1. (a) **What is a project?** (1) A project is a sequence of unique, complex and connected activities that have one goal or purpose and that must be completed by a specific time, within budget, and according to specification. (2) Better definition: A project is a sequence of finite dependent activities who successful completion results in the delivery of the expected business value that validated doing the project. (b)**What is a program**? A collection of related projects, which may need to be completed in a specific order for the program to be complete. Programs may have more than one goal. (c) What is a portfolio? A collection of projects that share a common link to one another. 2.**What is project management?** Project management is organized common sense. **Challenges**: (a). Flexibility and adaptability (b). Deep understanding of the business and its systems (c) Take charge of the project and its management. **Process:** scope, plan, launch, monitoring & controlling, closing. **Benefit:** Organization manage projects so that they can be more certain the project will deliver value

3. Projects for which traditional project management approaches maybe appropriate: (1) The goal and the solution are clear (2) About 20% fall into this category, often includes infrastructure projects (3) Little change expected (4) Limitations: 1. Change intolerant 2. Focused on delivering according to time and budget constraints 3. Rely on compliance to plan rather than necessarily business value (5) Characteristics: 1. Low complexity 2. Few scope change requests 3. Well-understood technology infrastructure 4. Low risk 5. Experienced and skilled project. (6) Linear life-cycle model --change intolerant (7) Incremental lifecycle model-- Includes long cycle feedback loops 4. Projects for which agile project management approaches may be appropriate

#### 1. Need is clearly defined but how to meet that need isn't so obvious 2. Medium to high complexity 3. Characteristics: 1. critical problem with a known solution (1). You need to discover an acceptable solution by completing the project! (2). Variable scope 2. A previously untapped business opportunity 3. Change driven--Requires just-in-time planning 4. Successful solution is critical to organization 5. Requires and has meaningful client involvement 6. Small, and where-ever possible, co-located, multidisciplinary teams 7. Lifecycle models (1). Iterative (2). Adaptive

#### Projects for which extreme project management approaches may be appropriate: (1) Where solution and goal are not known or clearly defined (2) New product development, process improvement (3) Characteristics 1. Research and development 2. Very high risk 3. High change 4. Often high speed and high failure 5. Highly iterative

##### 5. Five Process Groups **are not a project management lifecycle. They are simply groupings of processes by project phases. A specific project management lifecycle is defined using these processes.**

**6.** Agile organizations harness change by (1) Responding to changing conditions and (2) Creating change that competitors cannot respond to adequately

**7.** " APM” is the work of energizing, empowering, and enabling project teams to rapidly and reliability deliver business value by engaging customers and continuously learning and adapting to their changing needs and environments"2."APM is a way of managing projects to deliver customer value via **adaptive planning**, **rapid feedback**, **continuous improvement** and **intense human interaction and collaboration**"2. APM is related to the rolling wave planning and scheduling project methodology. It uses iterations (“time boxes”) to develop a workable product that satisfies the customer and other key stakeholders. Stakeholders and customers review progress and re-evaluate priorities to ensure alignment with customer needs and company goals. Adjustments are made and a different iterative cycle begins that subsumes the work of the previous iterations and adds new capabilities to the evolving product. Principle of APM: (1) Focus on customer value (2) Iterative and incremental delivery (3) Experimentation and adaptation (4) Self-organization Continuous improvement **8.** Agility is the ability to both create and respond to change in order to profit in a turbulent business environment

##### 9. APM Methodologies: APM Methodologies are based on the principles of complexity theory and the notions of complex adaptive systems. They view a project as non-linear, complex adaptive system where change is normal and take a “barely sufficient” approach to plans, process and control while focusing on delivering customer value. The focus of APM is on people and their interactions, the dynamic interactions of individuals and teams and give individuals the power to make quick decisions. They are adaptive rather than predictive, even self-adapting their own processes. APM are chaotic – that is, there is both chaos and order. Product goals are achievable but they are not predictable. While processes aid consistency, they are not repeatable. They value collaboration and barely sufficient methodology, which is based on practices (what happens in reality) not processes (what is described in manuals).

