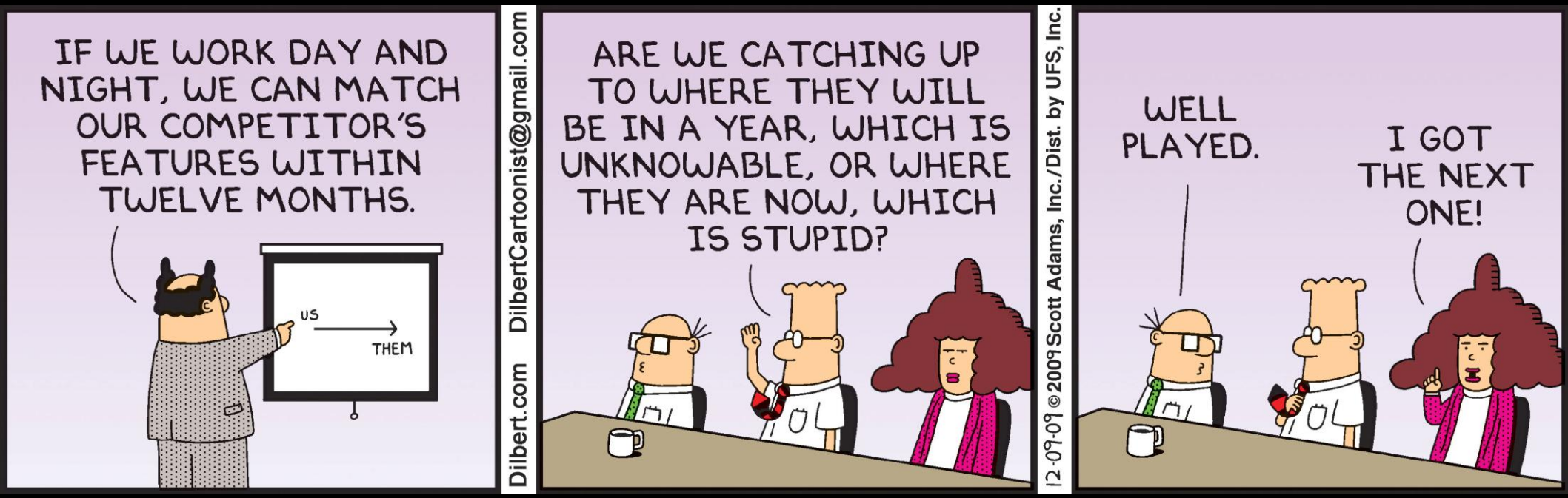
DAT255 / DIT543 SOFTWARE ENGINEERING PROJECT



