Six Principles of Software Process Change

1.Major changes to the SW process must start at the top. -Senior management leadership is required to launch the change effort and to provide continuing resources and priority.

2.Ultimately, everyone must be involved. -a team effort

3.Effective change requires a goal and knowledge

of the current process. - you must know where you are.

effective change program requires a reasonable understanding of the current status; assessment; priorities

4.Change is continuous. -Software process improvement is not a one-shot effort; it involves continual learning and growth.

human intensive processes are never static

5.SW process changes will not be retained without conscious effort and periodic reinforcement. - rarely sustained for long without reinforcement. This means that new methods must be carefully introduced and periodically monitored, or they too will rapidly decay.

adoption of new process methods involves four stages

-installation

-learn to perform

-gradually improve

-ingrained

6.SW process improvement requires investment. - planning, dedicated people, management time, and capital investment

Some Common Misconceptions about the Software Process

• We must start with firm requirements. - widespread but fallacious view; the requirements must change; an incremental development process–that is, to gradually increase the level of detail

-Stable Requirement -gradually evolved; Requirements problems; properly resolved at the requirements level and not just patched in the design or the code

-Premature Requirements Freezes -incremental approach is that it takes too long; requirements, which is occasionally feasible, but not the general case.

• If it passes test, it must be OK. -Converse is certainly true; testing is a progressively less reliable indicator of system quality.

• Software quality cannot be measured. -No measure that comprehensively represents programming quality; Quality measures; software quality should use numbers, which requires measurement

• The problems are technical- the problems of software cost, schedule, and quality remain; the highest-priority problems of most software organizations are not technical

• We need better people- the true cause of errors is the process, not the people; Most of the problems can only be fixed through management action   
• Software management is different- traditional management methods can and should be used; many unique characteristics to software, they all require more management discipline, not less

Strategy for Implementing Software Process Change

• While the software people will generally welcome a concerted effort to improve their processes, the changes must be handled properly, or they will generate resistance.

• An effective change process has three phases:

1.Unfreezing 2. Moving 3. Refreezing

*unfreezing* is best initiated by an effort to understand the true problems in the organization and to make people aware of the opportunities for change. With a software process assessment

Champions, Sponsors, and Agents

Champions are the ones who initiate the change

process

senior management role is equally crucial: someone in authority needs to recognize the value of the work and sponsor

change agents who will lead change planning and implementation.

The Elements of Change

Three key elements of effective change are • Planning  
• Implementation  
• Communication

Key Practices of Agile Process

• Iterative development-Each iteration is a self-contained mini-project composed of activities; he goals for the end of an iteration is an iteration release, a stable, integrated and tested partially complete system.   
• Risk-driven and client-driven-Risk-driven iterative development chooses the riskiest, most difficult elements for the early iterations; Client-driven iterative development implies that the choice of features for the next iteration comes from the client – whatever they perceive as the highest business value to them

• Time-boxing- All the modern ID methods (including Scrum, XP, and so forth) either require or strongly advise timeboxing the iterations.

Fixed time for each iteration  
• Evolutionary development- Evolutionary iterative development implies that the requirements, plan, estimates, and solution evolve or are refined over the course of the iterations, rather than fully defined and “frozen” in a major up-front specification effort before the development iterations begin; Evolutionary methods are consistent with the pattern of unpredictable discovery and change in new product development.   
• Adaptive planning- Adaptive development implies that elements adapt in response to feedback from prior work; The intent is the same as evolutionary development, but the name suggests more strongly the feedback-response mechanism in evolution   
• Incremental delivery- Incremental delivery is the practice of repeatedly delivering a system into production (or the marketplace) in a series of expanding capabilities; Incremental deliveries are often between three and

twelve months.   
• Evolutionary delivery- Evolutionary delivery is a refinement of the practice of incremental delivery in which there is a vigorous attempt to capture feedback regarding the installed product and use this to guide the next delivery.

Agile Development

Agile development methods apply timeboxed iterative and evolutionary development, adaptive planning, promote evolutionary delivery, and include other values and practices that encourage agility

The Agile Manifesto

• Individuals and interactions over processes and tools

• Working software over comprehensive documentation

• Customer collaboration over contract negotiation

• Responding to change over following a plan

Principles behind the Agile Manifesto

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

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

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

4.Businesspeople and developers must work together daily

throughout the project.

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

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

7.Working software is the primary measure of progress.

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

9.Continuous attention to technical excellence and good design enhances agility.

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

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

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

evaluating their strengths and weaknesses

assessing other software organizations.

