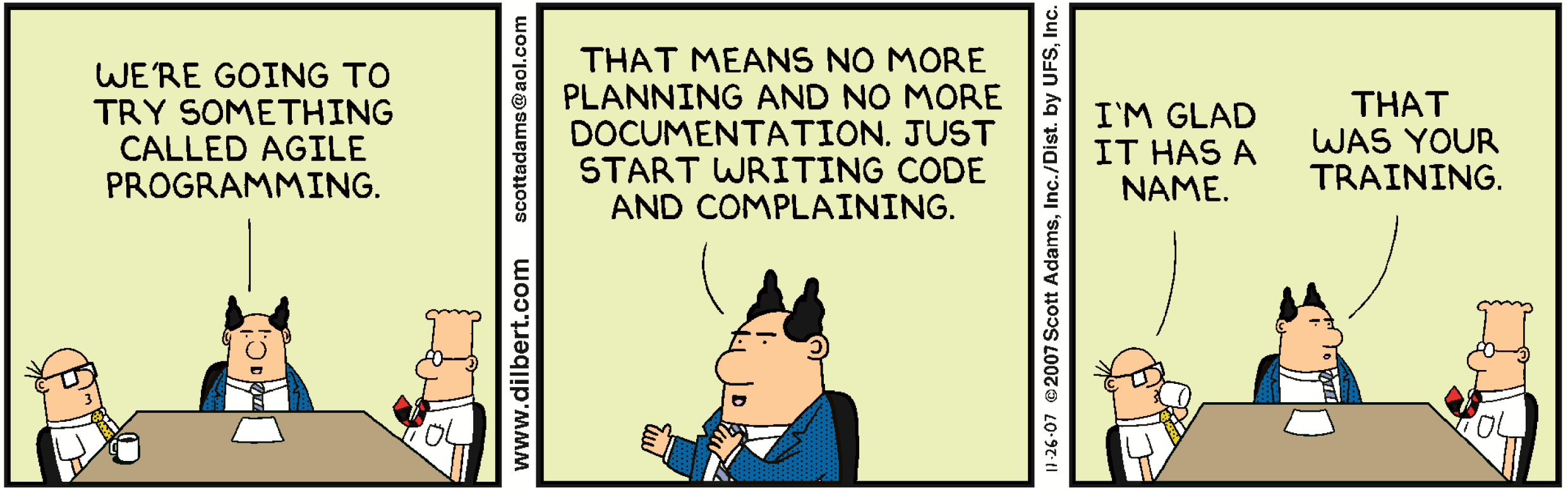
Agile Training



Recap

Start-up / Initiation

PMBOK *vs* **PRINCE2**

Planning

Scope: WBS

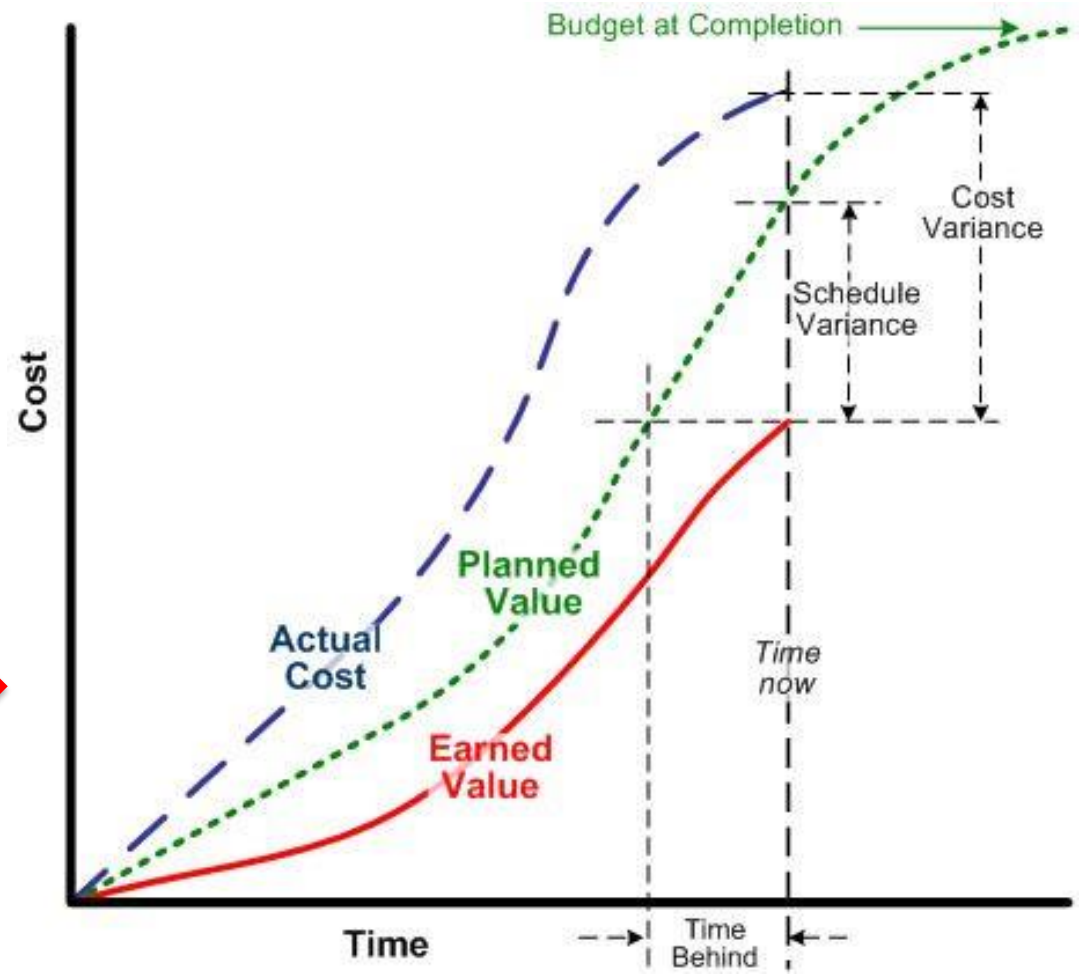
Time: Gantt, PND, CPM, PERT

Cost:

Earned Value Analysis (EVA)

Benefits:

Key Performance Indicators (KPIs)



● REC

This session is being recorded.

WARWICK

CS352 Project Management for Computer Scientists

6. Lean and Agile

Part 1

interact at:

warwick.ac.uk/pm4cs/**6**

Dr Ian Saunders

warwick.ac.uk/pm4cs/6

POLL OPEN

Homework: Best/Worst Performing WPs

Best @ schedule, Worst @ schedule, Best @ Cost, Worst @ Cost

WP	Budget	Actual % Done	Planned % Done	Actual Cost (AC)	Earned Value (EV)	Planned Value (PV)
A	£30,225	93%	100%	£30,458	£28,109	£30,225
B	£30,720	84%	100%	£23,055	£25,805	£30,720
C	£15,300	83%	100%	£6,999	£12,699	£15,300
D	£44,828	85%	83%	£37,571	£38,104	£37,207
E	£10,409	75%	70%	£7,817	£7,807	£7,286
F	£8,890	98%	100%	£6,138	£8,712	£8,890
G	£3,458	90%	100%	£2,736	£3,112	£3,458
H	£51,092	10%	20%	£8,347	£5,109	£10,218
I	£1,404	44%	30%	£432	£618	£421
J	£1,258	25%	30%	£756	£315	£377
K	£10,373	60%	60%	£2,904	£6,224	£6,224
L	£4,531	11%	15%	£72	£498	£680
M	£59,186	0%	0%	£0	£0	£0

