

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Agile Methods

Matthieu Moy

UCBL

2018

Matthieu Moy (UCBL)
Agile Methods
2018
< 1 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Problems and Solutions in Project Mgmt (level 0)

Level 0: no project management ("code and fix"). Maybe OK when working alone, but ...

- Planning impossible
- One developer leaving the team ⇒ failure of the whole project?
- Not reproducible: may work today, but tomorrow?
- ...

Matthieu Moy (UCBL)
Agile Methods
2018
< 2 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Problems and Solutions in Project Mgmt (traditional)

Traditional project: rationalize the process, document as much as possible (V cycle, "say what you do, do what you say")

- Planning easy: all specifications are there, metrics from previous projects too, ...
- One developer leaving the team ⇒ the next ones will read the docs
- Reproducible: just follow the procedures

But ...

- Metrics: 9 pregnant women's question, metrics management vs human being management
- Is following the plan the most clever option if the client changes their mind?
- Is the documentation up to date?
- What if procedures are sub-optimal? How to change them?

Matthieu Moy (UCBL)
Agile Methods
2018
< 3 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Problems and Solutions in Project Mgmt (agile)

Agile: keep in mind that working software is the main goal

- If specifications are broken, change them.
- Doing specification is hard. Discussing a prototype is easier ⇒ Release early. Release often (Eric S. Raymond, The Cathedral and the Bazaar)
- Many traditional project management techniques turn out to have bigger overhead than benefit. Abandon them. ("eliminate waste", lean principle)
- Developers are important. Give them power.

Matthieu Moy (UCBL)
Agile Methods
2018
< 4 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Manifesto for Agile Software Development

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.


<http://agilemanifesto.org/>, 2001.

Matthieu Moy (UCBL)
Agile Methods
2018
< 5 / 45 >


Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Time in Agile Methods


Traditional =




Agile =



Start → Project → End




Start → ... → ... → ...



Matthieu Moy (UCBL)
Agile Methods
2018
< 6 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Who care?



COMPANY EXPERIENCE AND ADOPTION

Company Experience

HOW MANY?

97% The percentage of respondents' organizations that practice agile development methods.

HOW LONG?

The length of time respondents' organizations have been practicing agile development methods:

Time	2017	2016
< 1 year	10%	15%
1-2 years	26%	25%
3-5 years	35%	32%
5+ years	30%	28%

Percentage of Teams Using Agile

52% of respondents stated that more than half of teams in their organizations are using agile practices.

Percentage	Statement
25%	All of our teams are agile
27%	More than 1/2 of our teams are agile
46%	Less than 1/2 of our teams are agile
2%	None of our teams are agile

<https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report>

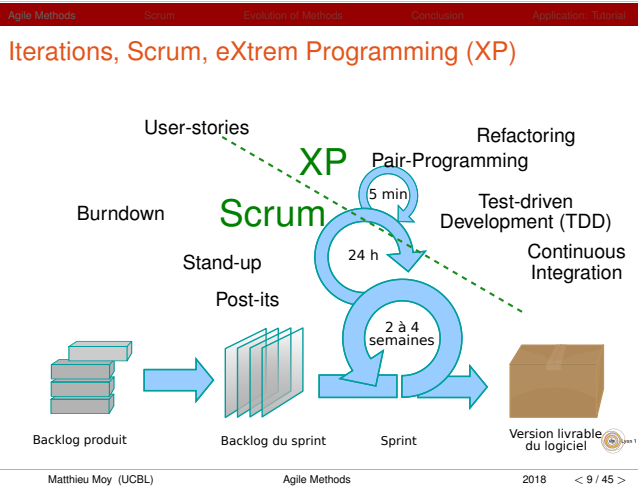
Matthieu Moy (UCBL)
Agile Methods
2018
< 7 / 45 >

Agile Methods
Scrum
Kanban
DevOps
XP
Agile Manifesto

Scrum, Lean, Kanban, DevOps, XP, ...?

- Global trend, many complementary tools or variants:
 - eXtrem Programming:** focused on coding practices (code review, tests, ...)
 - Scrum:** divide a project into iterations, plan each iteration at once
 - Kanban:** ≈ lightweight variant of Scrum
 - Lean:** Global approach (company-wide) to eliminate waste
 - DevOps:** Automate what can be (from development to deployment)

Matthieu Moy (UCBL)
Agile Methods
2018
< 8 / 45 >



Scrum

- Set of practices to organize a team ($\approx 7 \pm 2$ developers)
- Focused on human interactions (inside and outside the team)¹
- Short iterations (called "sprint") & delivery cycle: 1 sprint = 1 week to 1 month
- The most popular in companies today(?)
- Scrum Roles:
 - Product Owner (PO):** discussion with the client
 - Scrum Master:** facilitator (\neq boss), helps the team follow Scrum (or not)
 - Developers:** Write code, take decisions

