AGILE SOFTWARE ENGINEERING: SUMMARY

JOHN STOUBY PERSSON

AALBORG UNIVERSITY
DENMARK

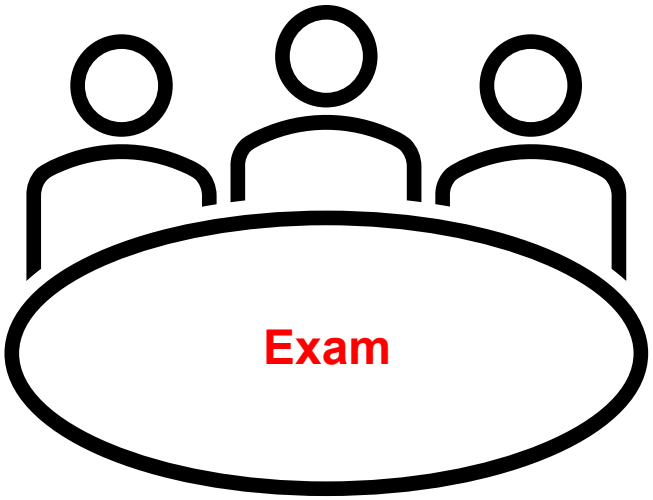
Last lecture's objectives

Knowledge about the software engineering problems that configuration management and DevOps helps to solve.

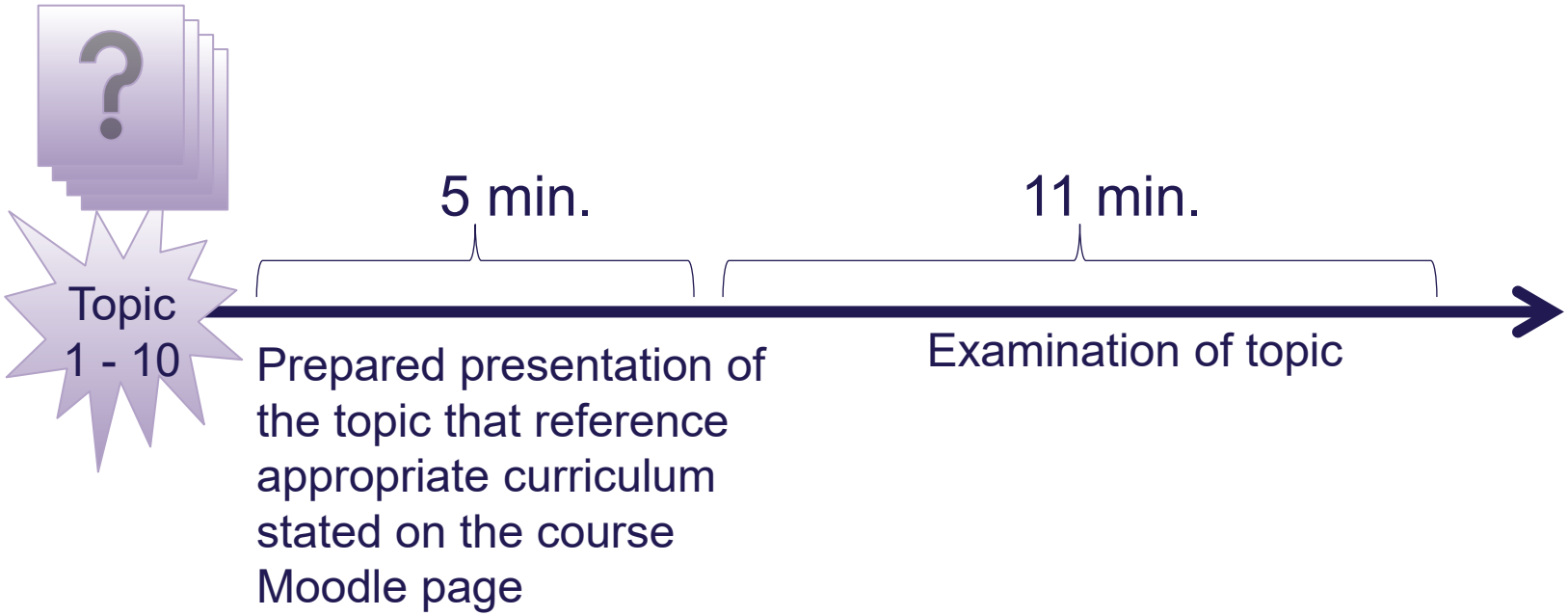
Skills in defining and managing processes for changes, versions, builds, and releases of software.