Steps for Software Process Improvement

Part 1: Software Process Maturity (Initial)

framework for software process management,

Part 2: The Repeatable Process

the actions needed to establish basic control over the software process,

orderly and continuous process improvement

Part 3: The Defined Process

specify the development process and the technical and managerial concepts,

more orderly reaction

Part 4: The Managed Process

quantitative software process control

quantitative quality and process management,

sustained process improvement

Part 5: The Optimizing Process

Preventing problems, use quantitative process methods

software process is the set of tools, methods, and practices

objectives of software process management

produce products according to plan

improving the organization’s capability

characteristics software process -predictable, cost estimates and schedule commitments must be

meet users’ functional and quality expectations

factors to consider in measuring

the process

-properly represent

-sufficiently well-defined and verified

-mere act of measuring human processes change them

address problems in the software process, need to treat task that can controlled, measured, and improved.

improvement of software development organizations follows six steps

1. Understand the current status
2. Develop a vision
3. Establish a list of required process improvement actions
4. Produce a plan
5. execute the plan
6. Start over at step 1

**CMM**

• Initial  
• Repeatable-stable process with a repeatable level of statistical control

• Defined  
• Managed-comprehensive process measurements and analysis.

This is when the most significant quality improvements begin   
• Optimizing

**CMMI**

• Initial

• Managed

• Defined

• Quantitively Managed

• Optimizing



**Chaotic Project:** inadequate project management,Without a plan destroys technical work.Chaotic project is seriously late,The schedule does not match actual,No plan even with a good technical shape–Lack of documented planning– A small amount of additional function expected turns out to be hundred thousands of code.– No idea how big the project really is – No basis to know where they stand– No way to justify added resources when they get into trouble– Even though the technical work would be sound, the customer was convinced the project would be a disaster. Why Software Organizations Are Chaotic: often make a guess rather than a plan,temptation to believe in magic,project follows an escalating cycle,new technical challenges put pressure on the process. Chaotic Forces:Unplanned Commitments:insufficient time and resources,hidden trap,Gurus,Magic:need effective management,Problems of Scale: software products become larger, they are much more difficult to understand and prototypes or multiple releases are needed because of those new complications.– The total function cannot be implemented in time.– Some needs cannot be understood without operational experience on a partial system.– Some design issues cannot be resolved until a preliminary system has been built and run.– A release discipline helps sort out user priorities.– No one successfully builds large software systems in one shot anyway.The Implications of Software Scale:affect the individual, the management system, and the technical methods and tools that we use,Software Process Entropy: many forces on the software process that push us toward disorganisation i.e.Dynamic requirements,Increasing system size ,Human nature.The Way Out: Plan the work.• Track and maintain the plan.• Divide the work into independent parts.• Precisely define the requirements for each part.• Rigorously control the relationships among the parts.• Treat software development as a learning process.• Recognize what you don’t know• When the gap between your knowledge and the task is severe, fix it before proceeding.• Manage, audit, and review the work to ensure it is done as planned.• Commit to your work and work to meet your commitments.• Refine the plan as your knowledge of the job improve

|  |
| --- |
| The Project Plan ● The Project Plan: The Work and how it’s done, benchmark to compare with actual performance. ● Project Planning Principles: quality program, precise of user needs, conceptual design, subsequent requirements, detailed design and implementation Strategy, implementation details, framework for negotiating the time and resources ● Planning Considerations: initial resource estimates unexpectable, early agreement on the essential functions, harmonious teamwork between the users and the software engineers ● The Planning Cycle: initial requirements, work Breakdown Structure (WBS), Product size estimates, Resource estimates, Project schedule ● Goals and Objectives: negotiation period, WBS: estimate of size of the product, Size Measures: Early estimates to actual, Function points from users’ perspective determines the number and complexity of inputs, summed to produce an overall function point rating, ● Estimating: Wideband Delphi Technique: - Group of experts given estimate forms, meet and anonymously complete the form, any method can be inaccurate, ● Estimating Inaccuracies: Normal statistical variations, estimating bias caused by the project stage, ● Estimating Contingencies: inflate the resource estimates, add to the schedule, and increase ● Productivity Factors: number of programmer months and time required, orderly procedure, – Organization productivity data, developing productivity data, Using productivity data, Resource estimating ● Developing Productivity Data: Identify a number of recently completed programs that are as similar, get data on the size, in LOC, count of the programmer months ● Using Productivity Data: derive base productivity numbers, ● Scheduling: total resource needs, organization’s historical experience, the overall project schedule objective, a staffing plan is developed, preliminary schedule, preliminary plan, ● Project Tracking: ability to determine project status, checkpoints, earned-value project scheduling, ● The Development Plan: submitted to management for review and approval, Project purpose and scope, Project goals and objectives, Organization and responsibilities, Management and technical controls, Work definition and flow, Development environment, Software development methodology, Configuration management, Verification and validation, Quality assurance provisions ● Planning Models: COCOMO, SLIM, error by 500 percent or more, used to augment, used effectively to check the estimates for errors or oversights, ● Prereq to a SW cost estimate the establishment of estimating group, rarely improve their estimating accuracy to this degree w/O the support of an orderly planning process |