#### 10. Basics of Agile Methodologies (1) Small releases – helps manage complexity, provides early feedback. One to three months (2) Iterative and incremental development – plans requirements design code and tests are evolved incrementally through multiple passes or iterations. Iterations are fixed length (usually 2 weeks), which maximizes feedback. The fixed scope retains stability  (3) Collocation (…) (4) Release plan/feature backlog – desired features are defined at a high level and prioritized by the customer. Prioritization is done collaboratively in a release planning game (based on game theory). Developers provide effort estimates & customers decide the business priority. (5) Iteration plan/task backlog – high-level features from the release plan are elaborated(详尽的) upon and prioritized along with their implementation tasks in an iteration plan or task backlog. Prioritization is done collaboratively with developers in an iteration planning game. Developers provide effort estimates & customers decide business priority. (6) Tracking – features and tasks are tracked within an iteration. They only count as complete when they are 100% done..  (7) Self-organizing teams – team members self-organize by completing tasks collaboratively from backlogs without top-down management control. Simple, lean and adaptable – all aspects of work, including processes, are kept simple, lean and adaptable to maximize customer value and to accommodate change.  (8)Pairing–developers (and others) (9) perform production work in groups of two to collaboratively construct and share knowledge and enhance quality.  (10)Test-driven development – developers write tests before they write code and evolve the code to meet the tests. Test often specify rather than validate the code. NOTE: Test-driven and pairing are specific to extreme Programming

**Agile Methodologies: (1) Modelling, but not simply to place in a file collecting dust (2) Documentation – but not hundreds of pages which are rarely used and are not kept up to date (3) Plans – but accept the limits of planning in a changing environment.** 11. Advantages of APM: APM is useful in developing critical breakthrough technology or defining essential features. Through the use of continuous integration, verification, and validation of the evolving product and frequent demonstration of progress all work, the likelihood that the end product will satisfy customer needs is increased significantly. This approach leads to early detection of defects and problems.12. Limitations and Concerns of APM: It does not satisfy top management’s need for budget, scope, and schedule control. Its principles of self-organization and close collaboration can be incompatible with corporate cultures. Additionally, its methods appear to work best on small projects that require only five-nine dedicated team members to complete the work. APM requires active customer involvement and cooperation.  **Applying Agile to Large Projects**: (1) Scaling by using several teams to work on different features of a large-scale project at same time. (2) Staging which requires significant up-front planning to manage the interdependencies of different features It also involves developing protocols and defining roles to coordinate efforts and assure compatibility and harmony.Scrum consists of four phases: (1) analysis (2) design (3) build (4) test. **Key Roles and Responsibilities in the Scrum Process: (1) Product Owner** -- Acts on behalf of customers to represent their interests. **(2)** Development Team -- Is a team of five-nine people with cross-functional skill sets is responsible for delivering the product. **(3)** Scrum Master -- Facilitates scrum process and resolves impediments at the team and organization level by acting as a buffer between the team and outside interference. **13. Importance of critical thinking**: Critical thinking gives you a tool that lets you take what you have learned about the solution to a problem and then apply that solution in different ways to different, unfamiliar problems. **14**. Reflection helps you become more active and aware of the learning process by requiring you to focus not only on what you have learning but also on how you learnt it. **15.** **Leadership** is about providing direction, is a process whereby an individual influences a group of individuals to achieve a common goal. **Management** is about ensuring something is accomplished, is to exercise executive, administrative, and supervisory direction of a group or organization. All management functions can potentially provide leadership, vice versa. Managers are people who do things right and leaders are people who do the right thing. The difference may be summarized as activities of vision and judgment -- effectiveness -- versus activities of mastering routines – efficiency **16**. **Essential management skill:** 1. Technical skill 2. Human skill 3. Conceptual skill **17**. (1) Management produces order and consistence (Planning and budgeting, Setting timetables); Leadership produces change and movement (Establishing direction, Clarifying the big picture) (2) Managing=coping with complexity (Formulate plans and objects, Take corrective action); Leading=coping with change. (Recognize the need to change to keep the project on track, Provide direction and motivation) **18**. A feasibility study has six key elements: 1. Clear & well-understood definition of the scope of proposal 2. What is the current situation? --Statement of the “world” is it is currently 3. Requirements: (1). Statement of the problem (2). State of the world after project implemented 4. Approach: 1. considers the options / various alternatives 2. Build versus buy / In-house versus contract 3. Explanation of why preferred option was selected 5. Evaluation: 1. Cost-effectiveness of selected approach 2. Includes estimations for other potential approaches 3. High-level project schedule 6. Review & decision **High power, interested people**: fully engage and make the greatest efforts to satisfy. **High power, less interested people**: keep them satisfied, but not so much that they become bored with your message. **Low power, interested people**: talk to them to ensure that no major issues are arising. These people can often be very helpful with the detail. **Low power, less interested people**: again, monitor these people, but do not bore them with excessive communication. **Project scope**: The work that must be accomplished to deliver the product scope = the Statement of Work (SoW) **Product scope**: The features and functions characterizing a desired product, service or result = the outcome of the project **What is a project scope?