https://github.com/hburden/DAT255/blob/master/README.md

ME MYSELF & I

Håkan Burden
RISE Viktoria

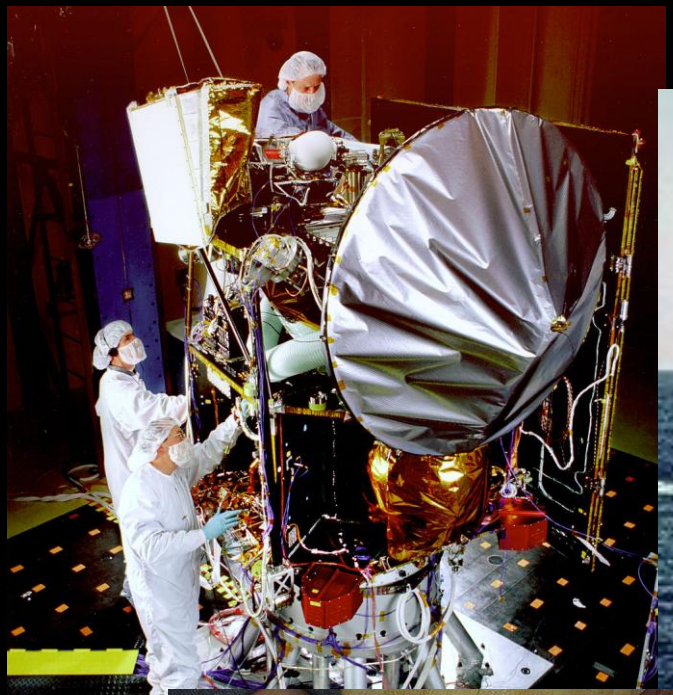
Contact:
burden@chalmers.se

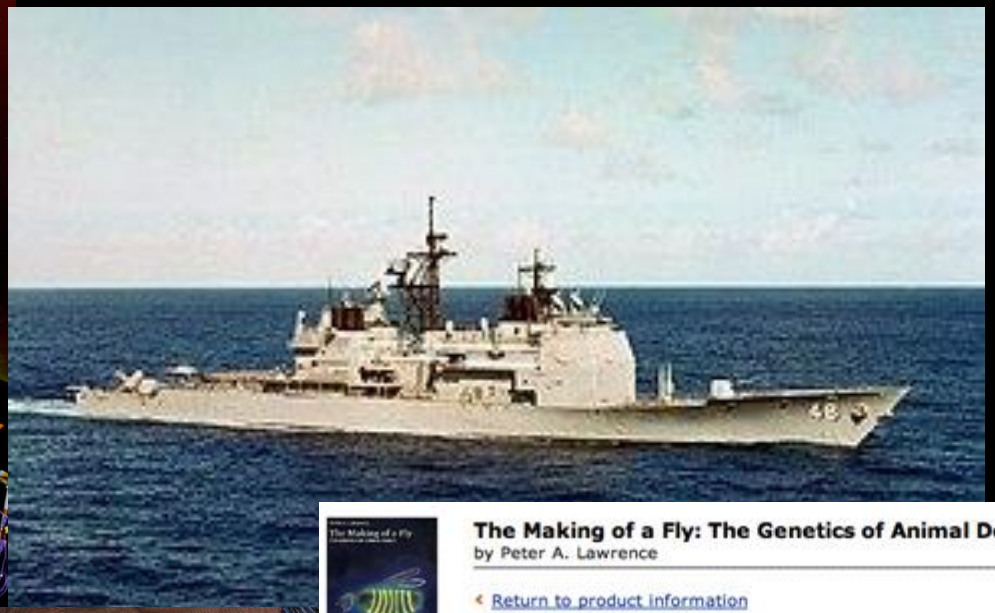
TODAY

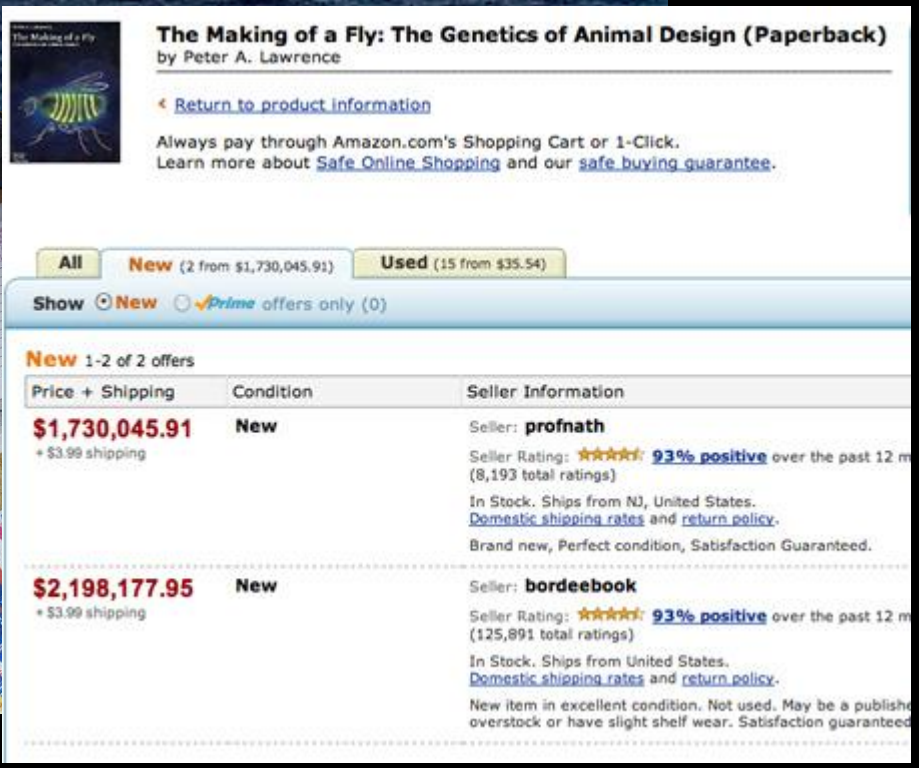
What is Software Engineering?