Competencies for managing software configurations in software engineering.

Lecture objectives



The exam (online using MS Teams)



A presentation may include 1) explaining the topic's problems, 2) definition of key concepts, 3) linkages to the debate on agile vs. plan-driven, and 4) its practical implementation. (showing a handout is allowed)

Exam topics

1. Software Process Model – waterfall
2. Software Process Model – Incremental / Iterative
3. Comparison of plan-driven and agile
4. Key features of Scrum
5. Key features of XP
6. Product Planning and Refinement
7. Risk Management
8. Quality Management
9. Test
10. Configuration Management and DevOps

Evaluering af kurset (7 minutter)

	Styrker	Svagheder
Indhold (fx. faglig dybde vs. brede)	?	?
Struktur (fx. forudsigelighed & opsamling)	?	?
Formidling (fx. forståelighed & interaktivitet)	?	?
Relevans (fx anvendelighed i projekter & praksis)	?	?

Forbedringsforslag?

1: Software Process Model - waterfall
Example questions

1. What is Software Engineering (SE) a response to? (complexity, failure)
2. What are the SE process activities? (specification, design, development, validation, evolution)
3. What is a software process model? (Set of related activities that leads to a software product)
4. Name the 3 typical process models? (waterfall, incremental/iterative, integration and configuration)
5. Describe characteristics of the waterfall model. (Activities in sequence, handover of work-products between phases, milestones and related work products are used to monitor progress)
6. When should you consider to use waterfall? (On projects using embedded systems; are life critical; or very large)
7. How can I decide if agile or waterfall is best fit for my situation? (Boehm: Analyze the home ground)
8. What is the deference between plan driven versus agile processes? (activities in sequence versus all activities at the same time)
9. Describe how the incremental model works, can it be plan-driven, can it be iterative? (You can iterate within increments; you can have increments planned)

2: Software Process Model – Incrementiel / Iterativ
Example questions

1. What is Software Engineering (SE) a response to? (complexity, failure)
2. What are SE process activities? (specification, design, development, validation, evolution)
3. What is a software process model? (Set of related activities that leads to a software product)
4. What are Sommerville's 3 typical models? (Waterfall, Incremental/iterative, Integration and configuration)
5. Describe the incremental/iterative model (You slice the big plan into smaller slices)
6. Can the incremental/iterative model be both plan-driven and agile? (Yes, e.g., in plan-driven you cut a predictive waterfall plan into slices, In iterative you work in fixed timeboxes and regularly update the full backlog for the project)
7. What are the advantages of incremental/iterative model? (a) price on requirements changes are less, b) easier to get feedback, c) customer gets earlier an opportunity to use part of the product and obtain the related value where use and value comes at the very end when using waterfall)
8. What disadvantages are there? (a) The process is invisible – management support for measurable progress can increase documentation cost, b) a software systems infrastructure tends to deteriorate, as new increments are added)
9. How can I determine if incremental/iterative or waterfall fits me? (Boehm: Analyze the home ground)
10. Describe how the incremental model works, can it be plan-driven, can it be iterative? (You can iterate within increments; you can have increments planned)

