was a less frequent occurrence for these companies. The main estimation techniques used were expert knowledge and analogy to past projects also the Component of the process; fixed price budgets can prove beneficial for both developers and customers, and experience and past project data should be documented and used to aid the estimation of subsequent projects.

S.Bhalerao and Maya Ingle[4] presented the study of both traditional and agile estimation methods with equivalence of terms and differences. This study investigated some vital factors affecting the estimation of an agile project with scaling factor of low, medium and high. Also, an algorithm Constructive Agile Estimation Algorithm (*CAEA*) is proposed for incorporating vital factors.

Daniel D. Galorath[3] proposed a 10-step estimation process that begins by addressing the need for project metrics and the fundamental software estimation concepts. It shows how to build a viable project estimate, which includes the work involved in the actual generation of an estimate, including sizing the software, generating the actual software project estimate, and performing risk/uncertainty analysis.

Meso and Jain have compared ideas in agile development to those in Complex Adaptive Systems by providing a theoretical lens for understanding how agile development can be used in volatile business environments.

5. PROPOSED WORK

Client gives customer the requirements in form of user stories, and a backlog is created with those requirements. The time required for completion of project depends upon size of the user story. It needs to be certain about time taken by the project to be completed. The uncertainties in the timing is a big problem so project takes some marginal time called as slack that will compensate for extra time taken in the project work other then the optimal development time. However there is no formula for calculating this Slack. The proposed algorithm and formula suggests that how to calculate the slack time. For calculations, the time taken as hours rather than days because that will lead to actual result.

Effective time per day for Sprint Related work = Average work day time - Time allocated for other activities.

For example Average work day time = 10 hours
Time Allocated for other activities = 5 hours

Emails and Phone: 1 Hrs
Lunch: 1 Hrs
Meetings: 2 Hrs
Bug fixes: 1 hrs
Available time for Sprint Related work = 5 hours

5.1 Proposed Diagram

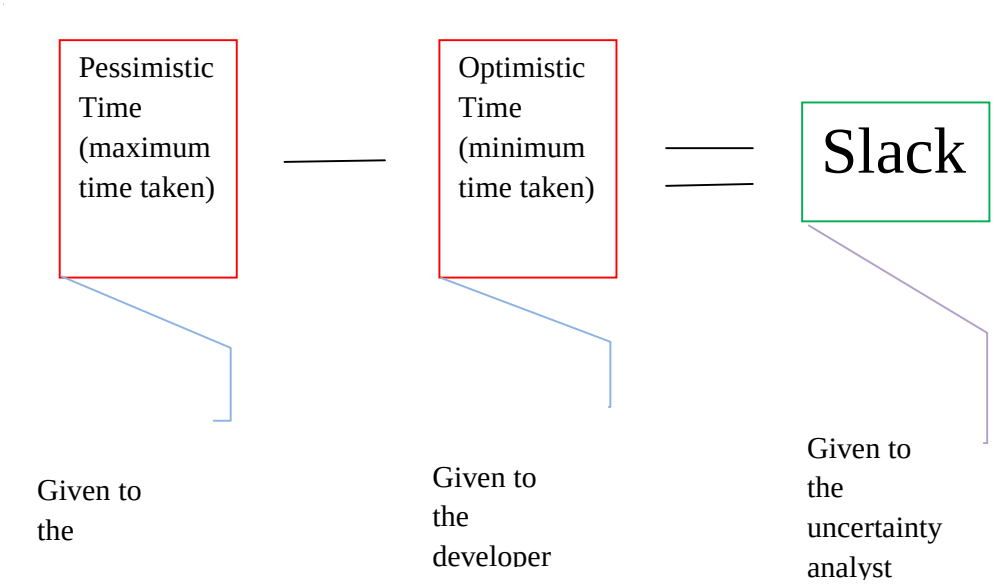


Figure 2: Showing what actually happening in this scenario of calculating slack

In the project there is a team of about 7 persons, one is project manager, four developers and two testers who are doing pair programming. For four developers total development time will be the ideal available time for development related work * 4.

The optimistic value of time period for one iteration is given to the developers so that they have to complete task in that period, and the pessimistic value is given to the customer or client. The calculation for pessimistic timing will be done by using probability of delay.

5.2 Proposed Formulas

The difference between these two timings will be helpful for exactly calculating the slack time.

