Ingegneria del Software

Corso di Laurea in Informatica per il Management

Scrum

Davide Rossi

Dipartimento di Informatica

Università di Bologna

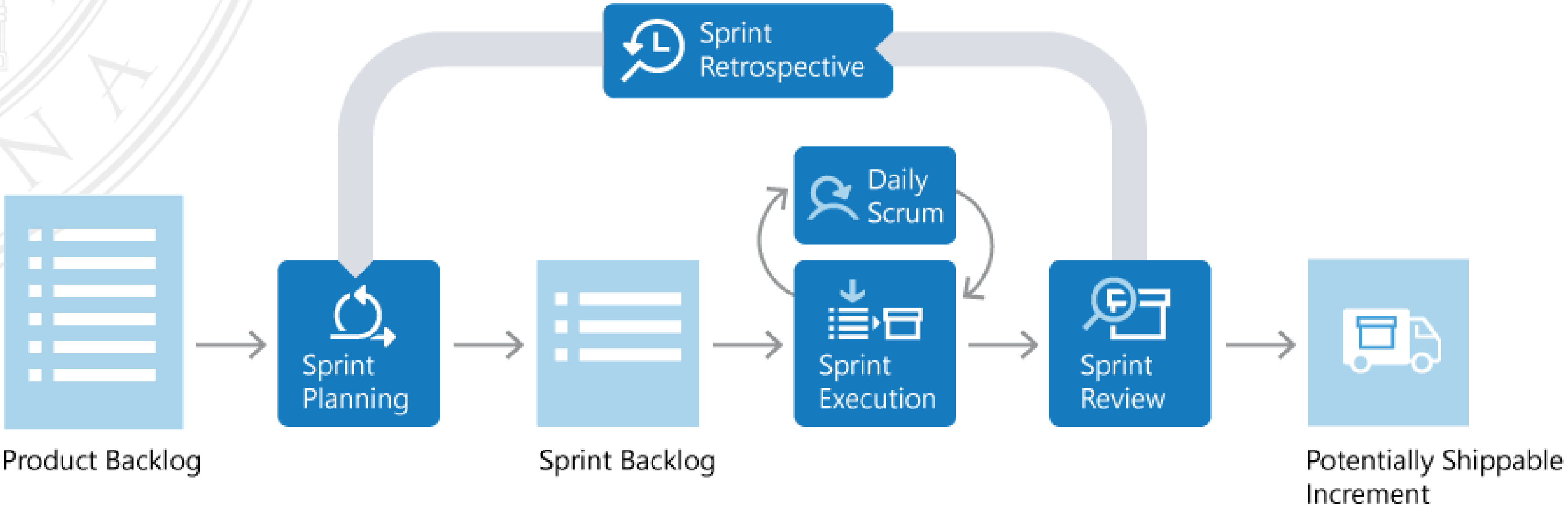
What is Scrum

- Scrum: A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value
- Can be used in different contexts, software product development is one of the many
- But it is the main one

Scrum and agile

- Scrum is part of the agile world. It is not a full development method, however: the main focus is on project management
- Scrum can (and usually is) adopted WITH other development methods (integrating or replacing project-related practices)

Scrum lifecycle



Sprint

- Scrum projects progress as a sequence of sprints
- A sprint is a time-boxed iteration
- Includes full design/code/test cycles
- Ends with a potentially shippable increment (PSI)

Roles, events, artifacts

Scrum is characterized by

- Roles
- Artifacts
- Events

Scrum roles

- Core (committed a.k.a. pigs)
  - Product owner
  - Scrum Master
  - Development Team
- Additional (involved a.k.a. chickens)
  - Customers
  - Executive management

Product owner

- Represents the stakeholders within the project
- Decides priorities, deadlines, and features
- Accepts or rejects work

Development team

- Self-organizing (work items are pulled, not assigned)
- 5-9 people
- Cross functional
- Full time

Scrum Master

- A *servant leader*
- A facilitator (removes impediments)
- Responsible for correct application of scrum principles and practices
- Supports the team members, usually chairs meetings

Artifacts

- Product backlog
- Sprint backlog
- Burn down chart

Product backlog

The Product backlog is an ordered list (sorted by the product owner) of *things* that have to be done:

- Functional requirements (usually in the form of user stories)
- Bug fixes
- Non-functional requirements
- Technology-related requirements
- Chores (items producing value for the team but not directly for the stakeholders)