Learning goals
Learning activities
Assessment

Practical stuff







The Making of a Fly: The Genetics of Animal Design (Paperback)
by Peter A. Lawrence

‹ Return to product information

Always pay through Amazon.com's Shopping Cart or 1-Click.
Learn more about Safe Online Shopping and our safe buying guarantee.

All | New (2 from $1,730,045.91) | Used (15 from $35.54)

Show ⊙New ○Prime offers only (0)

New 1-2 of 2 offers

Price + Shipping	Condition	Seller Information
$1,730,045.91 + $3.99 shipping	**New**	Seller: **profnath** Seller Rating: ★★★★★ **93% positive** over the past 12 m (8,193 total ratings) In Stock. Ships from NJ, United States. Domestic shipping rates and return policy. Brand new, Perfect condition, Satisfaction Guaranteed.
$2,198,177.95 + $3.99 shipping	**New**	Seller: **bordeebook** Seller Rating: ★★★★★ **93% positive** over the past 12 m (125,891 total ratings) In Stock. Ships from United States. Domestic shipping rates and return policy. New item in excellent condition. Not used. May be a publishe overstock or have slight shelf wear. Satisfaction guaranteed

SOFTWARE CRISIS

Projects running over-budget.
Projects running over-time.
Software was very inefficient.
Software was of low quality.
Software often did not meet requirements.
Projects were unmanageable and code difficult to maintain.
Software was never delivered.

COMPLEXITY

“The complexity of software is an esse
property,notanaccidentalone.”

Fred Brooks, 1986

MLOC

- Simple app: 0.001
- Windows NT 3.1, 1993: 5
- Firefox: 10
- MS Office 2013: 40
- Facebook: 60
- Modern car: 100

SOFTWARE ENGINEERING

Systematic & disciplined approach
to the development and maintenance
of software
to assure quality of processes and products