1. ***Duration for developer*** = = $\sum_{i=1}^{n} Ti$ *[optimistic time] / work per time*
2. ***Duration for customer*** = = $\sum_{i=1}^{n} Ti$ *[pessimistic time] / work per time*
3. ***pessimistic time for each task*** *=(percent probability of delay *optimistic time)/100+optimistic time*
4. ***Slack time for a task*** *= duration for customer - duration for developer*
5. ***Time Uncertainty****= (slack/duration for developer)*100*
6. ***Total slack time****= (Total pessimistic time-Total optimistic time)/(Total working hours per*

*day*Number of developers)*

5.3 Proposed Algorithm

The proposed algorithm explains the various steps involved in removing the uncertainty in time in agile environment.

1. Identify the tasks of each user story based on the requirements, suppose for each user story there are n number of tasks so for each user story the tasks are t_1, t_2, t_3.....t_n.
2. Identify the optimistic time for each task and probability of delay for each task. Metric for Optimistic time value is hours and probability of delay will be in percentage.
3. Calculate the time (pes) for each user story by using the formula $\sum_{i=1}^{n} T_I$ [pessimistic time] where "i" denotes the tasks and pessimistic time for each task =(percent probability of delay *optimistic time)/100+optimistic time
4. Compute the overall Slack time for all the user stories using the formula Slack time = duration for developer - duration for customer

Duration for developer= $\sum_{i=1}^{n} Ti$[optimistic time] / Effective working hrs

Duration for customer = $\sum_{i=1}^{n} Ti$[pessimistic time] / Effective working hrs

5. Calculate the uncertainty percentage which is

= (slack time/duration for developer)*100

5.4 Proposed Activity Diagram

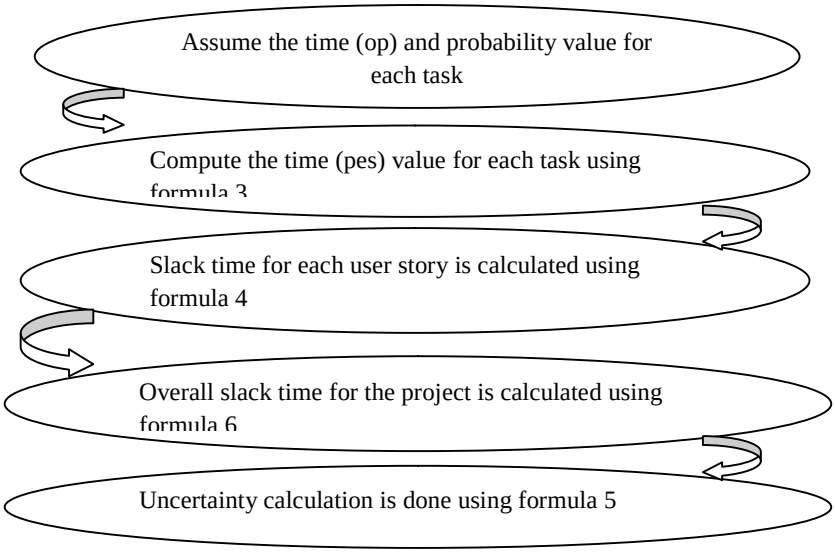


Figure 3: Steps involved in calculation of slack and uncertainty

6. EVALUATION AND RESULTS

In this section the feasibility of our algorithm is shown by calculating the estimated values of pessimistic time using probability of delay, value of slack and percentage of uncertainty for a project. We had considered the number of effective working hours as 5 per day .In this section the feasibility of our algorithm is shown by a case study in which the values are being calculated. We have considered the user stories of project which is **Letters of Credits**; the client is HP-client. A

graph can be drawn taking the optimistic time and pessimistic time values against each other, the bar graph and line graph both show the difference between both the values.