Stories, epics, themes

- Stories describing high level features are usually collected early but are underspecified and are refined as the project progresses
- Epics are large user stories; they usually need more than one sprint to be fully developed
- Epics are split in smaller, more detailed stories when they climb up the backlog
- Themes are collection of related stories

Sprint backlog

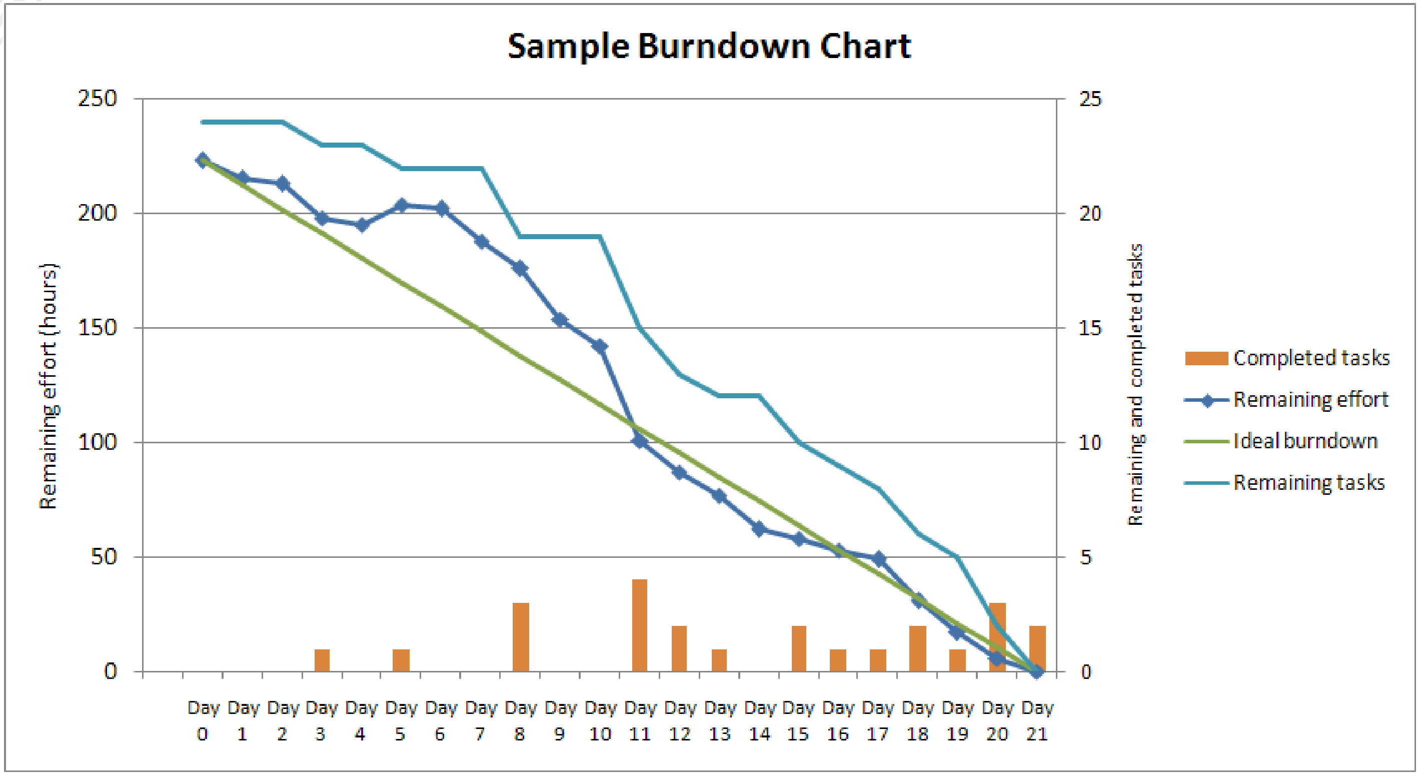
- List of ***tasks*** derived from the top-most product backlog items
- Determined by the development team
- Has to be completed in the current sprint
- Each task is associated to an effort (hours required to complete the task); effort should be less than one work day. If larger then split it.

