CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | |

Internal Assesment Test - I

Sub:	Agile Technologies							Code:	**14SCS423**
Date:	28/03/2017	Duration:	90 mins	Max Marks:	50	Sem:	IV	Branch:	Mtech(CSE)

Answer Any FIVE FULL Questions

		Marks	OBE	
			CO	RBT
1	Explain how you can enter agility with respect to organizational, personal and technical success.	[10]	CO1	L1
2.	What is agile method? List the various agile methods. Can we develop our own agile method? How can you find a mentor for your agile development team?	[10]	CO1	L1
3	Explain the agile manifesto in detail.	[10]	CO1	L1
4	What are the responsibilities of Onsite Customer, product manager , domain expert and interaction designer in the XP team	[10]	CO2	L1
5	Explain any four pre-requisites for adopting XP in your organization.	[10]	CO2	L1
6	Discuss the issues involved in applying XP to existing project. .	[10]	CO2	L1
7	Explain how pairing is done in organization. Also explain Driving and Navigating with respect to Pair Programming.	[10]	CO4	L3
8	Explain the step by step procedure for conducting iteration retrospective.	[10]	CO4	L3

Course Outcomes		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1:	Describe the agile software development methodology.	0	0	0	0	0	0	0	0	0	0	2	0
CO2:	Describe the XP Lifecycle, XP Concepts, Adopting XP	0	0	0	0	0	0	0	1	0	0	1	0
CO3:	Explain the different aspects of Mastering agility.	0	0	0	0	0	1	0	1	0	0	0	0
CO4:	Apply the various practices of Thinking, Collaborating, Releasing, Planning and Development in organization.	2	0	0	0	3	3	0	2	0	0	0	0

Cognitive level	KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PO1 - *Engineering knowledge*; PO2 - *Problem analysis*; PO3 - *Design/development of solutions*; PO4 - *Conduct investigations of complex problems*; PO5 - *Modern tool usage*; PO6 - *The Engineer and society*; PO7- *Environment and sustainability*; PO8 – *Ethics*; PO9 - *Individual and team work*; PO10 - *Communication*; PO11 - *Project management and finance*; PO12 - *Life-long learning*

#132, AECS Layout, IT Park Road, Kundalahalli, Bangalore – 560 037
T:+9180 28524466 / 77

CMR INSTITUTE
OF TECHNOLOGY

Scheme and Solution – (IAT 1)

Department of Computer Science and Engineering

Agile Technologies (14SCS423)

Scheme

Q.1 Explanation on Personal Success (3 marks)

Explanation on Technical Success (3 marks)

Explanation on Personal Success (4 marks)

Q.2 Definition of Agile method(1 mark)

List of Agile Method (2 Mark)

Explanation on creating agile method (4 marks).

Explanation on finding mentor (3 marks)

Q.3 Agile manifesto values (2 marks)

Agile manifesto principles (8 marks)

Q.4 Short notes on On-site customer (3 marks)

Short notes on Product Manager (3 marks)

Short notes on Domain expert (2 marks)

Short notes on Interaction designer (2 marks)

Q.5 Explanation on any four prerequisites each carrying 2 and half marks

Q.6 Explanation on any five issues involved in applying XP to existing project each carrying 2 marks.

Q.7 what is Pairing (2 marks)

How pairing is done (4 marks)

Driving and navigating (4 marks)

Solution

1. **Explain how you can enter agility with respect to organizational, personal and technical success.**

Agile development focuses on achieving personal, technical, and organizational successes.

Organizational Success

- Agile methods also set expectations early in the project, so if your project won't be an organizational success, you'll find out early enough to cancel it before your organization has spent much money.
- Specifically, agile teams increase value by including business experts and by focusing development efforts on the core value that the project provides for the organization. Agile projects release their most valuable features first and release new versions frequently, which dramatically increase value.
- Agile teams decrease costs as well. They do this partly by technical excellence; the best agile projects generate only a few bugs per month. They also eliminate waste by cancelling bad projects early and replacing expensive development practices with simpler ones.

Technical Success

- XP programmers work together, which helps them keep track of the nitpicky details necessary for great work and ensures that at least two people review every piece of code.
- Programmers continuously integrate their code, which enables the team to release the software whenever it makes business sense.
- Extreme Programming includes advanced technical practices like TDD that lead to technical excellence.

- The whole team focuses on finishing each feature completely before starting the next, which prevents unexpected delays before release and allows the team to change direction at will.