Scrum

Scrum’s practices include:

– self-directed and self-organizing team

– no external addition of work to an iteration, once chosen

– daily stand-up meeting with special questions

– usually 30-calendar day iterations

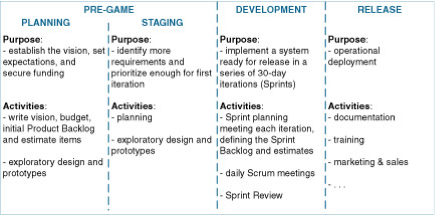
– demo to external stakeholders at end of each iteration

– each iteration, client-driven adaptive planning

The Scrum lifecycle is composed of four phases:

Pre-game  
1.Planning 2. Staging

3.Development 4. Release



Work products, Roles, and Practices

Roles: Chickens vs. Pigs

Chickens: involve, consult on the project and are informed of its progress.

Pigs: committed to the project and accountable for its outcome

Pre-game planning and staging - Sprint planning - Sprint – Self directed and self organizing teams - Scrum meeting - Don't add to iteration - Scrum master firewall - Decision in one hour - Blocks gone in one day - Chickens and pigs - Teams of seven - Common room (preferred) - Daily build - Sprint review

the Scrum Meeting – or scrum – is the heartbeat of Scrum and the project.

Each workday at the same time and place, hold a meeting with the team members standing in a circle, at which time the same special questions are answered by each member

• The Scrum Meeting provides a daily forum to update tasks, and surface and remove impediments.

• The meeting is ideally held in a stand-up circle to encourage brevity.

• On average, 15 or 20 minutes for 7–10 people. – Longer meetings are common near the start of an iteration.

• Non-team members (chickens) are outside the circle.

• It is held next to a whiteboard at which all the tasks and blocks are written when reported.

#Scrum Master reinforces vision  
–She needs to daily share and clarify the overall project vision, and

goals of the Sprint, perhaps at the start of the Scrum meeting.

# Replace ineffective Scrum Master

–  The manager/Scrum Master is the servant of the developers, not vice versa.

–  If Scrum Master is not removing blocks promptly, acting as a firewall, and providing resources, the Scrum founders encourage replacing the Scrum Master.

Values

1.Openness

• The openly accessible Product Backlog makes visible the work and priorities.

• The Daily Scrums make visible the overall and individual status and commitments.

• Work trend and velocity are made visible with the Backlog Graph.

2.Respect Or team responsibility rather than scapegoating.

• The individual members on a team are respected for their different strengths and weaknesses, and not singled out for iteration failures.

• The whole team rather than a manager, through self- organization and direction, adopts the attitude of solving “individual” problems through group exploration of solutions, and is given the authority and resources to react to challenges, such as hiring a specialist consultant to compensate for missing expertise.

3. Courage

• Management has the courage to plan and guide adaptively and to trust individuals and the team by avoiding telling them how to get the iteration done.

• The team has the courage to take responsibility for self-direction and self- management

XP is founded on four values: – Skillful development practices – Communication

– Simplicity  
– Feedback  
– Courage

– Collaboration

– Quick and early software creation

– Skillful development practices

XP recommends 12 core practices

1. Planning game

2. Small, frequent releases

3. System metaphors

4. Simple design

5. Testing

6. Frequent factoring

7. Pair programming

8. Team code ownership

9. Continuous integration

10. Sustainable pace

11. Whole team together

12. Coding standards

XP Lifecycle

1.Like many projects, XP can start with

exploration. Some story cards (features) maybe written, with rough estimates.

2. In the Release Planning Game, the customers

and developers complete the story cards and

rough estimates, and then decide what to do for the next release.

3.For the next iteration, in the Iteration Planning Game, customers pick stories to

implement. They choose stories – and thus steer the project – based on current status, and their latest priorities for the release.

Overtime is seriously discouraged in XP; it is viewed as a sign of a dysfunctional project, increasingly unhappy people, and dropping productivity and quality.

4. Developers implement the stories within the agreed timeboxed period, continually collaborating with customers (in the common project room) on tests and requirement details.