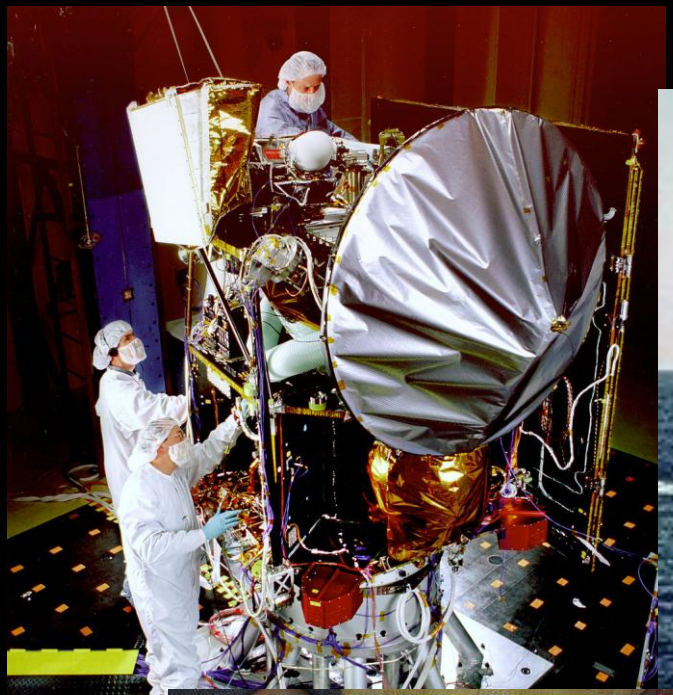
WATERFALL APPROACH

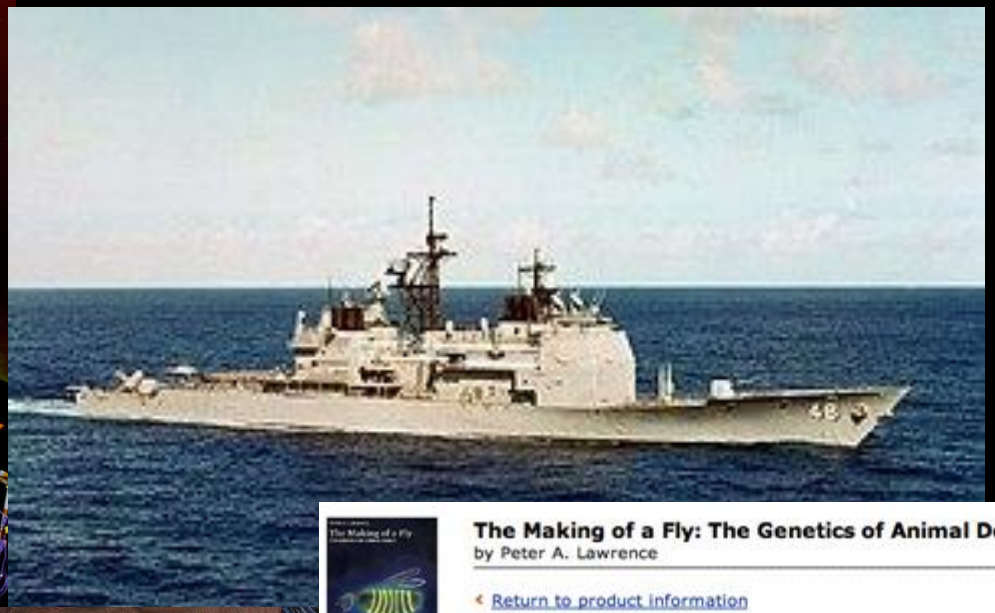
Specify
- Problem and solution
- Customer expectations

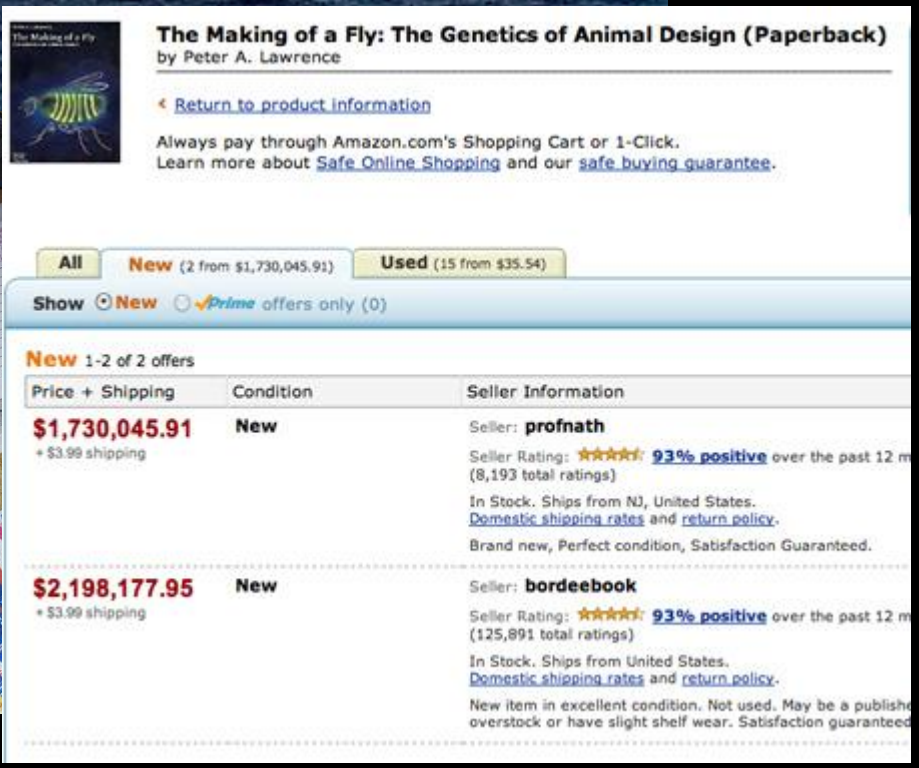
Implement
- Learn tools and technology
- Docs, configs, ...

Test
- Ensure quality

Evolve
- Debug
- Refine







The Making of a Fly: The Genetics of Animal Design (Paperback)
by Peter A. Lawrence

‹ Return to product information

Always pay through Amazon.com's Shopping Cart or 1-Click.
Learn more about Safe Online Shopping and our safe buying guarantee.