Tasks

Audio
Community & Marketing
Art
Engineering
Production
IN PROG
DONE
BLOCKED
IF you add a task(s) after scrum, put here
BACKLOG

Burn down chart

Publicly displayed chart showing remaining work in the sprint backlog



Events

- Sprint planning
- Daily scrum
- Sprint review
- Sprint retrospective

Sprint planning

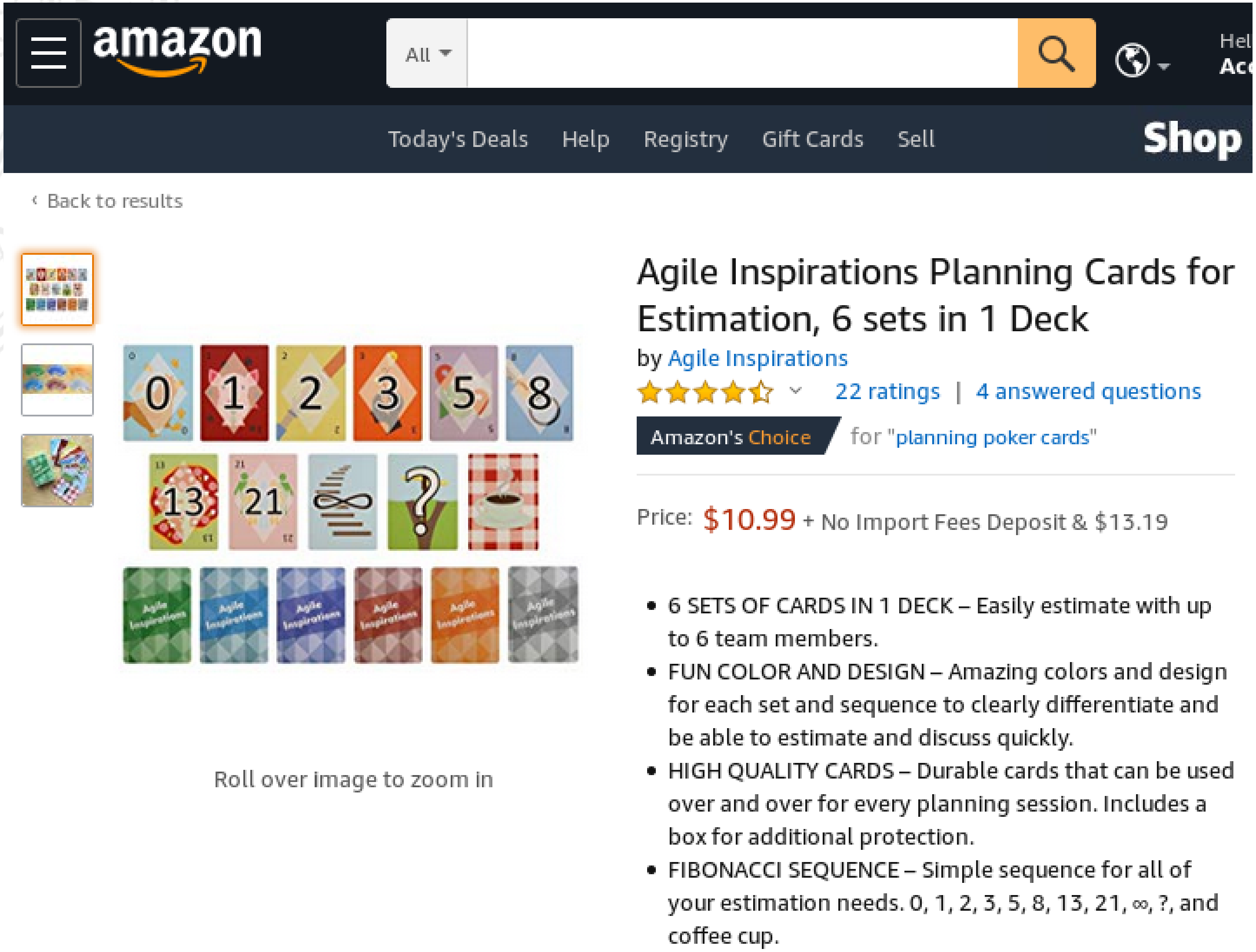
- One day, all pigs
- Two phases

1. The product owner defines a goal and presents the topmost related items from the product backlog, and each is discussed it order to detail it and estimate the required effort
2. (Team only) Selected items are broken down into tasks and the sprint backlog is populated