1. D, H, C, H
2. H, I, J, L
3. I, H, L, J
4. H, D, H, C
5. I, H, C, H

Homework: Best/Worst Performing WPs

WP	Budget	Actual % Done	Planned % Done	Actual Cost (AC)	Earned Value (EV)	Planned Value (PV)	Schedule Variance (SV)	Cost Variance (CV)	Schedule Perf. Index (SPI)	Cost Perf. Index (CPI)	Cost Schedule Index (CSI)
A	£30,225	93%	100%	£30,458	£28,109	£30,225	-£2,116	-£2,349	0.93	0.92	0.86
B	£30,720	84%	100%	£23,055	£25,805	£30,720	-£4,915	£2,750	0.84	1.12	0.94
C	£15,300	83%	100%	£6,999	£12,699	£15,300	-£2,601	£5,700	0.83	1.81	1.51
D	£44,828	85%	83%	£37,571	£38,104	£37,207	£897	£533	1.02	1.01	1.04
E	£10,409	75%	70%	£7,817	£7,807	£7,286	£520	-£10	1.07	1.00	1.07
F	£8,890	98%	100%	£6,138	£8,712	£8,890	-£178	£2,574	0.98	1.42	1.39
G	£3,458	90%	100%	£2,736	£3,112	£3,458	-£346	£376	0.90	1.14	1.02
H	£51,092	10%	20%	£8,347	£5,109	£10,218	-£5,109	-£3,238	**0.50**	0.61	0.31
I	£1,404	44%	30%	£432	£618	£421	£197	£186	**1.47**	1.43	2.10
J	£1,258	25%	30%	£756	£315	£377	-£63	-£442	0.83	**0.42**	0.35
K	£10,373	60%	60%	£2,904	£6,224	£6,224	£0	£3,320	1.00	2.14	2.14
L	£4,531	11%	15%	£72	£498	£680	-£181	£426	0.73	**6.92**	5.08
M	£59,186	0%	0%	£0	£0	£0	£0	£0	1.00	1.00	1.00

Which of these formulae forecasts the estimate at completion by assuming remaining work will be completed with the same overall efficiency as work done so far (CSI)?

1 BAC÷CSI

2 AC + BAC – EV

3 AC + BAC – (EV÷CSI)

4 AC + (BAC–EV)÷CSI

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Lean

lean

/liːn/

adjective

1. having no superfluous fat
   - *"Ruth had a lean, muscular body”*
   - Slim, slender, healthy
   - (Of meat) not fatty
   - Not in excess

A bit of History

Ford Model T - 1926

Toyota AA - 1936

A bit of History

Ford Model T - 1926

- The “perfect product”
- Remove **cost** from manufacturing process
- **Large** batches, economy of **scale**

- “Seek perfection” – maximise **value** (to customer)
- Remove **waste** from manufacturing process
- **Small** batches, more **efficient** pipeline (flow)

Toyota AA - 1936

A bit of History

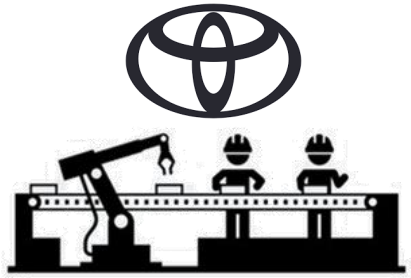
Ford Model T - 1926

- The “perfect product”
- Remove **cost** from manufacturing process
- **Large** batches, economy of **scale**

“any colour car you want
– as long as it is black”

- “Seek perfection” – maximise **value** (to customer)
- Remove **waste** from manufacturing process
- **Small** batches, more **efficient** pipeline (flow)

Toyota AA - 1936



7 Lean Manufacturing Principles

Toyota (2001)

1. Eliminate waste
2. Amplify learning
3. Defer Commitment
4. Deliver Fast
5. Empower the team
6. Build integrity in
7. See the whole

Toyota's "lean manufacturing principles" (2001)

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

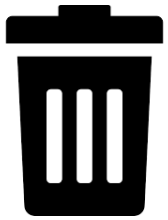
Three Wastes

Muri 無理
Overburden
Excessiveness
Unreasonableness

- People can't work at an unsustainable pace
- Machines can't work at an unsustainable pace

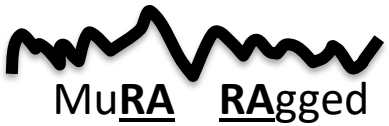
Muda 無駄
Wastefulness
Uselessness
No added value