All | New (2 from $1,730,045.91) | Used (15 from $35.54)

Show ⊙New ○Prime offers only (0)

New 1-2 of 2 offers

Price + Shipping	Condition	Seller Information
$1,730,045.91 + $3.99 shipping	**New**	Seller: **profnath** Seller Rating: ★★★★★ **93% positive** over the past 12 m (8,193 total ratings) In Stock. Ships from NJ, United States. Domestic shipping rates and return policy. Brand new, Perfect condition, Satisfaction Guaranteed.
$2,198,177.95 + $3.99 shipping	**New**	Seller: **bordeebook** Seller Rating: ★★★★★ **93% positive** over the past 12 m (125,891 total ratings) In Stock. Ships from United States. Domestic shipping rates and return policy. New item in excellent condition. Not used. May be a publishe overstock or have slight shelf wear. Satisfaction guaranteed

COMPLEXITY

“The complexity of software is an esse
property, not an accidental one.”

Continuous world → discrete system
Immateriality of software
Understanding problem domain
Managing development

AGILE MANIFEST

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

AGILE

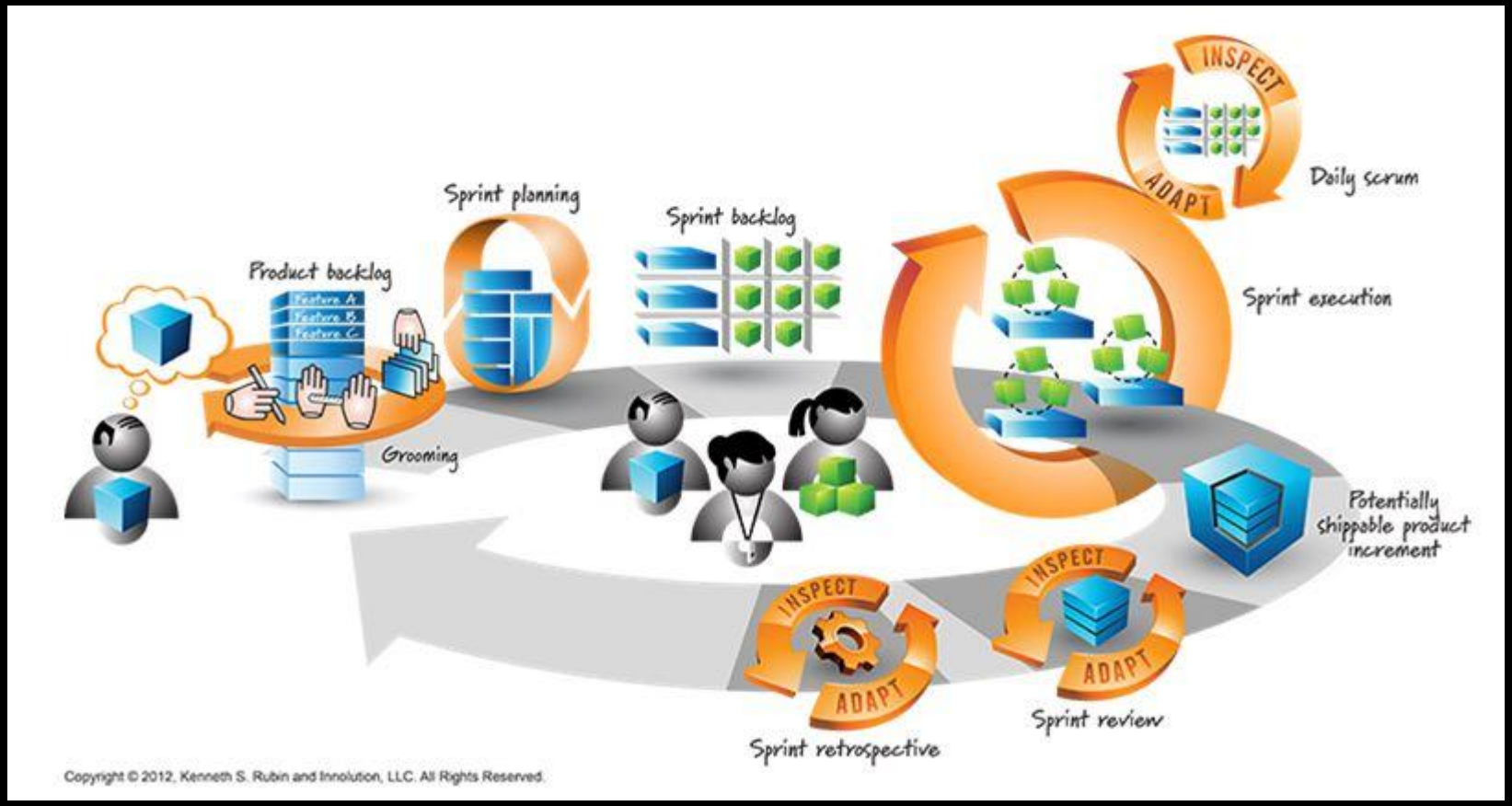
Processes

- Kanban
- XP - eXtreme Programming
- Test-driven development
- Feature-driven development
- Scrum