Table 1: Showing tasks of the project under taken and related values of optimistic time and probability of delay

N O.	DEVELOPMENT TASKS	OPTIMISTIC TIME	PROBABILITY OF DELAY
1	FSD REVIEW EXPORT OPENING	20	30
2	FSD REVIEW EXPORT REVIEW	10	20
3	FSD REVIEW OPENING CREATION	26	20
4	FSD IMPORT REVIEW CREATION	42	10
5	FSD EXPORT REVIEW CREATION	33	10
6	PH-2 REQUIREMENT STUDY	20	30
7	FSD EXPORT REVIEW CREATION	30	10
8	FSD IMPORT OPENING SIGN-OFF	45	20
9	FSD IMPORT REVIEW SIGN-OFF	50	10
10	FSD EXPORT OPENING SIGN-OFF	54	30
11	FSD EXPORT REVIEW SIGN-OFF	32	10
12	DEVELOPMENT REVIEW 1	34	20
13	DEVELOPMENT REVIEW 2	65	10
14	BACKUP ARCHIVE	30	20
15	PROJECT MONITORING	40	30
16	CONFIGURATION	23	20
17	UNIT TESTING	25	10
18	INTEGRATION TESTING	35	10
19	SYSTEM TESTING	25	20
20	TRAINING	20	20
21	PH1 UAT	5	30
22	PH1 UAT SIGN OFF	15	20

23	PH2 UAT SIGN OFF	20	10
24	PH2 DEVELOPMENT REVIEW1	20	20
25	PH2 DEVELOPMENT REVIEW 2	19	20
26	PH2 UAT	7	10

Table 2: showing user stories along with there corresponding associated tasks

NO.	USER STORY	ASSOCIATED TASKS
1	SRS	1-5
2	SRS REVIEW	6
3	DOCUMENTATION	6,7
4	FSD REVIEW IMPORT OPENING	8-10
5	NON-FUNCTIONAL DATA COLLECTION	20-23
6	PRE ENGAGEMENT SUPPORT	13-16
7	REWORK CODING	11,12
8	TESTING	17-19
9	CODE REVIEW	24,25
10	GO LIVE SUPPORT	21,26

Table 3: user stories with their optimistic time and calculated pessimistic time

USER STORY NO.	TOTAL OPTIMISTIC TIME	TOTAL PESSIMISTIC TIME
1	131	152
2	20	26
3	50	59
4	149	179.2
5	60	70.5
6	158	187.1
7	66	76
8	85	96
9	39	46.8
10	12	14.5

6.1 Numerical analysis

Total slack time= (Total pessimistic time-Total optimistic time)/Total working hours per day*Number of developers

Here total working hours per day are=5 hours
Number of developers=4
Total Slack time = (907.1-770)/20=6.85

Now the Percent uncertainty in the project= (Slack time/Pessimistic time)*100=
(6.85/45.35)*100=15.10%

6.2 Graphs

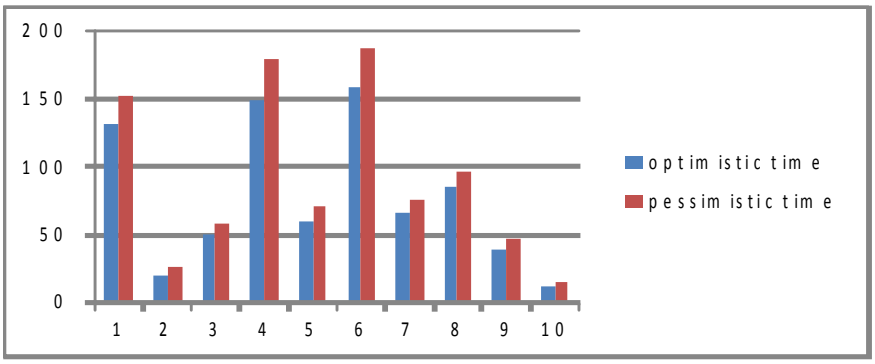


Figure 4: Bar graph representation of optimistic and pessimistic timing values

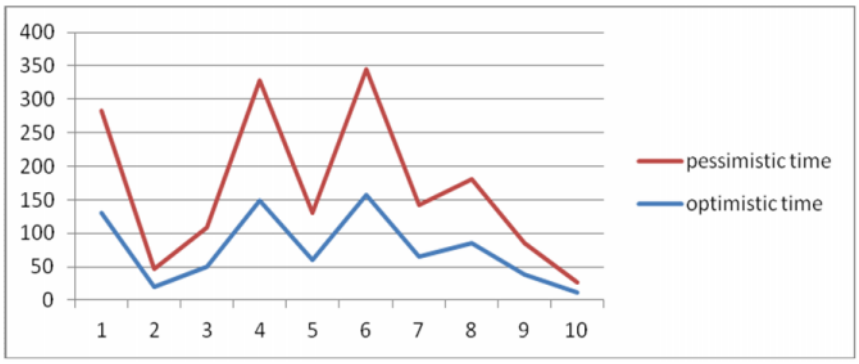


Figure 5: Difference between optimistic time and pessimistic time shown by lines

7. CONCLUSION

This uncertainty percentage tells us that by this percentage there are the chances of the project to get faulty. If this percentage value is much higher, then this may lead the developers think about removing the uncertainties first and then start the project. The slack value helps in improving the