Scrum estimation

- Tasks: hours
- User stories: *story points*
  Story points are a measure of the enhanced product value of a user story (story points usually progress as a Fibonacci sequence); assigned with *planning poker*

Yes, really



Agile Inspirations Planning Cards for Estimation, 6 sets in 1 Deck

by Agile Inspirations

22 ratings | 4 answered questions

Amazon's Choice for "planning poker cards"

Price: $10.99 + No Import Fees Deposit & $13.19

- 6 SETS OF CARDS IN 1 DECK – Easily estimate with up to 6 team members.
- FUN COLOR AND DESIGN – Amazing colors and design for each set and sequence to clearly differentiate and be able to estimate and discuss quickly.
- HIGH QUALITY CARDS – Durable cards that can be used over and over for every planning session. Includes a box for additional protection.
- FIBONACCI SEQUENCE – Simple sequence for all of your estimation needs. 0, 1, 2, 3, 5, 8, 13, 21, ∞, ?, and coffee cup.

Scrum estimation

- Capacity-driven planning
    - Capacity (in terms of hours) is used to pull stories from the product backlog
- Velocity-driven planning
    - Velocity is used to pull stories from the product backlog
    - Velocity is the measure of credits earned in a sprint where credits depend on story points

Daily scrum

- 15 minutes (stand-up) meeting
- Development team members answer three questions
  - What did I do yesterday that helped the Development Team meet the Sprint Goal?
  - What will I do today to help the Development Team meet the Sprint Goal?
  - Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?

Sprint review and retrospecting

- At the end of a sprint
- Review
  - Whole team + stakeholders
  - Increment is presented along with problems and solutions
  - Product backlog is discussed; timeline, budget and capabilities are reviewed
- Retrospecting
  - Scrum Master + development team
  - Improvement of the process for the next sprints are discussed

Scaling

- Scaling can be achieved via hierarchical scrum teams
- Specific events are scheduled to ensure overall progress (e.g. The Scrum of scrums after the daily meeting)
- LeSS (large scale scrum) is a framework intended for many teams working together on one product. It is proposed by C. Larman and B. Vodde.

Scrum considered harmful

- Flaccid scrum - https://martinfowler.com/bliki/FlaccidScrum.html
- Dark scrum - https://ronjeffries.com/articles/016-09ff/defense/
- Zombie scrum - https://medium.com/zombie-scrum-resistance

Kanban

- Kanban is a lean scheduling method to control a production chain for just-in-time production
- Kanban is Japanese work meaning billboard
- Introduced in the software development domain by David Anderson. Open Kanban is an open source version of Kanban for agile software development

Kanban principles

- Start with existing process
  - Kanban is a change management method that stimulates continuous, incremental and evolutionary changes
- Agree to pursue incremental, evolutionary change
- Respect the current process, roles, responsibilities and titles
- Leadership at all levels

The pipeline

- Kanban uses a continuous delivery approach (no cycles).
- All work is split in units and flows through a pipeline composed of stages.