- An activity that consumes resources without creating value for the customer



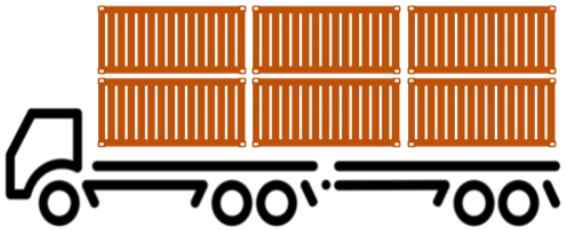
Mura 斑
Unevenness
Irregularity
Inequality

- Working hard in the runup to a deadline, doing nothing the week after
- Uneven schedule
- Variability in "flow"



Three Wastes

MURI
Overburdened



MURA
Fluctuation

MUDA
Waste

✓ **OPTIMUM**

https://www.acd.com.tr/en/what-is-muri-mura-muda

Three Wastes Quiz

warwick.ac.uk/pm4cs/6 POLL OPEN

Commuting to the office

Mass-production, for economy of scale

Moving between lecture theatres

Doing an all-nighter to meet deadline

Occasional defects occur in production

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6 POLL OPEN

Mass-production, for economy of scale

Moving between lecture theatres

Doing an all-nighter to meet deadline

Occasional defects occur in production

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6

POLL OPEN

Moving between lecture theatres

Doing an all-nighter to meet deadline

Occasional defects occur in production

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6

POLL OPEN

Doing an all-nighter to meet deadline

Occasional defects occur in production

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6

POLL OPEN

Occasional defects occur in production

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6

POLL OPEN

Zero-hours contracts

(overburden)

(waste)

(unevenness)

Three Wastes Quiz

warwick.ac.uk/pm4cs/6

(overburden)

(waste)

(unevenness)

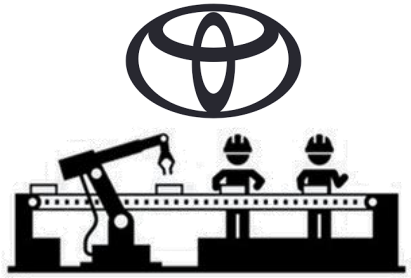
Leaderboard

Position	Participants	Score

Eliminate Waste

1. **Transport:** In excess of what is required to meet customer demand.
2. **Inventory:** Materials, products or resources in excess of customer demand.
3. **Motion:** The movements between process steps.
4. **Waiting:** Waiting on product, people or machines.
5. **Over Processing:** Processing in excess of what is required.
6. **Overproduction:** producing in excess of customer demand.
7. **Defects:** passing poor quality down the supply chain.
8. **Skills**: Not seeking out the expertise or creativity of your own people

Taiichi Ohno, Toyota motor company, 1943



7 Lean Manufacturing Principles

Toyota (2001)

1. Eliminate waste *(muda, muri, mura)*
2. Amplify learning
3. Defer Commitment *(lower inventory)*
4. Deliver Fast *(reduce waiting, transport)*
5. Empower the team *(utilise skills)*
6. Build integrity in *(defects are wasteful!)*
7. See the whole

Toyota's "lean manufacturing principles" (2001)