** A definition of the end result or mission of the project—a product or service for the client —in specific, tangible, and measurable terms **Purpose of the scope statement:** 1. To clearly define the deliverable for the end user. 2. To focus the project on successful completion of its goals 3. To be used by the project owner and participants as a planning tool and for measuring project success. **Characteristics of successful team: 1.** Good communication 2. Courage to confront and resolve conflict 3. Ability to give positive feedback 4. Ability to empathize with other members 5. Willingness to put aside personal goals in order to achieve team goals **Project Success Factors** (1) Clear, well-understood project objectives (client's needs, Constraints, Factors within and outside PM control) (2) Strong business case (3) Sufficient planning detail(Agreed deliverables, Accurate cost estimates)  (4)Motivated and committed team (5)Controlling the scope (6) Good risk management (7) critical success factors agreed with the client **19.** In agile project management, the scope of the project is captured in the "Product Backlog" The agile product backlog is (1) a prioritized features list, (2) containing short descriptions of all functionality desired in the product. **20**. Unlike many traditionally managed project, agile projects don’t start a project with a lengthy, upfront effort to document all requirements. Instead, an agile team and its product owner typically begin by writing down everything they can think of to include in the backlog. The product backlog is allowed to grow and change as the project progresses and as more is learned about the product and its customers. **Project charter is a documentation defines purpose of a team, how it works and the expected outcomes.** **21**. A typical scrum backlog comprises the following different types of items: (1) features (2) Bugs (3) Technical work (4) Knowledge acquisition **22**. User stories are short, simple descriptions of the desired functionality told from perspective of the user. User stories=requirements **23. Product Backlog Issues:** (1) Long lists of “stuff to do” - flat structure (2) No better than poorly developed WBS / PBS – that focus on delivering the items on the list rather than delivering value (3) A list of stories without any connection to the value that is to be delivered to users who are trying to solve problems. **24. some key words of user stories:** (1) “Stop exchanging documents. Instead, tell me your story.” (2) “Shared understanding and alignment are the objectives of collaborative work” (3) User stories are supposed to spark conversations. (4) “Story maps organize and structure those building blocks all of which enhances the communication process, which is most critical part of software development” (5) Stories get their name from how they should be used, not what should be written (6) A story is supposed to be a promise to have a conversation around the subject of the story, to eliminate misunderstandings through speech, not writing. This means that a user story can be very high-level and somewhat abstract, and the discussion around it will make it more tangible (7) Users stories help develop a shared understanding between the customer and the development team and should be focused on the big picture (8) User stories help frame the product. (9) When working with stories, it is important to keep in mind the now and later model; that is things are done "just-in-time" in agile projects. Do not provide too much detail too early. It is likely to be wrong and is a waste of time and resources. (10) User stories are used to capture functional and non-functional needs. (11) A user story is a short, simple descriptions of a feature from the perspective of the person who desires the new capability, usually a user or customer of the system. (12) **As a <type of user>, I want <some goal> so that <some reason (business value)>**. (13) User stories say nothing about HOW (14) User stories represent rather than document customer requirements. **25**. **Light-weight approach:** (1) Slicing user stories down in release planning (2) Tasking user stories out in sprint planning (3) Specifying acceptance criteria for user stories early in development ***26.*** *“The goal of using stories isn’t to write better stories.* **(is developing shared understanding)** *The goal of product development isn’t to make products”* **27**. Good story conversations are about who and why not just what. We need to remember that longer-term stuff happens as a consequence of good outcomes – **Impact. Outcomes are visible immediately following delivery but impact takes longer. 28**. **The overall goal must be to minimize output, maximize outcome and impact. 29. Your job isn’t to get the requirements right; it is to change the world!**  **30.** Stories are not the requirements! (1) Stories are not written form of requirements. -- Telling stories through collaboration with words and pictures builds shared understanding (2) Stories are discussions about solving problems that lead to agreements on what to build. (3) Your job is to maximize the outcome and impact obtained from what you chose to build. **31. User Story Maps** - A Goal-driven Product Backlog User Story Maps are a very powerful technique for assuring delivery of value and for assessing the completeness of your requirements. They are a tool for assuring each release or iteration adds value by enabling a user to solve, or more effectively solve, a problem – avoiding a delivery that only solves half of a problem. **32. Benefits of User Story Maps** (1) allow you to see the big picture in your backlog, (2)provide a better tool for making decisions about grooming and prioritizing the backlog (3) promote silent brainstorming and a collaborative approach to generating user stories (4) encourage an iterative development approach where early deliveries validate your architecture and solution (5) a great visual alternative to traditional project plan (6) a useful model for discussing and managing scope (7) allow you to add a visual dimension to planning and real options for your project/product **33. Summary of story map concepts** (1) Tasks are short verb phrases that describe what people do (2) Tasks have different goal levels (3) Tasks in a map are arranged in a left-to-right narrative flow (4) The depth of a map contains variations and alternative tasks (5)Tasks are organized by activities across the top of the map (6)Activities form the backbone of the map (7)You can slice the map to identify the tasks you’ll need to reach a specific outcome **34. Using your User Story Map to Plan** (1) Activities are the backbone: The essential activities required to deliver MVP. Do not prioritize, without any of these you do not have an MVP (2) The walking skeleton有: The highest priority tasks immediately below the backbone. The smallest possible system that provides end to end functionality (3) Prioritize tasks below the backbone: Move activities up or down (4) Releases can be marked out in “swim lanes”: Stories can be moved in or out of releases. Stories can have different heights within a release (5) Build all the major features a little at a time: Each release always adds value. **35. Issues for software development projects execution.