Personal Success

- **Executives and senior management:** They will appreciate the team's focus on providing a solid return on investment and the software's longevity.
- **Users, stakeholders, domain experts, and product managers:** They will appreciate their ability to influence the direction of software development, the team's focus on delivering useful and valuable software, and increased delivery frequency.
- **Project and product managers:** They will appreciate their ability to change direction as business needs change, the team's ability to make and meet commitments, and improved stakeholder satisfaction.
- **Developers:** They will appreciate their improved quality of life resulting from increased technical quality, greater influence over estimates and schedules, and team autonomy.
- **Testers:** They will appreciate their integration as first-class members of the team, their ability to influence quality at all stages of the project, and more challenging, less repetitious work.

2. What is agile method? List the various agile methods. Can we develop our own agile method? How can you find a mentor for your agile development team?

A method, or process, is a way of working. Agile methods are processes that support the agile philosophy. Examples include Extreme Programming and Scrum. Agile methods consist of individual elements called practices. Practices include using version control, setting coding standards, and giving weekly demos to your stakeholders. Most of these practices have been around for years.

Don't Make Your Own Method.

- Creating a brand-new agile method is a bad idea if you've never used agile development before.
- Every project and situation is unique, of course, so it's a good idea to have an agile method that's customized to your situation.

- Rather than making an agile method from scratch, start with an existing, proven method and iteratively refine it. Apply it to your situation, note where it works and doesn't, make an educated guess about how to improve, and repeat.

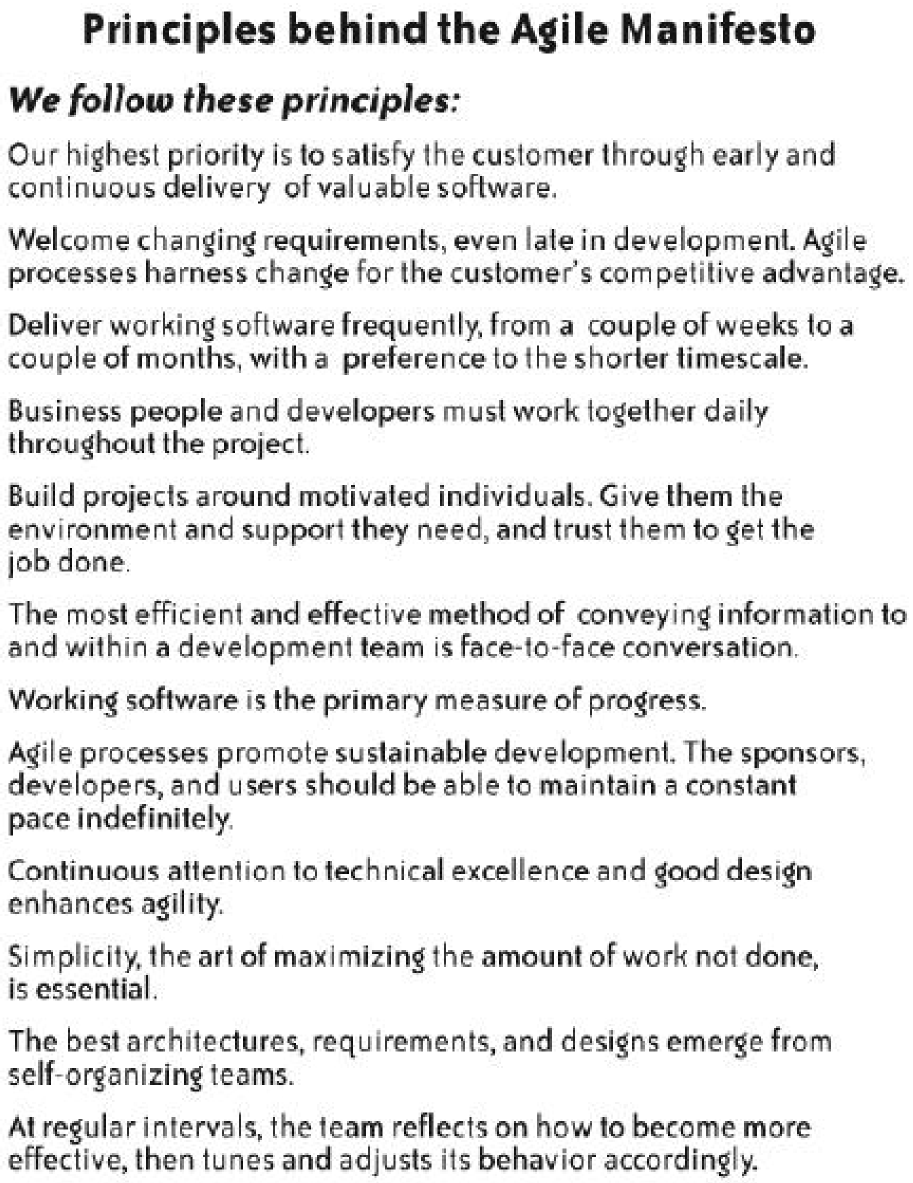
Find a Mentor

Mentor: an outside expert who has mastered the art of agile development.