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Pull

PUSH

PULL

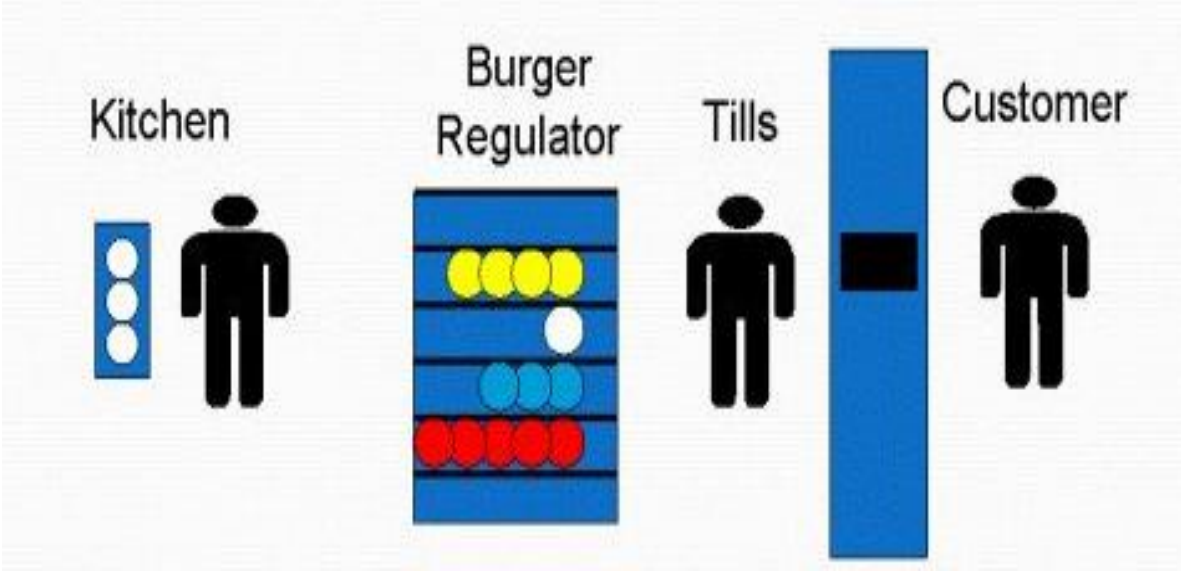
Pull

PUSH

PULL

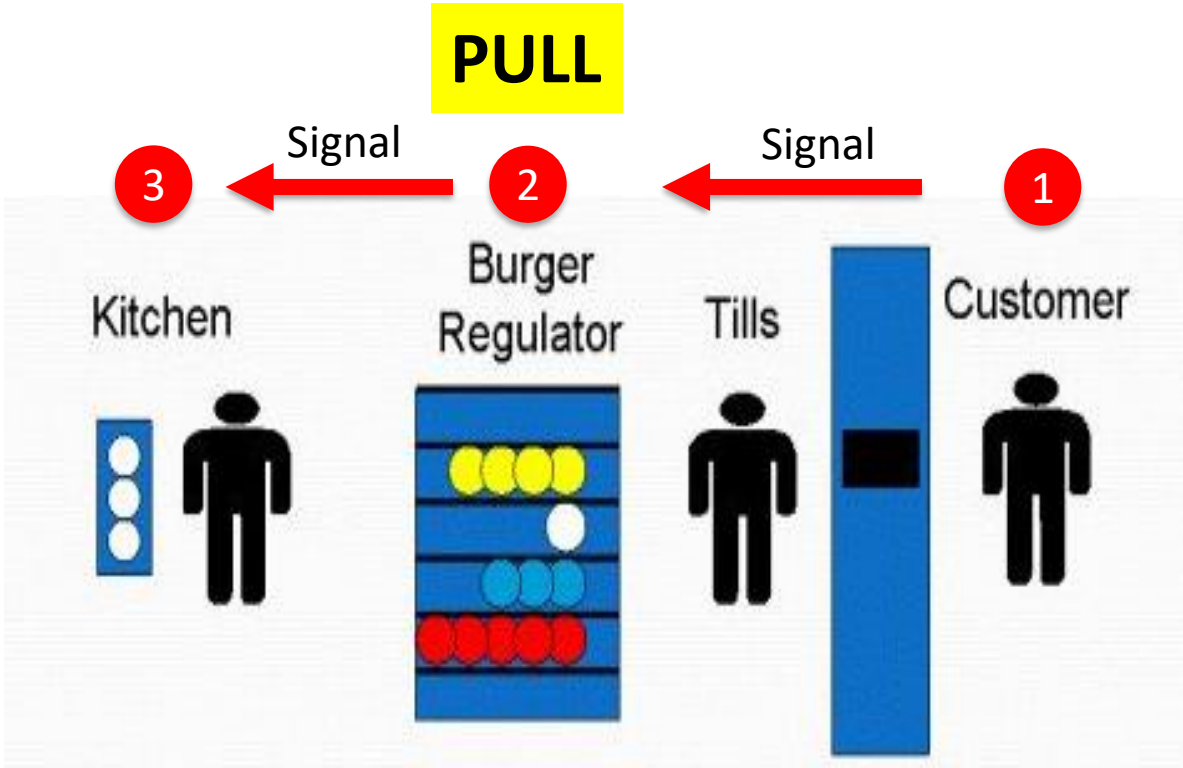
Muri (overburden) – working harder than we need to
Mura (unevenness) – inventory, overproduction
Muda (useless) – if demand changes