** Vital issues: (1) The evolving nature of requirements and understanding (2) The complexity of business needs and solutions (3) The criticality of technology to business success (4) The uncertainty of component maturity and/or quality (5) The uncertainty surrounding staff and/or resource availability (6) The variability of supplier capability and workmanship (7) The expense of failure or delay, both business and financial (8) The expense of maintenance (9) The cheapest bid often wins. **36. Tips to help minimize some of the issues** (1) The right stakeholders need to be involved in problem definition. (2) Stakeholders need to clearly understand their operational/business needs. (3) Stakeholders need to define the problem, not the solution. (4) Solution and delivery constraints need to be real, applicable and negotiable. (5) Development strategies (the selected life cycle model/s) should cater for the evolving nature of business and product requirements. (6) Progressive/evolving understanding of a required outcome/technical solution need is the norm. (7) Effective project management is not static – it needs to be disciplined and agile (8) It is important to focus on the entire scope of change to the business that is affected by the project. (9) The initial plan is only good for a limited time – things always change. (10) As things change during a project, information has to be gathered, decisions made, actions taken and implementation effectiveness monitored. (11) Accountabilities and authorities for decision making and product development should be established prior to the commencement of development work. (12) Measures of success should be defined prior to commencement of development work. (13) A consistent and coherent approach should be applied to all technical and management activities on complex and high-risk development work. (14) Proof of quality should be sought and delivered incrementally throughout a development activity. **37. What makes a good (project) plan?** A good project plan is one that is sufficiently reliable that stakeholders can use it to help with decision-making. However, accuracy improves as one moves through a project. With each iteration, the detail increases and therefore so does accuracy. 38. Planning is iterative. A good planning process is difficult. It needs to take account of the project variables of : Scope, Time, Cost, Risk, Resources but not quality(恒定). **39. Why planning fails:** (1) Planning by activity rather than feature (Activities don’t finish early, Lateness is passed down, Activities are not independent, Multitasking slows productivity) (2) Features are not developed according to priority / importance (3) Uncertainty is ignored (4) Estimates become commitments. **40. Phases of the Agile Project Management Lifecycle** (1) Initiation phase (2) Interim phase—releases (3) Final phase—the project retrospective (4) Rinse & Repeat **41. Effort versus Duration:** Effort--The number of labor units required to complete a schedule activity or work breakdown structure component. Usually expressed as staff hours/days/weeks. Duration--The total number of work periods (not including holidays or weekends) required to complete a schedule activity or WBS component. Usually expressed as workdays or work-weeks. Converting Effort to Duration--Planning involves estimating the effort required and then converting it to or calculating the duration required. We do not estimate duration. **42. Product Vision and the Product Backlog**: Focuses on the What, Who, Why, When, Constraints, and Assumptions. The product owner owns the product vision but his peers work with him to create it. Big deliverables appear on the backlog and acceptable inaccurate estimates, made by the product owner/architect. (1) Timing: With a new product, or with significant changes in the product design (2) Visioning could be: An elevator statement; A product box; A metaphor (3) Goal: To move the project into the portfolio. **43. Prioritization** of the backlog is driven by the customer, in conjunction with the developer. Together they choose features to fill up velocity of iteration, based on desirability. **44. Product roadmap:** Focuses on Releases - the Date, Theme/Feature Set, Objective, and Development Approach. When preparing the product roadmap the vision statement and the initial product backlog are reviewed. (1) Timing: Either during portfolio decisions or just after project approval (2) Who: Product owner (3) Goal: The whole product is sketched on a timeline. 45. **As the Product Roadmap is created**: (1) Epics added to product backlog – product appears (2) which are then ordered in time– the priority of work appears (3) all the epics are estimated – the project size appears **46. Guidelines for Agile Planning** (1) Plan and elaborate progressively at each planning level. (2) Choose a proper planning horizon and granularity of planning and workflow monitoring for each level of planning (3) Identify proper agile planners and stakeholders who must work together with a commitment (4) Define workflows at each level of planning and execution. **47. Slice through functionality**: Use the story map to create a development plan by "slicing" horizontally through the stories that represent the functionality. (1) First slice: It shows the functionality working from end-to-end. This first slice doesn’t work in all needed situations and if shipped users would be upset, but it allows real data to be used to see how it performs and helps the team learn about technical risk. (2) Second slice: It helps build functionality and gets closer to being releasable. It starts to indicate how well the system performs in reality so that the team can understand the “predictably unpredictable”. (3) Later slices: It helps refine picture and make it polished. These slices add in some of the unpredictable things. (4) Important: Don’t plan to release each slice, some may simply be an internal milestone. Each slice has a different learning goal. **48. Opening-, Mid-, and Endgame Strategy: (轮廓, 底色, 完稿).