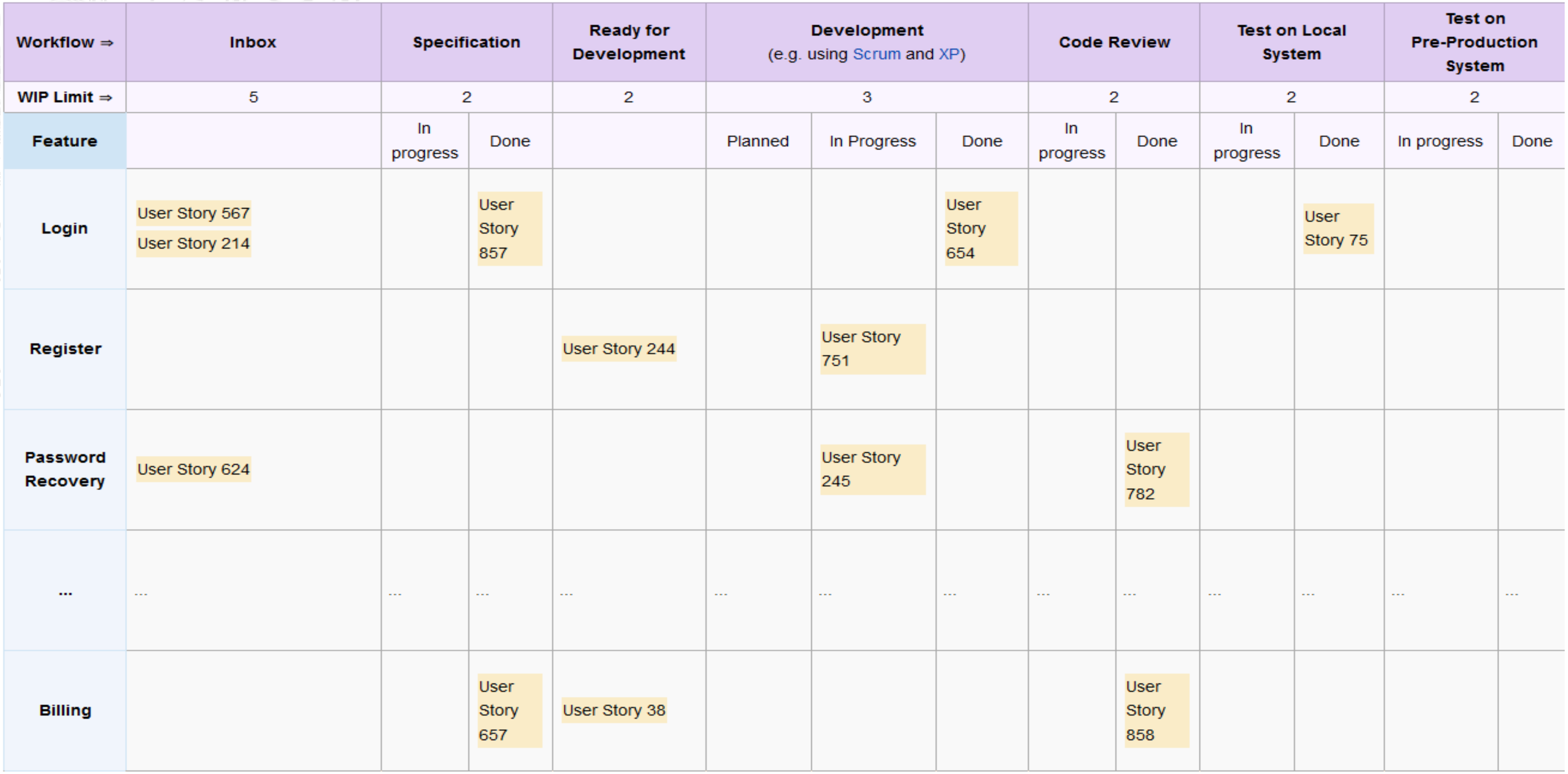
Visualization

- Visualization is a central concept in Kanban
- A Kanban board is a central artifact used to visualize the status of the work items and their progess (the pipeline)

Limit WIP

- Kanban tries to match the work-in-progress (WIP) with team's capacity
- To accomplish this, the stages of the pipeline are bounded to a limit
- The process progresses by moving work items through the pipeline, form “to do” to “done”, respecting the limits
- Cycle time is the basic metric used to evaluate the team progress