5. If not finished for release, return to step 3 for the next iteration.

**The Initial Process:** Levels: Initial: Chaotic-unpredictable cost, schedule, and quality performance,Repeatable:Intuitive-– cost and quality highly variable, reasonable control of schedules,informal and ad hoc process methods and procedures,Defined:Qualitative – reliable costs and schedules, improving but unpredictable quality performance,Managed:Quantitative – reasonable statistical control over product quality,Optimising- Quantitative basis for continued capital investment in process automation and improvement.Needed actions:Initial:Planning performance tracking, change control, commitment management, Quality Assurance,Repeatable:Develop process standards and definitions, assign process resources, establish methods,Defined:Establish process measurements and quantitative quality goals, plans, measurements, and tracking,Managed:Quantitative productivity plans and tracking, instrumented process environment, economically justified technology investments,Optimising-Continued emphasis on process measurement and process methods for error prevention

**The Project Plan** ● The Project Plan: The Work and how it’s done, benchmark to compare with actual performance. ● Project Planning Principles: quality program, precise of user needs, conceptual design, subsequent requirements, detailed design and implementation Strategy, implementation details, framework for negotiating the time and resources ● Planning Considerations: initial resource estimates unexpectable, early agreement on the essential functions, harmonious teamwork between the users and the software engineers ● The Planning Cycle: initial requirements, work Breakdown Structure (WBS), Product size estimates, Resource estimates, Project schedule ● Goals and Objectives: negotiation period, WBS: estimate of size of the product, Size Measures: Early estimates to actual, Function points from users’ perspective determines the number and complexity of inputs, summed to produce an overall function point rating, ● Estimating: Wideband Delphi Technique: - Group of experts given estimate forms, meet and anonymously complete the form, any method can be inaccurate, ● Estimating Inaccuracies: Normal statistical variations, estimating bias caused by the project stage, ● Estimating Contingencies: inflate the resource estimates, add to the schedule, and increase ● Productivity Factors: number of programmer months and time required, orderly procedure, – Organization productivity data, developing productivity data, Using productivity data, Resource estimating ● Developing Productivity Data: Identify a number of recently completed programs that are as similar, get data on the size, in LOC, count of the programmer months ● Using Productivity Data: derive base productivity numbers, ● Scheduling: total resource needs, organization’s historical experience, the overall project schedule objective, a staffing plan is developed, preliminary schedule, preliminary plan, ● Project Tracking: ability to determine project status, checkpoints, earned-value project scheduling, ● The Development Plan: submitted to management for review and approval, Project purpose and scope, Project goals and objectives, Organization and responsibilities, Management and technical controls, Work definition and flow, Development environment, Software development methodology, Configuration management, Verification and validation, Quality assurance provisions ● Planning Models: COCOMO, SLIM, error by 500 percent or more, used to augment, used effectively to check the estimates for errors or overs

**Managing Software Organizations** ● Role: successfully completed, organization-wide agreement, requires continuing management focus, ● Basic principles: plan based on commitments, commitment is by one person to do something for other, resolves conflict between line and staff, Oversights, and review system tracks progress against the plan ● Commitment Discipline: supported by plans, estimates, reviews, met by committed by people ● Making a Commitment: a planned completion date and some consideration or payment, when the coordinated efforts then mutual commitments are essential ● elements of an effective commitment: willingly, not lightly, agreement on what has to be done, by whom, when, publicly stated, advance notice if commitment not met ● The Commitment Hierarchy: done by committed individual not done in vacuum, a management team making commitments, ● The Software Commitment Process: top of the organization, made by senior exec, made only after formal review, enforcement mechanism to review and concurrence is maintained ● Establishing a Commitment Process: required planning before commitment, training courses, Estimating, review and approval procedure ● The Management System Goals: technical and business strategy, provide quality products, provide quality products, improve continually ● Product and Period Plans: To resolve the inherent conflict -> annual operating plans, specify the task, skilled resource allocation, ● Management Oversight: reviews and a contention system to resolve conflicts, each staff and line annual plan and reviews, separate plan is consolidated into org plan ● The Contention Process: requires a parallel contention system to encourage openness, best decision based on full understanding, ● The Principles of the Contention System: Reviewed and agreed by all parties, all parties present for decision, when no disagreement SM decides if knowledgeable agreement is there, if any decision is deferred, ● The Quarterly Review: forum for resolving conflicts and monitoring progress, assessment of progress on plan, and against goals, ● Project Phase Reviews: assess project progress periodically, review at key points, DoD standards MIL-STD-2167, MIL-STD-498, ISO/IEC 12207, ISO/IEC 15504(spice), ISO/IEC 15288, ISO 9000 Serie ● Establishing a Project Management System: if project management is not aware of then system can’t be effective