** **49. “Ready”—**Story: (1) is defined clearly enough that all members of the team understand what must be done (2) includes a clear statement of resulting business value that allows the product owner to prioritize. (3) A story is not "ready" if it depends on something outside the team’s control. Increasing the risk of iteration fail. **49. Decomposition of Epics:** Don’t break big software development into big plans. Break it down into smaller pieces that don’t quite look like the finished product. Breaking bigger things down to smaller "evaluable" parts means you can learn sooner. **50. Good user stories follow Bill Wake’s INVEST model:** Independent, Negotiable, Valuable, Estimable, Small, and Testable. **51. The right size depends on perspective:** size, generally smaller is better than larger. It needs to be small enough for team to understand and to develop in a short time period which is usually less than one iteration. It needs to be big enough to represent business value in its own right. **52. Discovery**: discovery teams are the hub of collaboration. 53. Story-writing workshop: The goal of the workshop is to collaborate to develop solid shared understanding. The focus is on answering the question: exactly what will be build. **53. User story in details:** Remember the client, product owner or whoever is in that role, understands the problem much better than they are able to predict how to solve it and the person who understands the technology is often the most qualified to solve the problem. **54. Planning the end-to-end experience: Plan end-to-end experiences NOT features**.Features are part of the product development process: they are not part of what users see as the solution to a problem. **To develop truly valuable, and, therefore, successful, solutions developers must provide users with seamless experiences, not just an integration of features.** **55. End-to-End experience:** is what the customer sees, feels, and does when he or she uses your product, device, or service in a real-life situation, from the very beginning to the very end. Understanding true end-to-end experience requires insight into the customers’ motivations and overall situation. The big picture helps identify customer needs and opportunities. The big picture is deeply rooted in the real life environment, situation, and motivation that surrounds its use. **Optimizing features do not make an experience! 56. MVP:** is the smallest solution release that successfully achieves the desired outcomes. **SMURFS (Specifically, Marketable, Useful, Releasable, Feature, Set). 57. Estimation**: is carried out in any project on a number of different occasions. Just how frequently, depends on the project management approach adopted. Relative estimation and the use of story point is an evolution of the approach to estimation; an evolution toward simplicity. (1) Estimating: The process of forecasting or approximating the time and cost of completing project deliverables. (2) Scheduling According to the: "The process used to determine the overall project duration and when activities and events are planned to happen. This includes identification of activities and their logical dependencies, and estimating of activity durations, taking into account requirements and availability of resources." **58. Estimation is hard**: Your estimates will never be "correct", but they will get better as you revisit and refine your estimates. The greater your experience with estimation the better your estimates will become. Another problem which all project management approaches face is that the project manager is asked to provide a quick "ballpark" estimation of time required to complete the project which then becomes a commitment despite informing management that it is only a rough estimate. The uncertainty of scope, technology and the team itself, all work together to increase the difficulty of estimation. **59.** Complexity relates to the project, not the approach to management. **60. Minimizing uncertainty:** (1) One approach is to create a "walking skeleton" as early as possible. This approach helps flush out difficult design and requirements assumptions early, and helps to develop confidence within the team and with the customer. (2) Agile approaches attempt to overcome uncertainty by increasing the frequency of feedback by shortening the time between estimation and receiving feedback on the accuracy of the estimate. APM also increases the frequency of estimating as it does not carry out detailed estimation of activities too far into the future where uncertainty is greater. **61.** Duration: the amount of time that passes on the clock to do "something" Velocity: the measure of the team's rate of progress. **62. Units of measure**: The first step is to determine the unit of estimation. (if we don't understand the unit, the estimate means nothing) **63. Relative units of measure**: (1) Story points A story point is a unit of measure used to express the overall size of a user story, feature or other piece of work. What matters are the relative value. **64. Story points are additive, ideal time is not.** 65. Derive your estimates collabo -ratively rather than doing them on your own as they will be much more accurate. **65. Estimation approaches:** (1) Expert opinion: used more frequently with traditional rather than APM; can be individual or group; relies on "gut feel" based on experience; Disadvantage for APM (2) Analogy: estimator compares the story to an assortment of other stories which have already been estimated - known as triangulation; NOT against a single baseline (3) Disaggregation: by splitting the feature/project into smaller features and estimating each; need to make sure you don't miss any tasks, which it is very easy to do; sanity check (4) Planning Poker: combines expert opinion, analogy, and disaggregation; enjoyable; produces quick and reliable estimates. \*\*The goal of estimation is not to derive an infallible estimate but to achieve a valid estimate cheaply. The goal is not absolute precision but reasonableness. **66. Size** does NOT equal duration, it is relative. We estimate the size of the task from which we derive the duration and it is the duration that we use to determine the schedule. **67. Story points** are an abstract unit of measure which indicate the size of a story. The team determines how story points translate to effort and establish their own fixed comparison of points. A story point is an amalgamation of the (1) amount of **effort** involved in developing (2) the story the **complexity** of the story and its development (3) the **risk** involved (4) the **quality** required. **68.