3: Comparison of plan-driven and agile
Example questions

1. What is the difference between plan-driven and agile? (plan-driven aim to predict desired results, agile expects change and uses frequent inspect and adapt to create best value)
2. How does Böehm & Turner define primary factors? (a) Application (small, rapid change, turbulent environment), b) Management (onsite, qualitative control, tacit knowledge), c) Technical (Prioritized informal requirements, simple design, d) People (Cockburn L2 and L3 developers)
3. What is the meaning of the 5 axes in the Home Ground Decision Tool? (Criticality, Personnel, Dynamism, Culture, Size)
4. Why do requirements change? (Business, technology, learning from use)
5. What is continuous integration in agile and how does it differ from prototype development? (A shippable product is maintained while prototypes should be discarded)
6. Agile key concepts (inspect and adapt to achieve desired value)
7. XP practices (Customer on site, pair programming, planning game, TDD, continues integration, sustainable pace)
8. Scrum vs. plan-driven roles (PO+SM+Team versus Lots of roles incl. Managers and specialists)
9. Scrum practices (Sprint Planning+Daily Scrum+Sprint Review+Sprint Retrospective,+Baclog refinement)
10. Agile vs. plan-driven Artifacts (Product Burndown+Sprint Burndown+Scrum board versus Project Plan, Gant chart, Requirement specification, etc.)
11. Plan-driven counterparts (predict what to deliver, plan the work, work the plan, knowledge sharing through documentation)

4: Key features of Scrum
Example questions

1. What is Scrum? (Iterative agile method)
2. Describe essential elements from Scrum. (a) Scrum roles, (Product Owner, Scrum Master, Team), b) Scrum practices (Sprint Planning+Daily Scrum+Sprint Review+Sprint Retrospective,+Backlog refinement), c) Scrum artifacts (e.g., Product Burndown+Sprint Burndown+Scrum board)
3. What is the focus of Scrum to the development process? (Focus on an empirical instead of defined process, and therefore the three pillars of Scrum are Transparency, Inspect and Adapt)
4. Can you mention one or more Core values in Scrum? (Commitment (to iteration goal), Focus (on iteration goals) , Openness (to work and progress) , Respect (or team responsibility) and Courage (for management to trust team, for team to take responsibility))
5. Can you mention some typical errors or mistakes in the use of Scrum? (a) Scrum master implemented as manager who tells team what to do (right way: Facilitator for team), b) Customers are not involved in each iteration, c) New requirements or tasks are added to team during iteration)
6. Can you say something about eXtreme Programming (XP) and what techniques that in particular fit to Scrum? (e.g., Customer on-site, user stories, planning game, etc.)
7. How defines Böhm & Turner the primary factors to balance plan-driven and agile? (a) Application (small, rapid change, turbulent environment), b) Management (onsite, qualitative control, tacit knowledge), c) Technical (Prioritized informal requirements, simple design), d) People (Cockburn L2 and L3 developers)
8. Why do requirements change? (Business, technology, learning from use)
9. How can you manage requirements and requirements change? (Change process, analysis of impact)

5: Key features of XP
Example questions

1. What is eXtreme Programming (XP)? (An agile (Iterative and incremental development method with focus on collaboration, early software creation and skillful development practices))
2. What values are XP based on according to Larman? (Communication, Simplicity, Feedback, Courage)
3. How is eXtreme Programming (XP) extreme? (E.g., if test is good do it all the time)
4. Name some of the key practices in XP? (unit test, pair review, Customer on-site, continuous integration, testing including Early test, Unit Test and TDD).
5. What is a user story? (Brief feature request, a promise for conversation. Written on card and criteria for confirmation written on the back)
6. What is the format of a user story? (”As a <user> I want <feature> so that <why>”)
7. How does XP describe Lifecycle for a System? (Exploration, Planning, Iterations to first release, productionizing, maintenance)
8. What is the iteration called in XP? (Iteration ☺)
9. What is Test Driven Development? (A work cycle: write test first, then simpelst code to pass test, then refactor code)
10. Why is Test Driven Development good? (You have a safety net, a low cost of defect, always doing small steps, Less fear to change code, and better code quality (readable, maintainable, less bugs))

6: Product Planning and Refinement
Example questions

Sub-topics include (Requirements Elicitation, Product Vision, Product Roadmap, User story mapping, Product backlogs)