The hardest part of finding a mentor is finding someone with enough experience in agile development. Sources to try include:

- Other groups practicing XP in your organization
- Other companies practicing XP in your area
- A local XP/Agile user group
- XP/Agile consultants
- The XP mailing list: extremeprogramming@yahoogroups.com

3. Explain the agile manifesto in detail.



Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity, the art of maximizing the amount of work not done, is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Values

Individuals and Interaction over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following the plan

4. What are the responsibilities of Onsite Customer, product manager, domain expert and interaction designer in the XP team

Onsite Customer

- Determine what is valuable for stakeholders. They interact with programmers and
    - ✓ Clarify project's vision
    - ✓ Identify the features and stories.
    - ✓ How to group the features into small, frequent releases
    - ✓ Manage risks
    - ✓ Creating release plan (release planning).
- They help in requirement gathering.
- They research a lot of information and gather requirements and provide it to programmers.
- Product managers, domain experts, interaction designers, and business analysts can play the role of the on-site customer.
- Difficult to find people qualified and willing to be on-site customers.
- 2: 3 ratio: As a rule of thumb, start with two on-site customers (including the product manager) for every three programmers.

Product Manager

- Should
    - ➢ Document the vision and communicate with stakeholders
    - ➢ Promote the product and its vision.
    - ➢ Handle organizational politics
    - ➢ Setting priorities for release
    - ➢ Review the work
    - ➢ providing direction for the team's on-site customers
    - ➢ involving real customers
- He has knowledge about the market status
- He convinces the stakeholders and decides what goes into the product and what stays out.
- He should participate in every retrospective, every iteration demo, and most release planning sessions.

Domain Expert

- Every industry has its own rules. These rules are domain rules, and knowledge of these rules is domain knowledge.
- Thus software which is developed by IT industry should implement these rules faithfully and correctly.
- Programmers have little knowledge about domain rules.
- The domain experts
- Have expert knowledge about the domain rules
- Domain experts spend most of their time with the team, figuring out the details of upcoming stories and standing ready to answer questions when programmers ask.
- Example: Financial Analyst, PhD Chemist

Interaction Designer

They define the UI of product.

- They understand the different type of users and their needs
- They understand how a user interacts with software.
- They perform tasks such as
    - Interviewing users
    - Reviewing the paper prototype of UI
- Many companies don't have interaction designer. This role is played by programmer or graphic designer.
- They contribute to release planning by advising the team on user needs and priorities.
- In each iteration, they help the team create mock-ups of UI elements for that iteration's stories.
- As each story approaches completion, they review the look and feel of the UI and confirm that it works as expected.

5. Explain any four pre-requisites for adopting XP in your organization.

Prerequisite #1: Management Support

It's very difficult to use XP in the face of opposition from management. Active support is best. You will need the following:

- A common workspace with pairing stations.

- Team members solely allocated to the XP.
- A product manager, on-site customers, and integrated.

You will often need management's help to get the previous three items. In addition, the more management provides the following things, the better

- Team authority over the entire development process, including builds, database schema, and version control.
- Compensation and review practices that is compatible with team-based effort.

If you want management to support your adoption of XP, they need to believe in its benefits. How will adopting XP help them achieve those successes? What are the risks of trying XP, how will you mitigate those risks, and what makes XP worth the risks? Talk in terms of your managers' ideas of success, not your own success.

Prerequisite #2: Team Agreement

If team members don't want to use XP, it's not likely to work. XP assumes good faith on the part of team members there is no way to force the process on somebody who is resisting it. It is never a good idea to force someone to practice XP against his will. In the best case, he will find some way to leave the team, quitting if necessary. In the worst case, he will remain on the team and silently sabotage your efforts.

If only one or two people refuse to use XP, and they are interested in working on another project, let them transfer so the rest of the team can use XP. If no such project is available, or if a significant portion of the team is against using XP, don't use it.

Prerequisite #3: A Colocated Team

XP relies on fast, high-bandwidth communication for many of its practices. In order to achieve that communication, your team members need to sit together in the same room.

Prerequisite #4: The Right Team Size
Teams with fewer than four programmers are less likely to have the intellectual diversity they need. They will also have trouble using pair programming, an important support mechanism in XP. Large teams face coordination challenges. Although experienced teams can handle those Challenges smoothly, a new XP team will struggle.