project as we can remove some causes which are leading to the delay. Also the value of slack is important because now the project manager is sure about the completion time of the project and the relations with the customer can be improved as customer is well satisfied with the project completion at the given dead line. The approach used in the paper is very easy to understand and do not need any hard and fast calculations. After having exact values it would be easier for the manger to read out the reports because the work is shifted from the theoretical portion to the exact numerical data. Hence this paper is an approach to solve the problem of uncertainty of time.

REFERENCES

[1] Siobhan Keaveney and Kieran Conboy , "Cost Estimation and agile development projects" Product-Focused Software Process Improvement, 435-440

[2] Du, G., McElroy, J., &Ruhe, G. (2006). Ad hoc versus systematic planning of software releases–a three-staged experiment. Product-Focused Software Process Improvement, 435-440.

[3] Daniel D. Galorath, "The 10 Step Software Estimation Process For Successful Software Planning, Measurement and Control"galorth incorporated 2006.

[4] S. Bhalerao and Maya Ingle, "Incorporating Vital Factors In Agile Estimation Through Alogorithmic Method" International Journal of Computer Science and Applications, Ó2009 Technomathematics Research Foundation ,Vol. 6, No. 1, pp. 85 – 97

[5] Rashmi Popli, Anita and Naresh Chauhan. "mapping of traditional software development methods to agile methodology"

[6] Logue, K., &McDaid, K. (2008). Agile Release Planning: Dealing with Uncertainty in Development Time and Business Value. Engineering of Computer Based Systems, 2008. ECBS 2008. 15th Annual IEEE International Conference and Workshop on the (pp. 437-442). IEEE.

[7] McDaid, K., Greer, D., Keenan, F., Prior, P., Taylor, P., & Coleman, G. (2006). Managing Uncertainty in Agile Release Planning.Proc. 18th Int. Conference on Software Engineering and Knowledge Engineering (SEKE'06) (pp. 138-143).

[8] Logue, K., &McDaid, K. (2008). Agile Release Planning: Dealing with Uncertainty in Development Time and Business Value. Engineering of Computer Based Systems, 2008. ECBS 2008. 15th Annual IEEE International Conference and Workshop on the (pp. 437-442). IEEE.

[9] RashmiPopli, NareshChauhan," Research Challenges of Agile Estimation" Journal of Intelligent Computing and Applications" July- Dec 2012.

[10] RashmiPopli, NareshChauhan," "Scrum- An Agile Framework", International Journal of Information Technology and Knowledge Management (IJITKM) ISSN: 0973-4414", Vol-IV, Number-I, 20 Aug 2010.

Authors

RASHMI POPLI

Rashmi Popli is pursuing her Ph.D in Computer Engineering from YMCA University of Science & Technology, M.Tech(CE) from M.D University in year 2008,,B.Tech(IT) from M.D University in the year 2004.She has 9 years of experience in teaching. Presently she is working as an Assistant Professor in department of Computer Engineering in YMCA University of Science &Technology, Faridabad, Haryana, India. Her research areas include Software Engineering, Software Testing and Software Quality.

PRIYANKA MALHOTRA

Priyanka Malhotra is a research scholar pursuing her M.tech in computer engineering (Compter Networks) from YMCA University of Science &Technology, Faridabad, Haryana, India. Completed B.tech (Cse) from Kurukshetra Unviersity, Kurukshetra

DR. NARESH CHAUHAN

Naresh Chauhan received his Ph.D in Computer Engineering in 2008 from M.D University, M.Tech(IT) form GGSIT,Delhi in year 2004,B.Tech(CE) from NIT Kurukshetra in 1992.He has 22 years of experience in teaching as well as in industries like Bharat Electronics and Motorola India Pvt. Ltd. Presently he is working as a Chairman and Professor in the department of Computer Engineering ,YMCA University of Science and Technology, Faridabad, Haryana, India.. His research areas include Internet Technologies, Software Engineering, Software Testing and Real Time Systems.

.