Board, progress, limits

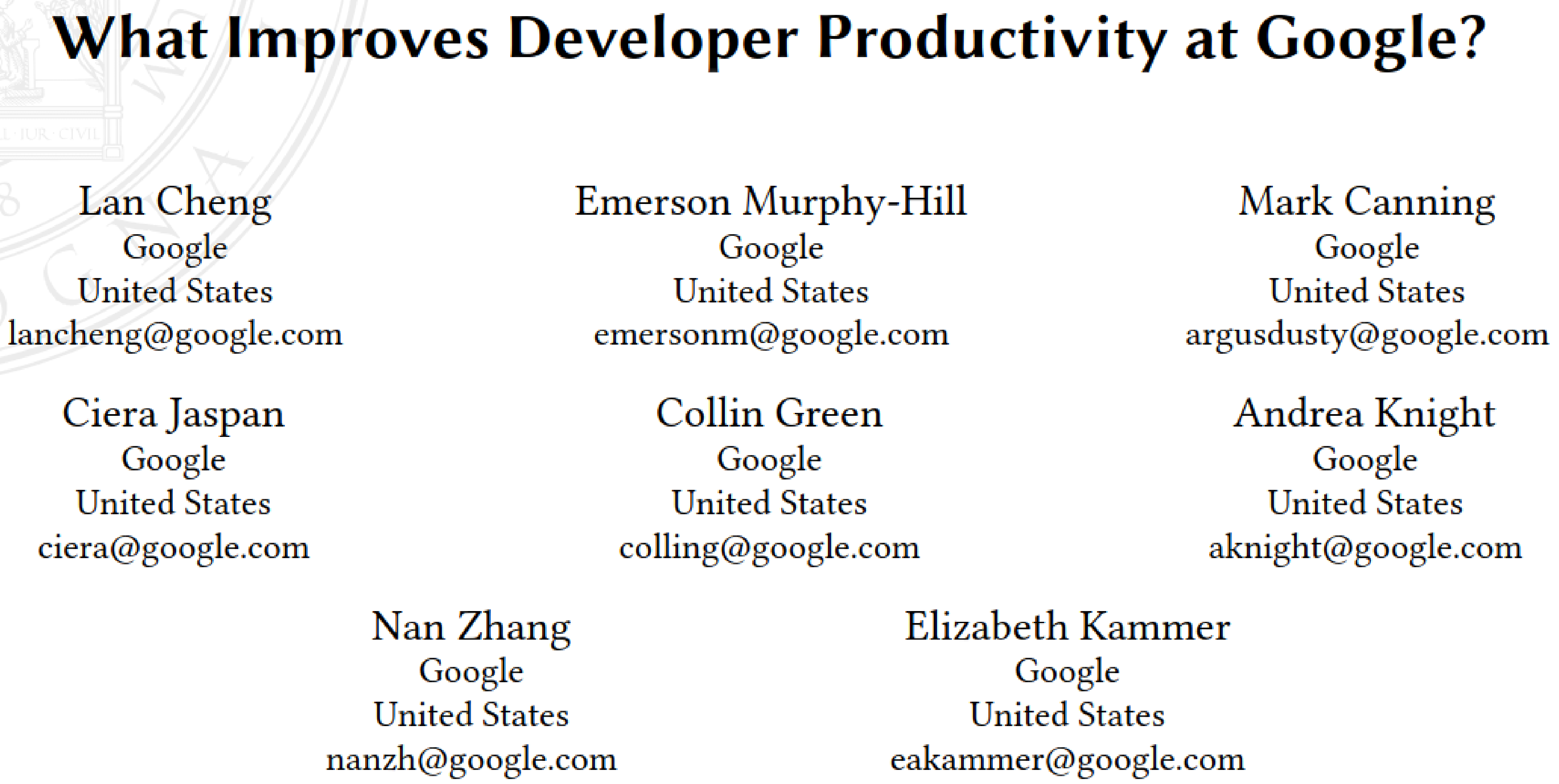


Workflow ⇒	Inbox	Specification		Ready for Development	Development (e.g. using Scrum and XP)			Code Review		Test on Local System		Test on Pre-Production System	
WIP Limit ⇒	5	2		2	3			2		2		2	
Feature		In progress	Done		Planned	In Progress	Done	In progress	Done	In progress	Done	In progress	Done
Login	User Story 567 User Story 214		User Story 857				User Story 654				User Story 75		
Register				User Story 244		User Story 751							
Password Recovery	User Story 624					User Story 245			User Story 782				
...	...	...	...	...	...	...	...	...	...	...	...	...	...
Billing			User Story 657	User Story 38					User Story 858				

Agile methods and technical excellence

- The agile manifesto was about developers taking control of the process.
- Nowadays we have *agile project managers.*
- Technical aspects are often neglected, the focus is on the process. This is not what agile was about.
- Technical excellence still is the main driver of productivity.

The relevance of technical excellence



What Improves Developer Productivity at Google?