** **Size is relative. Story points are relative**. 69. Sometimes when a team has difficulty using story points, because they can't visualize the size, they may use t-shirt sizes or "dog points". A dog point represents the height of the dog at the shoulder. **70. Estimation using ideal days** is based on (1) the story is the only thing you will work on (2) everything is available at the start and (3) there will be no interruptions. **As a measure of size**, ideal deals do not consider overheads. However, they can be converted using velocity into an estimate of duration. **71.** **Estimate size BUT derive duration**. **72. Velocity** is a measure of the team's rate of progress and is calculated by summing the total number of story points that have been completed during the iteration. To determine the total size estimate for the project, sum the story point estimates for each story. Using the team's velocity we can then estimate the number of iterations required. Duration is calculated by mapping the number of iterations required onto a calendar. **73. Re-estimation:** Story points are not an estimate of the time to implement a story. The amount of time it will take is a function of the size. Because of these two factors, it is not necessary to re-estimate when development slips behind projected progress. **Only re-estimate when it is believed that a story's relative size has changed**. **73. Considerations favoring story points**: (1) help drive cross-functional behavior (2) Story point estimates do not decay (3) Story points are a pure measure of size (4) My ideal days are not your ideal days. **Considerations favoring Ideal Days:** (a). Ideal days are easier to explain to people outside the team:  (1) They have an intuitive feel (直觉) on it" (2) Need to explain the "cone of uncertainty" and progressive refinement of planning/estimation accuracy. (b). At first, ideal days are easier to estimate. **74. Key advantages of using story points to estimate size:** (1)forces the use of relative estimating - have shown better at this than estimating time (2) focuses on estimating size, not duration (3) puts estimates in units that can be added together -- time-based estimates are not additive. **75. Planning poker** is a consensus-based, iterative estimating and planning technique that is loosely based on wideband Delphi. Use hours not points when estimating the sprint backlog. If you must average, take the median value not the mean. Teams lose out when they average. Some agreement will be stronger than others. Planning poker is ideally suited to estimating the product backlog. It is often used shortly after the initial Product Backlog is created. Planning poker should be used at least once per iteration. More accurate estimates are achieved because it uses multiple expert opinions from all disciplines. It leads to more accurate estimates because estimators need to justify their estimates.**76. The basis of relative estimation**: compare the task with something you have done in the past. **77. Velocity** is the average number of story points that a team can complete in an iteration. 78. Approaches to calculating velocity: (1)Use historical values (2)**Run an iteration(ideal way)** (3)Make a forecast. **78. Remember it is better to be roughly right than precisely wrong. 79. Iteration lengths:** the iteration length usually remains constant throughout the project, though they may be different for different teams on the same project. 80. Selecting an iteration length (1) the length of the release being worked on (2) the amount of uncertainty involved (It is important to choose the iteration length that maximizes feedback to the whole team. You need to consider how much feedback is required, the frequency and timeliness. Affecting this consideration is whether the project is an internal or an external project.) (3) the ease of getting feedback. **81. An appropriate iteration length** is one that encourages everyone to work at a consistent pace throughout the whole iteration. Too long: slack at first and stress at last. Too short: no recover. **82. Effective project management is not static** –the project management itself needs to be disciplined and agile. **83. A good project plan** is one that is sufficiently reliable that stakeholders can use it to help with decision making. As one moves through the project accuracy improves and detail should increase with each iteration. Regardless of approach, traditional or agile, planning is iterative. **84. Schedule=Release planning.** The purpose of release planning is to answer how much will be done? When can we ship this set of features? How many teams should be on this project? Necessary inputs: (1) Velocity (2) length of the project (3) prioritized product backlog. **85**. Progress is measured by the delivery of working software. The schedule is developed using prioritized requirements or features. Scheduling a project is done at a high level of abstraction of user stories. Don’t forget to include activities such as training in the schedule and make sure you choose an approach which reflects your environment **86. Release planning helps you** (1) decide WHAT must be developed for the product to be releasable. (2) determine HOW LONG development (3) convey expectations to management (4) with strategic planning (5) release planning provides a baseline against which to measure progress. It also provides context – shows how each iteration leads towards the whole. **87. Timing:** Release planning happens after initial approval of the project and then again at the start of every release cycle. **88. Approaches:**(1) start with an end date and see how much can be completed within that time frame. (2) Take a set of user stories and see how long it will take to develop them. **89. Baseline your release plan:** The baseline is your promise. It helps to keep rhythm and momentum and makes sure the team can commit to delivering the release. Each release usually contains a number of features. As the project progresses, always compare your estimates with your actuals. If you fail to do this your estimates will never improve in accuracy and you will never become a better estimator. **90. Wisdom=Experience+Reflection.** **91.** In the agile methodology Scrum, an iteration is called a Sprint. The next iteration commences immediately following the conclusion of the preceding iteration. **92. Approaches: velocity-driven and commitment-driven.