Kanban



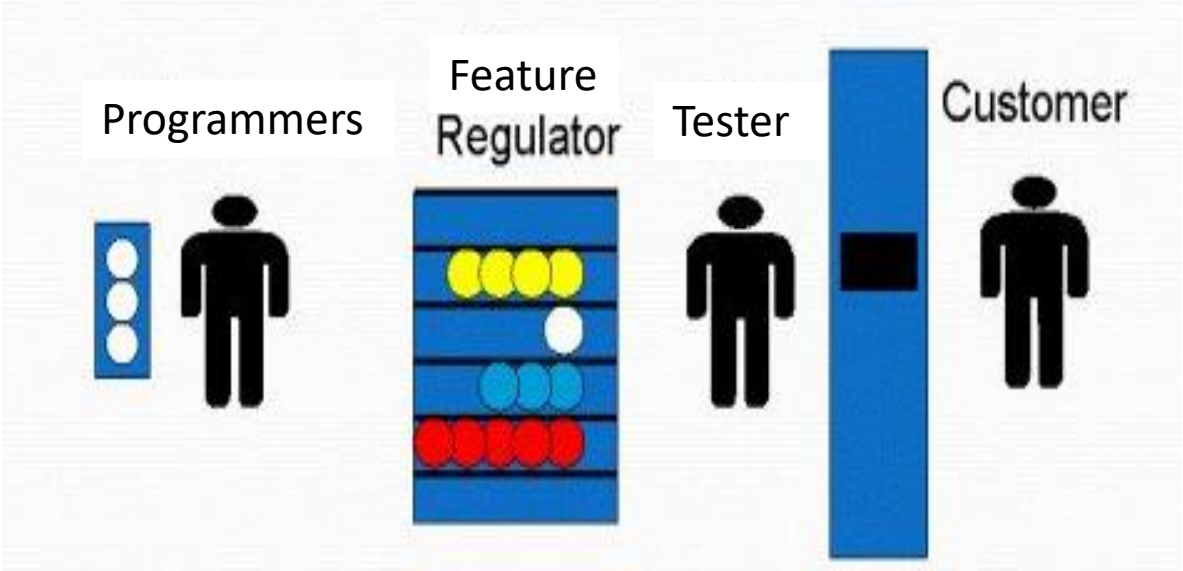
- As orders come in, burgers removed from regulator
- Replenished on-demand (JIT) by kitchen
- Not made to a forecast and pushed at customer!