1. What main requirement activities are there? (Elicitation and analyses of needs, specification of requirements, validation of requirements)
2. What are the steps in requirements elicitation? (a) Discovery & Classification, b) Categorization c) Prioritization & Negotiation, d) Documentation)
3. Why is it difficult to elicit requirements? (Many stakeholders with conflicting needs, stakeholders talk their own language, tacit knowledge and unconscious actions are not communicated, stakeholder and requirements engineer talk two different language)
4. What techniques can be used to elicit requirements? (Interview, Ethnography, Prototypes)
5. What is a recognized way to communicate requirements? (Stories / scenarios)
6. How are requirements documented? (a) Waterfall: Approved requirements document with strict change management, b) Scrum: Product vision and product backlog, reviewed and updated every sprint, c) Product Planning: Product vision, Release plans and/or product roadmaps, c) XP: User stories)
7. How are requirements negotiated with stakeholders? (a) Waterfall: Up front in the requirements phase – state it now or it will be difficult later to get it, b) Scrum: Ongoing refinement of product backlog with stakeholders, say what is most important now, we will continue, c) XP Customer on site)
8. How is plan-driven projects planned? (Plan the work, work the plan, assume you can predict what is delivered)
9. How is progress measured? (Through milestones and related documentation to provide a measure of progress according to plan)
10. What is agile planning? (Welcomes changes, works from a prioritized product backlog, where content is constantly refined from the top and adjusted to learning. This is called sprint planning in Scrum and the planning game in XP)
11. How can we estimate work? (a) using experience-based estimation (Planning poker), b) Algorithmic-based use of models, c) velocity-based for the team measured in story points)

7: Risk Management
Example questions

1. What is a risk? (Something that may happen and causes a loss)
2. Provide examples of risks and their categories. (categories: project, technical, business / risks: Keyperson from team leaves, a supplier is not delivering as promised)
3. How do you do risk analysis?
    a) Identify risks and calculate risk exposure (RE = probability * loss) and describe consequence
    b) Prioritize according to risk exposure (RE), establish cut-line
    c) Establish for each risk above the cut-line (RMMM: **m**itigation (prevention) plan – how can we prevent risk to happen, contingency plan (**m**anagement) plan – if it happens anyway, what do then do, how to **m**onitor development of the risk – called RMMM plan
4. How are risk management part of project management?
    a) Waterfall / plan-driven: a) Risk and risk plans are part of the plans in project management, b) Development of other plans contribute to identification of risk, c) It is planned how.
    b) Agile – inspect and adapt to produce the right product: a) Daily Scrum: Do you have any impediments, b) Sprint review: Inspects risk related to product and stakeholder, c) Sprint retrospective: Addresses risks related to how the team works.
5. What is the spiral model and how is it related to risk management? (uses prototypes iteratively to assess risks)
6. What are Boehm's primary risks? (Personnel shortcomings, unrealistic schedule, wrong function, ..)

8: Quality Management
Example questions

1. How can quality be defined? (Correspondence between experience and expectation of a product)
2. How is quality assured? (We plan how and when to do verification and validation)
3. What is Verification and Validation? (VER = compliant to spec, VAL = fit for use)
4. What techniques do we typically use for verification and validation? (Test is often used for verification, and review or evaluation are used for validation)
5. What is inspections and test good for? (code coverage, regression test, simplified debugging, documentation)
6. Why can't we have all quality attributes? (tradeoffs are necessary, e.g., reusability vs efficiency)
7. What is the V-model? (A model showing the relationship between test at different levels and primary activities driving the test)
8. What are the different tests of the V-model? (a)acceptance testing, b) system testing, c) system integration testing, d) sub-system integration testing, e) unit testing)
9. What should be considered when writing unit-tests? (Show the component works, reveal defects, possible inputs and outputs – to partition test data)
10. What agile practices support V&V? (a) Definition of Done, b) Sprint Review, c) Check before check-in, d) Never break the build, e) Fix problems when you see them, f) Culture where team members assumes responsibility for ensuring high quality, g) XP: Customer on site, h) XP: Pair programming
11. How does Pair-programming help ensure quality? (ongoing peer-review)