¹The fact that the vocabulary of team sports is used is not a coincidence

Journey of a feature in Scrum

- 1 Client wants "something"
- 2 Discussion with the PO (Product Owner)
- 3 Agreement on a (set of) user stories ("as a ... I want to ... in order to ...")
- 4 User stories not started = product backlog (PO fills-in the backlog, developers empty it)
- 5 Start of iteration: decide on the sprint backlog
- 6 User stories split into technical tasks
- 7 Each task goes from TODO \rightarrow ongoing \rightarrow Done.
- 8 Demo at the end of iteration
- 9 Release (or not)
- 10 Retrospective and celebrate!

Scrum Board



Scrum Board

- Can be physical (post-its) or virtual (GitLab or GitHub's issues, Trello, ...)
- Virtual: more traceability, multiple geographical sites, ...
- Physical also has benefits:
 - ▶ 1 post-it = 1 unit of information. Text doesn't fit on post-it \Rightarrow split into smaller items.
 - ▶ (Use a felt tip pen, not a thin pen)
 - ▶ Always visible on the wall \Rightarrow you can't claim you forgot!
 - ▶ No remote access \Rightarrow if your boss wants to see the board, she must come in the room.
 - ▶ Flexible (take a pen and draw a line Vs ask the admin of the project to create a column)
 - ▶ Satisfaction of moving post-its to DONE :-)

Role: Product Owner

- Role: discuss product specifications. Must understand the client needs (not necessarily a computer-scientist)
- Preserve developers from direct customer interactions (distraction), but provides as much information as possible to the client.
- Does not take technical decisions.
 - ▶ Example: "Replace technology X with technology Y" is not decided by the PO
 - ▶ "Improve the scalability of the system" can be turned into a user-story and asked by the PO
 - ▶ The team may decide that technology Y is needed to accomplish it.
- Decides on the priority of story (= which one to do first)
- Discusses/negotiates the sprint backlog with developers
- Does not change the sprint backlog during the sprint

Role: Team Member (= developer)

- Role: develop the product.
- Includes development, debugging, testing
- Scrum's ideal: collective code ownership. Each individual may have special skills, but no overspecialization ("this is not my code, I can't modify it").
- Estimate the amount of work for each user-stories (e.g. planning poker):
 - ▶ Use arbitrary time unit: "story points" (\neq man.day), each team may have a different notion of story point.
 - ▶ Too large story \Rightarrow ask the PO to split it
- Split user stories into technical tasks (e.g. "write the HTML", "write CSS", "add entry in DB", ...)

Role: Scrum Master

- Role: facilitator, protect the team against distraction
- Experimented in team management and/or Scrum
- May be a developer, or not
- Make sure everybody work in good condition
- Works for the team, not the other way around

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

Steps of a sprint

- 1 Sprint planning: discuss/agree on the sprint backlog
- 2 Development, continuous testing & integration. Daily scrum meetings.
- 3 Demo
- 4 Retrospective: discuss and improve
- 5 goto 1 (forever?)

Matthieu Moy (UCBL)
Agile Methods
2018
< 17 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

Sprint planning (1 meeting, a few hours)

- Prepared by the PO: product backlog = set of user-stories, sorted by priority
- Evaluation (story points) of first stories
- By experience, 1 sprint = X points \Rightarrow stop when $\sum(\text{story points}) = X$
- Example discussion:
 - Team: we evaluate this story to 40 points.
 - PO: that's too high!
 - Team: that's not your business ...
 - PO: OK, this is high priority but it's too long for now, I'm changing its priority.
- Or:
 - PO: can we reduce the scope of the story to make it fit in 20 points?
 - Team: yes, for example we can make a rough UI and finish the business logic for 20 points.
 - PO: OK, I'm splitting the story, we'll make a nice UI in the next sprint.

Matthieu Moy (UCBL)
Agile Methods
2018
< 18 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

End of sprint planning

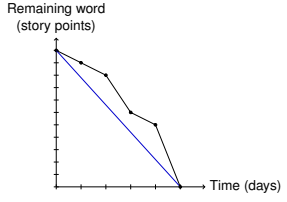
- Team split user-stories (user-spec) into technical tasks
- Some teams evaluate technical tasks in hours of work. Some just split the story points of the story.
- Make a nice scrum board!
- Initialize the burndown

err, what's that?