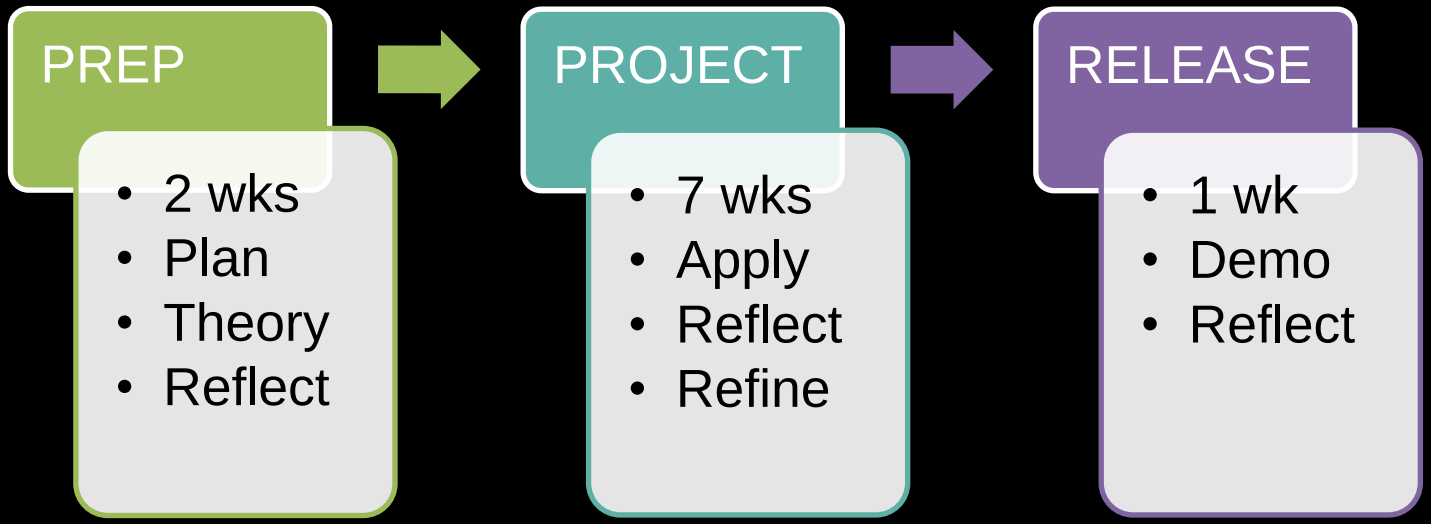
Practices

- User stories
- XFT Cross-functional Teams
- Stand-up meetings
- Short iterations
- Continuous testing
- Coding standards
- Sustainable pace
- Pair programming
- Customer value

SCRUM


INSPECT
ADAPT
Daily scrum
Sprint planning
Sprint backlog
Product backlog
Sprint execution
Grooming
Potentially shippable product increment
INSPECT
ADAPT
Sprint review
INSPECT
ADAPT
Sprint retrospective
Copyright © 2012, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

COURSE OVERVIEW



COURSE PROJECT

http://stmvalidation.eu/about-stm/

REFLECTION

What is
in relation to what might or should be and
includes feedback to reduce the gap

R. Smith. Formative Evaluation and the Scholarship of Teaching and Learning.
New Directions for Teaching and Learning, vol. 88, 2001, pp. 51-62

LEARNING GOALS

Knowledge and understanding
Skills and abilities
Judgement and approach

KNOWLEDGE AND UNDERSTANDING

The student should be able to

- identify the complexities of software design and development
- describe the fundamentals of software engineering, such as stakeholders and requirements
- describe the difference between the Customer, the Solution, and the Endeavour as well as the different methods used for each

Course evaluation 2014:

> “I'd rewrite it as 'Being able to efficiently adapt the codebase to customer requirement changes'.”