Lan Cheng
Google
United States
lancheng@google.com

Emerson Murphy-Hill
Google
United States
emersonm@google.com

Mark Canning
Google
United States
argusdusty@google.com

Ciera Jaspan
Google
United States
ciera@google.com

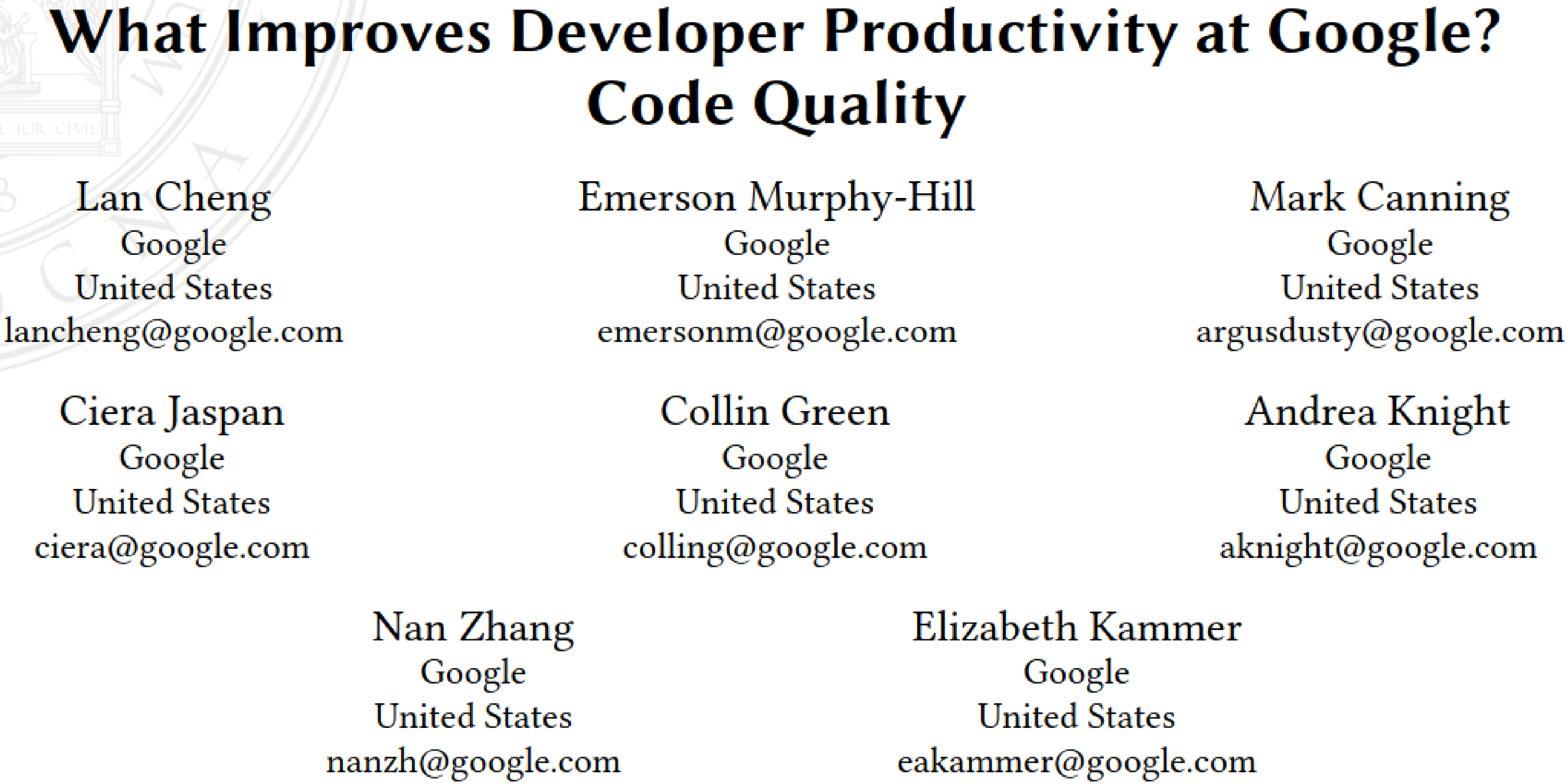
Collin Green
Google
United States
colling@google.com

Andrea Knight
Google
United States
aknight@google.com

Nan Zhang
Google
United States
nanzh@google.com

Elizabeth Kammer
Google
United States
eakammer@google.com

The relevance of technical excellence



What Improves Developer Productivity at Google? Code Quality

Lan Cheng
Google
United States
lancheng@google.com

Emerson Murphy-Hill
Google
United States
emersonm@google.com

Mark Canning
Google
United States
argusdusty@google.com

Ciera Jaspan
Google
United States
ciera@google.com

Collin Green
Google
United States
colling@google.com

Andrea Knight
Google
United States
aknight@google.com

Nan Zhang
Google
United States
nanzh@google.com

Elizabeth Kammer
Google
United States
eakammer@google.com