Matthieu Moy (UCBL)
Agile Methods
2018
< 19 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

The Burndown Chart



- 1 point every day
- Helps medium-term planning (are we late?)
- \Rightarrow helps having constant pressure all along the sprint (\neq "Cool, we're on time" followed by "sleepless night before release")
- Count remaining work, not work done. Why:
 - Task estimated to 7
 - 5 points done
 - 4 point remaining (sorry, we underestimated the task!)
- Hopefully a decreasing function

Matthieu Moy (UCBL)
Agile Methods
2018
< 20 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

Sprint Planning

- Checklist:
 - Sprint planning ready
 - Stories evaluated and divided into technical tasks
 - Burndown initialized (first point)
 - All this is clearly visible (displayed on the wall)
- On your marks, set, ... Go!

Matthieu Moy (UCBL)
Agile Methods
2018
< 21 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

Sprint (development)

- Write "good" code. Not specified by Scrum, but usually:
 - Pair programming
 - TDD
 - Continuous integration
- No compromise on quality (cf. technical debt)
- Daily Scrum (short meeting) every day
- Keep a sustainable but sustained pressure. 40h/week max.

Matthieu Moy (UCBL)
Agile Methods
2018
< 22 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

Daily Scrum, aka "daily stand up"

- Short meeting (\approx 15 min)
- Stand up meeting (\neq everybody reading mail on laptop while one guy talks)
- In front of the Scrum board
- Every developer answers 3 questions:
 - What did I complete last day?
 - What will I complete next?
 - What's blocking me? (\leadsto do I need help?)
- Time-boxed (responsibility of Scrum master). Examples:
 - Scrum master: "Interesting point, but we're getting technical. Can we continue offline?"
 - Scrum master: "Time is out, we're stopping. Tomorrow, let's be quicker so that everybody gets time to talk"
- Move post-its to "ongoing" and "done"
- Update Burndown

Matthieu Moy (UCBL)
Agile Methods
2018
< 23 / 45 >

Scrum
Scrum
Burndown Chart
Sprint Planning
Retrospective
Daily Scrum

"Moving post-its"

- Move from "todo" to "ongoing":
 - Be careful, your problems are in the "ongoing" column.
 - Some methods (e.g. Kanban) limit the number of items in the "todo" column (\leadsto "I can't start this, we need to finish another task before I do").
- Move from "ongoing" to "done":
 - Some team attach a "definition of done" to each story, some have a project-wide definition.
 - "done" is "done": implemented, tested, nothing left to do. 95% of done is not "done".
- Some team add other columns (e.g. "to review by PO")

Matthieu Moy (UCBL)
Agile Methods
2018
< 24 / 45 >

End of Sprint Demo/Review

- Demo of working software to the client and/or PO
- “Hey, look how clever my code is” is not a demo: show the value for the client.
- Encourages end-to-end implementations (e.g. business logic without UI ~> no demo)
- “Demo effect” does not exist: test enough before, not during demo.
- Demo in front of other team: inform other team of what you’re doing, be proud of your work.
- Get feedback (helps for next sprint planning)
- Stories are validated by PO (or not)
- Measure velocity (number of story points validated) => gives an idea for next sprint

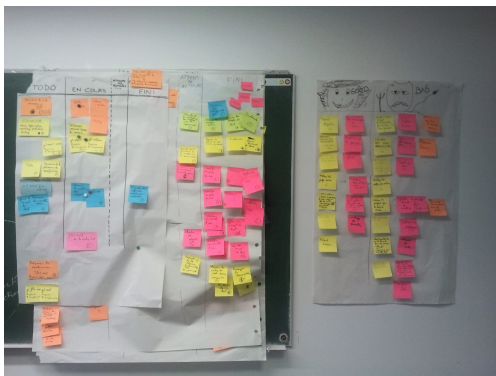


Retrospective

- One meeting: how the sprint went? how to do better?
- Usually in several phases:
 - Information gathering: everybody says/writes what comes to mind
 - Organize ideas (e.g. “good/bad”, “helped us/handicaped us”, ...)
 - Decisions for next sprint (concrete items: checklist if possible)
- Fundamental notion of Scrum and Agile methods: continuous improvement



Retrospective



Continuous Improvement



Celebrate

!



Sprint Planning

Same for next sprint



Steps of a sprint: summary

- 1 Sprint planning
- 2 Development and daily meetings
- 3 Demo
- 4 Retrospective
- 5 goto 1 (forever?)



Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

Scrum Already Has-Been?

DevOps Movement

Waterfall Agile Lean Continuous Integration Continuous Delivery Continuous Deployment Continuous Operations

<https://devops.com/devops-killed-developer-star/>

Matthieu Moy (UCBL) Agile Methods 2018 < 33 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

Trend: Short Cycles

- Traditional project: release every few years
- Scrum: release every few weeks
- DevOps: release every few hours/minutes?