SKILLS AND ABILITIES

The student should be able to

- elicitate requirements from and design a solution to a real-world problem
- plan and execute a small software development project in a team
- apply skills from programming courses and other relevant courses in a project-like environment
- learn new tools and APIs on his/her own

~ 20 h/week

Course evaluation 2014:

“A r e y o u k i d d i n g m e ? W e h a d to not only organize the project ourselves, search for information through teachers, supervisors, volvo and the internet (of which only the last seemed to have any constructive answers). We also had had to learn how to make an app for android, from scratch.”

JUDGEMENT AND APPROACH

The student should be able to

- reflect on the choice of software engineering methods used in the project

Course evaluation 2014:

“Scrum was introduce to late and therefor mine group had to change our way to work to late in the course.”

EACH WEEK

Reflect on 16 topics
Document reflection
Upload to team repo
Feedback during supervision
Final iteration is graded, 0-3p / topic

TEAM REFLECTION

Each week:

- your social contract,
  which means you should create one in the first week
- the success criteria for the team in terms of what you want to achieve with your application
- the three KPIs you use for monitoring your progress
- the roles you have used within the team
- relation to literature and guest lectures
  (how do your reflections relate to what others have to say?)

TEAM REFLECTION

Each week:

- the chosen scope of the application under development,
  including priority of features and for whom you are creating value
- your acceptance tests, such as how they were performed and with whom
- the design of your application (choice of APIs, architecture patterns etc.)
- the behavioural overview of your application
  (for instance through use cases, interaction diagrams or similar)
- the structural overview of your application
  (such as class diagrams, domain models or component diagrams)
- your user stories in terms of using a standard pattern, acceptance criteria, task breakdown and effort estimation
- code quality using a tool such as Findbugs
  1 point if your code includes issues concerning correctness or bad style,
  2 points if you have dodgy or performance issues and
  3 points if the code is fine,
  only asses the code you have written yourself

TEAM REFLECTION

Each week:

- the agile practices you have used for the current sprint
- the time you have spent on the course
  (so keep track of your hours so you can describe the current situation)
- the sprint review (either in terms of outcome of the current week's exercise or meeting the product owner)
- best practices for using new tools and technologies
  (IDEs, version control, scrum boards etc.)

TEAM GRADES

Team report 0-48p
Individual reports 0-2p

U: 00 - 20p
3/G: 21 – 30p
4: 31 – 40p
5/VG: 41 - 50p

INDIVIDUAL GRADE

EACH WEEK:

what do I want to learn or understand better?

how can I help someone else, or the entire team, to learn something new?

what is my contribution towards the team's application of scrum ?

what is my contribution t o w a r d s t h e t e a m ' s d e l i v e r i e s ?

INDIVIDUAL GRADE

Upload documentation each week to team repo

One file with a heading for each week

Feedback during supervision

INDIVIDUAL GRADE

Given all weekly assignments are completed:
Based on team grade
+/- for personal contribution

Evidence for active contribution

INDIVIDUAL GRADE

Given all weekly assignments are completed:
Based on team grade
+/- for personal contribution

Evidence for active contribution

PERSONAL CONTRIBUTION

Individually

Total = size(Team) x 10
Score in range(0, Total)

	Eva	Per	Li	Jay	Foo	
Eva	12	5	11	14	8	50
Per	14	14	5	10	7	50
Li	13	12	5	10	10	50
Jay	14	12	5	14	7	50
Foo	15	10	5	13	7	50
	68	51	31	61	39	

Upload using link on course homepage

Code contribution: gitinspector


Plan driven approach creates cost/time estimates
WATERFALL METHOD
Fixed Scope (Requirement)
Quality
Estimated Cost
Estimated Time
Subject to Cost, Time and Quality Risks

REALITY CHECK

What was the purpose of the lecture?
Which learning objectives were covered? How?
What was the relationship to the course assessment?

THIS WEEK

Monday:	First exercise at 15:15 in VasaC
Wednesday:	Scrum Lego exercise at 13:15 in VasaC
Friday:	Scrum lecture at 13:15 in VasaC

https://github.com/hburden/DAT255/blob/master/README.md

NEXT WEEK

Monday: Lecture on Software Quality in VasaC
Wednesday: Project Introduction in VasaC

https://github.com/hburden/DAT255/blob/master/README.md

QA

'Questions don't have to make sense, Vincent', said Miss Susan.

'But answers do'

Terry Pratchett
Thief of Time, 2001