9: Test
Example questions

1. What is test? (a set of practices supporting Verification and Validation)
2. What is the purpose of testing? (To ensure a program does what is intended to do, and discover bugs before it is put to use)
3. What is an example of a test supporting verification and validation? (VER: Unit test, component test | VAL: Prototype test, user acceptance test)
4. What is peer review? (evaluation of work by one or more people with similar competences as the producers of the work (peers). Work is mostly documents but can also be static analysis of code)
5. What is the difference between review and test? (Review is static, and there are no interactions between errors found in review. Test is dynamic, and after first bug, other bugs may be a side-effect)
6. When is either review or test good? (Review: For documents, designs, architectures, plans | Test: For functionality and dynamic use of the program)
7. What is the test focus of a unit test, integration test, and acceptance test? (Unit: Verify valid and invalid inputs | Integration: Verify interfaces are compatible and work as expected | Validation: fit for use, exploratory test)
8. When is test done? (Plan-driven: in the end (Often a dedicated test-team as part of QA) Agile: all the time (Test competence in the team, accept criteria on story, automated test, TDD))
9. What is the agile testing quadrant? (An agile categorization of different types of test (can be considered an alternative to the V-model) along two axes: Technology facing ←→ Business Facing and Support the team ← → Critique the product)
10. What is best, from an agile perspective, many manual test, or many automated test? (Many automated, few manual is best Why? Otherwise, test takes too long, are error prone, feedback from test is too slow)

10: Configuration Management and DevOps
Example questions

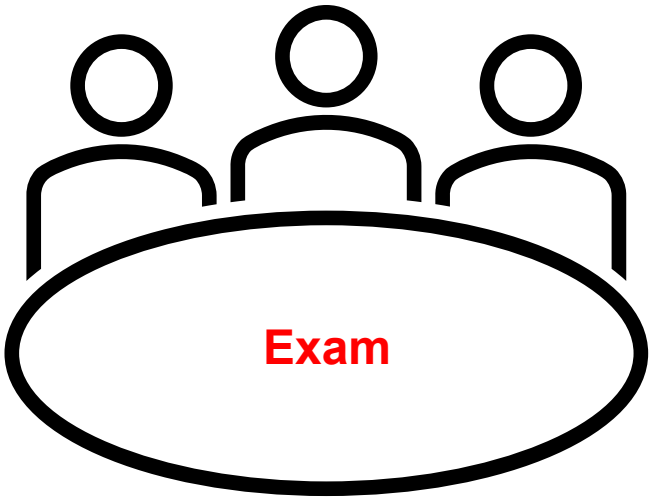
1. What is CM concerned with? (Policies, processes & tools for managing changing software systems)
2. What are the key activities in CM? (Version management, System building, Change management and release management)
3. What is CM, branching, merging (Techniques to support release, builds, baselines)
4. What is a baseline? (a description of a release that allows us to build the release again in the future. It includes a description of the version of code files that go into the release AND any documentation and external libraries as it was when the release was created.)
5. What goes into a release? (All code, data, configurations files, documentation)
6. What is DevOps, how can you define it? (DevOps is a practice of both the development and the operations and a development method for the IT systems that connect the different activities in the project. It is also defined from The Three Ways: Flow, Feedback, continuous learning)
7. What is the purpose of Continuous Integration? (When the code is checked in it will then automatically be integrated with a tool. CI will integrate the developers' work as early/often and get them constantly tested)
8. What is the purpose of Continuous Delivery and Deployment? (Continuous Delivery works with ensuring that the code can be safely deployed (production), to ensure that the business and service application work as expected and delivers every change to production. Continuous deployment works with the automated test and that every change is deployed to the production automatically while making the development and release process faster and more robust)