Matthieu Moy (UCBL) Agile Methods 2018 < 34 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

Eliminate Waste

- Scrum is a good tool to improve efficiency
- Still some project management overhead: what's useful? what's waste?
- What's next?

Matthieu Moy (UCBL) Agile Methods 2018 < 35 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

No Silver Bullet

- Consider each method as a tool: use the appropriate tool for the task
 - (But remember, when you have a hammer, everything looks like a nail)
- Don't apply without understanding
 - ▶ The book says "meeting every day", I'm the boss, so do one meeting everyday
- Don't over-interpret
 - ▶ I don't write docs because I'm agile
 - ▶ I can't give you any price because it's Scrum
 - ▶ It's Scrum, there's no boss, it's anarchy
- Adapt
 - ▶ Not too often: it may take time to get the benefits
 - ▶ Often enough: don't keep doing the same mistakes

Matthieu Moy (UCBL) Agile Methods 2018 < 36 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

Recommended Reading

- agilemanifesto.org a few principles
- extremeprogramming.org old site, but still a reference/source of inspiration on many aspects
- "Getting Real" par 37 signals (lean)
- "Lean Software Development: An Agile Toolkit" (Marie & Tom Poppendieck)
- And also: programming-motherfucker.com
- Pleasant to read
- Straight to the point (written in 1 week-end)
- Pragmatic
- French or English
- Another classic
- More complete

Matthieu Moy (UCBL) Agile Methods 2018 < 37 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

5 minutes must see

Scrum Master - Funny movie about The Power of Scrum

The power of Scrum,
<https://www.youtube.com/watch?v=P6v-I9VvTq4>

Matthieu Moy (UCBL) Agile Methods 2018 < 38 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

Club Agile Rhône Alpes

- <https://www.clubagilerhonealpes.org/>
- Coding Dojo
- Conferences ("Agile Grenoble" = ♥, "Agile Lyon" = probably good too ;-))
- Agile Play Ground

Matthieu Moy (UCBL) Agile Methods 2018 < 39 / 45 >

Agile Methods Scrum Evolution of Methods Conclusion Application Tutorial

TD: Lego4ScrumPaper4Scrum

- Idée : construire une ville en papier en appliquant Scrum
- Backlog :
 - ▶ 1 lotissement de 5 bâtiments sans étage
 - ▶ 10 bâtiments à un étage
 - ▶ Un magasin
 - ▶ Une école
 - ▶ Un hôpital
 - ▶ Une maternelle
 - ▶ Un arrêt de bus
 - ▶ Un jardin public
 - ▶ Un pont
 - ▶ Un statue de Douglas Engelbart
 - ▶ Un aéroport
 - ▶ Une mairie
 - ▶ Une station essence

Matthieu Moy (UCBL) Agile Methods 2018 < 40 / 45 >

Agile Methods

Scrum


Estimation & Backlog

Planning

Application Tutorial

Déroulement

- Équipes de 6-8 personnes
- Préparation du backlog
- 3 mêlées (time-box)
- Ranger ;-)



Matthieu Moy (UCBL)Agile Methods2018< 41 / 45 >

Agile Methods

Scrum

Estimation & Backlog


Planning

Application Tutorial

Préparation des équipes

3 min

- Organiser les équipes (6-8 personnes)
- Qui est scrum master?
- Un étudiant peut être PO, sinon c'est votre enseignant
- Préparer la salle (3 tables/équipe)



Matthieu Moy (UCBL)Agile Methods2018< 42 / 45 >

Agile Methods

Scrum

Estimation & Backlog


Planning

Application Tutorial

Estimation, backlog produit

7 min

- Estimer et prioriser chaque story
- Définition de “fait” ?



Matthieu Moy (UCBL)Agile Methods2018< 43 / 45 >

Agile Methods

Scrum


Estimation & Backlog

Planning

Application Tutorial

3 mêlées

- Sprint planning : 3 min
- Sprint : 7 min
- Demo/revue : 3 min
- Retrospective : 2 min



Matthieu Moy (UCBL)Agile Methods2018< 44 / 45 >

Agile Methods

Scrum


Estimation & Backlog

Planning

Application Tutorial

La fin ...

- On nettoie autour de chaque groupe
- On range la salle
- Retour sur l'activité et l'application de SCRUM:
 - Plus de ...
 - Moins de ...
 - Arrêter ...
 - Commencer ...
 - Garder ...



Matthieu Moy (UCBL)Agile Methods2018< 45 / 45 >