** They can be combined. Velocity-Driven is based on previous velocity. Commitment-driven planning, stories are added one by one until the team decides it can complete no more during that iteration. **Velocity-Driven Iteration Planning:** (1&2) Adjust priorities and determine the target velocity -- is based on learning from earlier iterations (3) Identify iteration goal -- is a concise description of what the team will achieve this iteration. (4) Select User Stories -- is based on the priority assigned to each story (5) Split user stories into tasks -- include only work that adds value to THIS iteration. Tasks must be specific, and must include unit tests. (6) Estimate tasks (after tasks or at the same time) -- a collaborative activity as it helps to identify misunderstandings and so leads to greater accuracy of estimation. **Commitment-Driven Iteration Planning:** (1,2,3,4&5) As for velocity-driven planning, except these steps are done iteratively, story-by-story. (6) Seek Team Commitment -- The significant difference with this approach is that the team is asked to commit to deliver new functionality not a set of tasks. This commitment to delivery of functionality includes team commitment to delivering any new tasks discovered during the iteration. (7) Steps 2,3,4,5 & 6 are carried out iteratively until the iteration is “full”. **93. Value of Commitment-Driven Planning**: As the name suggests, velocity-based planning is based on the team's past velocity. Velocity is a measure of coarse-grained estimates and is valid for estimating overall amount of work. Velocity is not valid for estimating finer-grained, shorter-term work, as it is too hard to average the variability of velocity over a number of iterations. **94. Sprint planning meeting:** (1) during the first part of the meeting, t team forecasts the functionality to be developed. The product owner presents the ordered, prioritized product backlog items to the development team and the entire scrum team collaborates on understanding the work of the Sprint (product backlog, latest product increment, projected capacity of the team, and past performance of the development team). (2) The team decides how it will build the chosen functionality into a “Done” product increment. The product backlog items selected for the sprint and the plan for delivering them is called the sprint backlog. (3) By the end of the meeting, the development team should be able to explain to the product owner and the scrum master how it intends to work as a self-organizing team to accomplish the sprint goal and to create the anticipated increment. **95. The Sprint Goal**: The goal guides and drives all work of the sprint. If the work turns out to be different than expected, the development team collaborates with the product owner to negotiate the scope of the sprint backlog within the sprint. The sprint goal may be a milestone in the larger purpose of the product roadmap. **96. During the sprint:** (1) no changes are made that would affect the Sprint Goal; (2) the development team composition remains constant; (3) quality goals do not decrease; but (4) the scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned. **97. Sprint retrospective:** purpose- to inspect the process, to learn and to look for improvement. T**he sprint retrospective meeting** is held after every iteration, frequently immediately following the sprint review meeting. It provides an opportunity for the Scrum team to inspect itself and to create a plan for improvements to be enacted during the next Sprint. **98. The primary reason for managing project is to manage the risk that the project:** (1) won’t deliver value (2) will cost more (3) will take longer (4) will need more resources and (4) to know when to terminate a project. **99. Project Monitoring & Control – Activities:** (1) comparing planned versus actual performance (2) assessing performance for tolerances (3) identifying new risks & analyzing, tracking existing risks. (4) Maintaining a register of products & their status (5) project status reporting & project metrics (6) forecasting cost and time (7) implementing & monitoring change. **100. Staying in Control of your Project:** To keep control of your project involves monitoring progress against planned actual start and finish dates, the percent of work completed, resource utilization, costs, schedule variance, milestone slippage. To do this you have to control and approve schedule change. **101. Project Tracking** Project Tracking is the management of projects and includes but is not limited to measuring and reporting the status of milestones, tasks, and activities required in achieving the pre-defined project results. The needs of the project determine the level of detail at which you need to track progress. It requires a careful balance of monitoring to ensure that you will achieve the desired results and respond to any incidents or roadblocks while avoiding micromanagement and reducing team members' ability to make decisions. Tracking projects may feel useful and productive but it can just be “busy work”. Any amount of tracking or tool use will not overcome problems related to project team productivity, schedule slippage and so on. You need to use any tools effectively. Figure out in advance what you want to track and why. Then focus on trends. Use post-mortem reports / reviews to track lessons learned. **Remember: Project tracking is not managing or controlling your project.** 1**02. Getting your project back on track**: Reducing the time allocated to a critical activity usually incurs additional direct costs. Cost-time solutions focus on reducing activities on the critical path to shorten overall duration of the project. Reasons: (1) Time-to-market pressures (2) Unforeseen delays (3) Incentive contracts (4) Imposed deadlines and contract commitments (5) Overhead and public goodwill costs (6) Pressure to move resources to other projects. **103. get your project back on schedule** (1) Look at the big picture. That means you shouldn't look at individual tasks in isolation! What is the impact to the overall schedule? (2) Don’t let quality drop. Doing tasks hastily or in a slapdash way can come back to bite you later on: you might need to do it all over again or at least fix the errors you have just inserted into your code-base. Don’t compound the problem by doing a low quality job. (3) Work with the team not against them -- don’t cast blame! Brainstorm what needs to be done to get buy-in from the team. Don’t panic; once you have worked out your new plan inform the stakeholders as soon as possible? **103. Using Agile Metrics**: (1) Keep it simple – you only need a few. (2) maximize stakeholder ROI (3) look at trends, not absolute numbers (4) Don’t manage by the numbers: Expect people to fake the numbers! If you reward behavior based on the metrics then they are motivated to fudge them. Expect to change what it is you measure. **104.