Group exercises

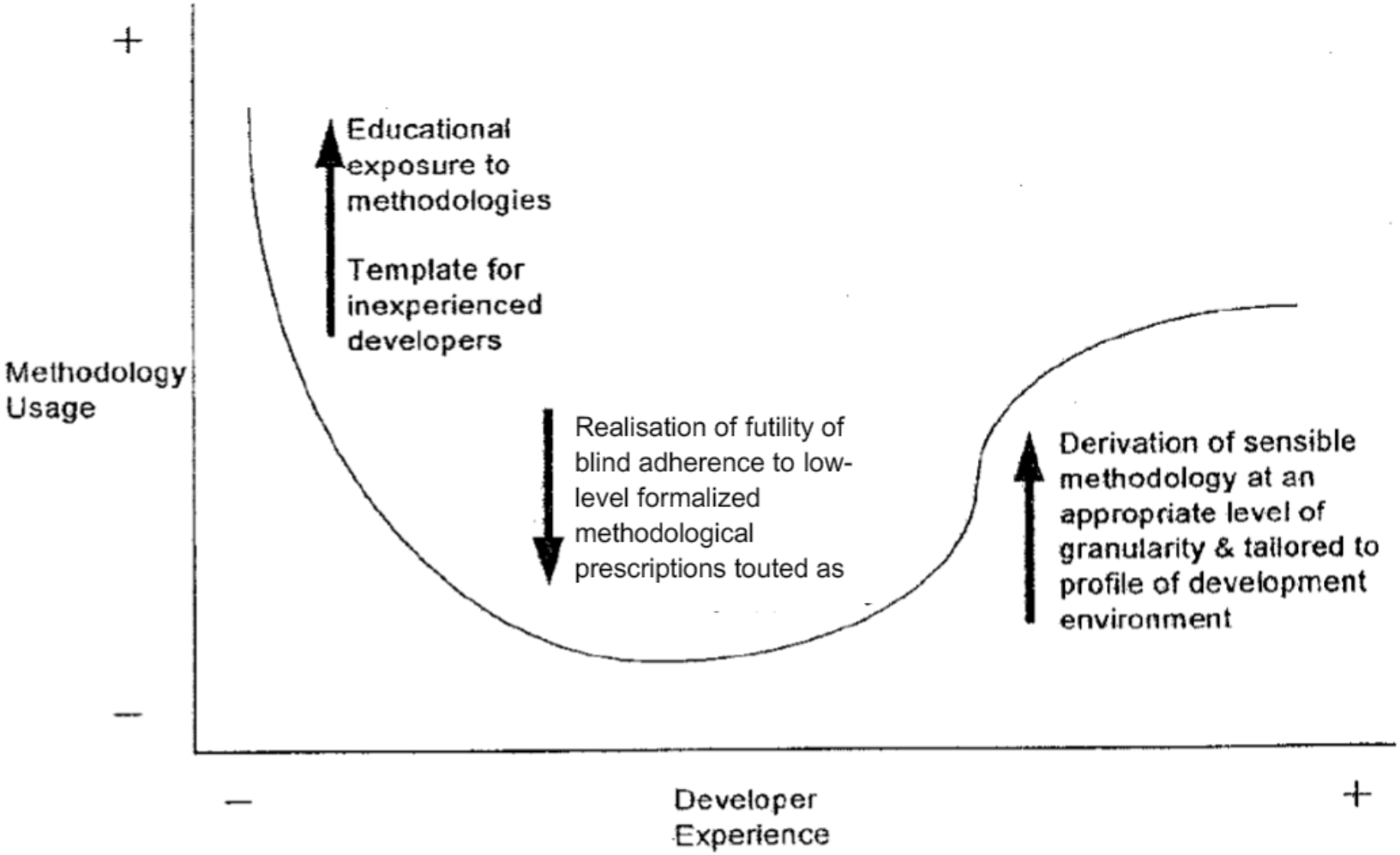
Exercise 1: Revisit and discuss the 10 exam topics' relevance for your semester project (100%).

1. Software Process Model – waterfall
2. Software Process Model – Incremental / Iterative
3. Comparison of plan-driven and agile
4. Key features of Scrum
5. Key features of XP
6. Product Planning and Refinement
7. Risk Management
8. Quality Management
9. Test
10. Configuration Management and DevOps

Lecture objectives



Use of methodologies with increasing developer experience



AALBORG UNIVERSITY
DENMARK

Fitzgerald, B. (1997). The use of systems development methodologies in practice: a field study. *Information Systems Journal*, *7*(3), 201-212.