If you don't have even pairs: The easiest solution to this problem is to add or drop one programmer so you have even pairs.
If your team is larger than seven programmers: Consider hiring an experienced XP coach to lead the team through the transition. You may also benefit from hiring another experienced XP programmer to assist the coach in mentoring the team.

6. Discuss the issues involved in applying XP to existing project.

Big Decision

- More time is required to handle existing technical debt.

- ➢ In other words, you incur new technical debt in order to meet your deadlines.
- ➢ set aside extra slack to pay down the technical debt- to increase production and reduce the bugs.
- ➢ As your technical debt decreases, your velocity will rise again
- ➢ You can also rewrite the project from scratch or stop for several weeks to do nothing but pay down technical debt.
- ➢ It takes much longer than expected
- ➢ you lose feedback from stakeholders as well as the opportunity to take advantage of new business opportunities

Bring order to chaos

In order to eliminate the chaos, structure your project. Apply

- ✓ All the “Thinking” practices
- ✓ All the “Collaborating” practices
- ✓ All the “Planning” practices

Version control, collective code ownership, and “done done” and customer reviews. Take your existing project plan and convert each line item into a story card. once the team is used to working in iterations, the customers and the project manager should start revising the stories to make them more customer-centric.

Pay the technical debt

The biggest problem facing legacy projects is usually excessive technical debt
Reduce existing technical debt by introducing extra slack into your iterations.
At first, your clean-up efforts will seem fruitless, but over time, you’ll see greater and greater benefits to quality and productivity.
Thus you can steadily pay down technical debt while continuing to make progress on new stories

Organize your backlog

- Bug database is full of to-dos, questions, feature requests, and genuine defects.
- Customers and testers, go through the database and eliminate duplicates and unimportant issues.
- Bug database-handle in every iteration
- If your bug database is in use by stakeholders, support personnel, or other people outside the team, find a way to keep new entries clean.

Fix important bugs

- make a fix or don’t fix decision for each bug.
- Programmers-estimate cost of bug fixing.
- Close or defer all the bugs that you decide not to fix in this release. You can revisit them when you plan the next release.
- If you have a lot of bugs, consider spreading bug fixes throughout your plan.

Move testers forward

- Important workload for testers- manual regression testing
- The programmers’ focus on test-driven development will slowly create an automated regression suite and reduce the pressure on the testers
- use your iteration slack to automate the remaining manual regression tests.

Emerge from darkness

This process allows

- ➢ reduce technical debt
- ➢ increase code quality
- ➢ Remove defects.

- Increase productivity

Depending on the amount of technical debt you face, it could take many months to get to the ideal of nearly zero new bugs each month.

It will take months more to

- finish your regression test suite,
- eliminate the need for a separate pre-release testing phase
- integrate your testers

If you don't see progress within two months, there may be something wrong. Talk to your mentor for advice.

7. Explain how pairing is done in organization. Also explain Driving and Navigating with respect to Pair Programming.

- A good rule of thumb is to pair on anything that you need to maintain, which includes tests and the build script.
- Never assign partners: pairs are fluid, forming naturally and shifting throughout the day. Over time, pair with everyone on the team. This will improve team cohesion and spread design skills and knowledge throughout the team.
- When you need a fresh perspective, switch partners.
- It's a good idea to switch partners several times per day even if you don't feel stuck. This will help keep everyone informed and moving quickly.
- When you sit down to pair together, make sure you're physically comfortable. Position your chairs side by side, allowing for each other's personal space, and make sure the monitor is clearly visible.
- When a pair goes dark—talks less, lowers their voices, or doesn't switch off with other pairs—it's often a sign of technical difficulty.

Driving and Navigating

Navigator sees ideas and problems much more quickly. Navigator should think about

- What other tests do you need to write?
- How does this code fit into the rest of the system?
- Is there duplication you need to remove?
- Can the code be clearer? Can the overall design be better?

As navigator, help your driver be more productive. Think about what's going to happen next and be prepared with suggestions. Similarly, when a question arises, take a moment to look up the answer while the driver continues to work. Some teams keep spare laptops on hand for this purpose. If you need more than a few minutes, research the solution together. Sometimes the best way to do this is to split up, pursue parallel lines of inquiry, and come back together to share what you have learned. Spike solutions are a particularly powerful approach.

As you pair, switch roles frequently—at least every half hour, and possibly every few minutes.
If you're navigating and find yourself telling the driver which keys to press, ask for the keyboard. If you're driving and need a break, pass the keyboard off to your navigator.