** Release burndown and iteration burnup charts are a primary method of communicating progress. Burnup: (1) shows work accomplished versus time (2) clearly indicates change in scope or estimates (3) is usually plotted for Release cycles that contain multiple Sprints. Burndown (1) shows work remaining (2) most commonly used with individual sprints. **104. Iteration review meeting is very informal. 105. Two approaches to giving constructive feedback:** (1) Straight and cold (2) sweeten it up a little. 区别在于but **106. Daily “stand-up” meeting:** 昨天干了什么,今天要干什么,有什么阻碍,用来解决问题的. 107. **effective meetings:** agenda, make decision(authority), record, discussion. **108. Governance** is the management framework within which project decisions are made. Why do we need it? (1) Make project decisions within the context (2) Communications work better when the people sending the messages have recognized authority. **Key questions**: Are we getting value for money? Do the proposed solutions meet our full expectations? **Three Pillars of Governance: structure, people, information**. 1**08. Progress report needs to summarize: schedule, budget, issue.** **109. We manage quality** so that we (1) understand what quality means to the customer (2) can take pre-emptive action as well as be responsive when problems with quality arise (3) understand the impact of changing project scope, schedule and budget. **110. Key features of agile approaches to Quality** (1)Quality is a whole-team responsibility (2) The concept of “done” is fundamental to quality development (3) Testing is an integral part of the process rather than something that is left until later (4) Testing is automated, but this is not the only form of testing (5) Continuous integration **111. Relationship between “done” and “conditions of satisfaction”**: (1)DoD is a special set of conditions which is applied to every user story before any story is considered complete (2) “CoS” are specific to an individual user story. **112. The importance of “done” for quality software** (1) Overcomes our tendency to indicate a task is complete, when what we really mean is that the programming is complete (2) Sets the minimum acceptable standard for the team (3) Enables the team to deliver potentially shippable software at the end of each iteration (4) Definition changes over time. **113. Quality is a Team Effort** (1) Customers suffer if product is of low quality (2) Entire team suffers if testing is not integrated into the process or it is not done at the right level (3) A good agile team is constantly vigilant of the state of its testing practices. **114. Benefits of investing in quality**: (1) An improved reputation for quality. (2) Lower post-release maintenance costs. (3) Smoother release cycles (4) Increased confidence that the system will work well and satisfy customers. (4) Protection from lawsuits initiated by unhappy customers, users, or those otherwise affected by system failure. (5) Reduce risk of loss of entire missions and even lives. **115. PMBOK Quality Management:** (1) Plan for Quality: Process of identifying quality requirements and/or standards for the project and product, and documenting how the project will demonstrate compliance. (2) Perform Quality Assurance: Process of auditing the quality requirements and results from quality control measurements to ensure appropriate quality standards and operational definitions are used. (3) Perform Quality Control: Process of monitoring and recording results of executing the quality activities to assess performance and recommend necessary changes **116. Quality Control Methods** (1) Formal and informal reviews: Peer reviews, walk throughs, inspections (2) Testing(3) Prototypes (4) Demonstration (5) Modelling (6) Simulation (7) Audits **117. Software Quality Characteristics**: Functionality, Reliability, Usability, Efficiency, Maintainability, Portability. **118. Testing and Maintenance:** (1) Testing can reduce the cost of quality (2) Successful testing is not a few people in a dark lab at the end of the project( Concurrent ,Cross-Functional, Testers are the only ones who design, write and run tests, Collaborative (3) Team must deliver what’s committed on time **119. The Paradox of Software Testing**: (1) Testing software tells us very little about its quality! (A failed test only provides evidence of non-quality; a successful test is only relevant to quality assessment if it previously failed) (2) Testing shows the presence of bugs not their absence. **120. Summary of quality management:** (1) Quality is a measure of conformance with requirements(set for products, processes and projects)(2) Quality management activities can be both reactive and proactive (3) There are costs associated with ensuring quality and the failure to ensure quality (4) Testing can help reduce the cost of quality (5) Successful tests don’t necessarily provide evidence of software quality (Testing shows the presence of bugs not their absence) (5) Software requirements are usually more than just functional requirements. **121. Risk:** (1) Uncertain or chance events that planning cannot overcome or control. (2)The combination of the probability of an event and its negative (or maybe positive) consequence (3) is a set of events that, should it occur, will have an effect on the achievement of objectives. **122. Risk Management**: A proactive attempt to recognize and manage internal events and external threats that affect the likelihood of a project’s success: (1) What can go wrong (2) How to minimize the risk event’s impact (3) What can be done before an event occurs (4) What to do when an event occurs. **123. Components of risks**: (1) uncertainty (2) impact. **124. Risk Evaluation:**  It evaluation involves comparing the level of risk found during the analysis process with previously established risk criteria. Risk analysis and the criteria against which risks are compared in risk evaluation should be considered on the same basis. The result of a risk evaluation is a priority list of risks for further action. **125. Risk treatment options:** a) risk avoidance; b) reduction of likelihood; c) reduction of consequences; d) risk transference; e) risk retention. **126. Benefits of Managing Risk:** (1) A proactive rather than reactive approach. (2) Reduces surprises and negative consequences. (3) Prepares the project manager to take advantage of appropriate risks. (4) Provides better control over the future. (5) Improves chances of reaching project performance objectives within budget and on time.