Kanban



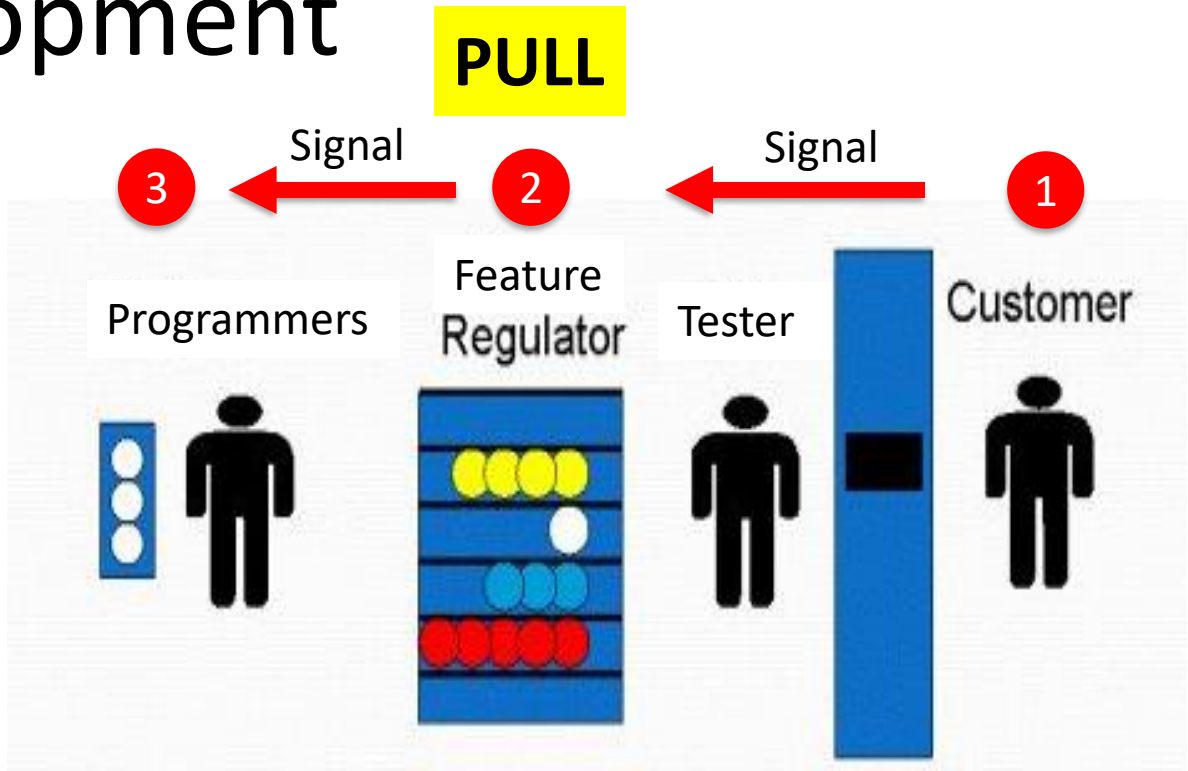
- As orders come in, burgers removed from regulator
- Replenished on-demand (JIT) by kitchen
- Not made to a forecast and pushed at customer!

Kanban Software Development

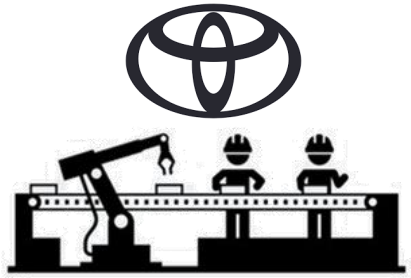


- As **work completed**, taken from regulator for **testing**
- **Work done** on-demand (JIT) by **programmers**
- Not made to a **requirements document** and pushed at customer!

Kanban Software Development



- As **work completed**, taken from regulator for **testing**
- **Work done** on-demand (JIT) by **programmers**
- Not made to a **requirements document** and pushed at customer!



7 Lean Manufacturing Principles

Toyota (2001)

1. Eliminate waste *(on demand)*
2. Amplify learning
3. Defer Commitment *(JIT)*
4. Deliver Fast *(maximise flow)*
5. Empower the team *(pull)*
6. Build integrity in
7. See the whole *(Kanban board)*

Toyota's "lean manufacturing principles" (2001)

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Minimum Viable Product (MVP)

Minimum Viable Product (MVP)

- No more than necessary
- No waste
- Lower cost
- Faster delivery

- Valuable
- Useful
- Beneficial

- A “Deliverable”
- Focus on customer

Minimum Viable Product (MVP)

- No more than necessary
- No waste
- Lower cost
- Faster delivery

- Valuable
- Useful
- Beneficial

- A “Deliverable”
- Focus on customer

Maximise flow of **value** (to customer)

Minimum Viable Product (MVP)

- No more than necessary
- No waste
- Lower cost
- Faster delivery

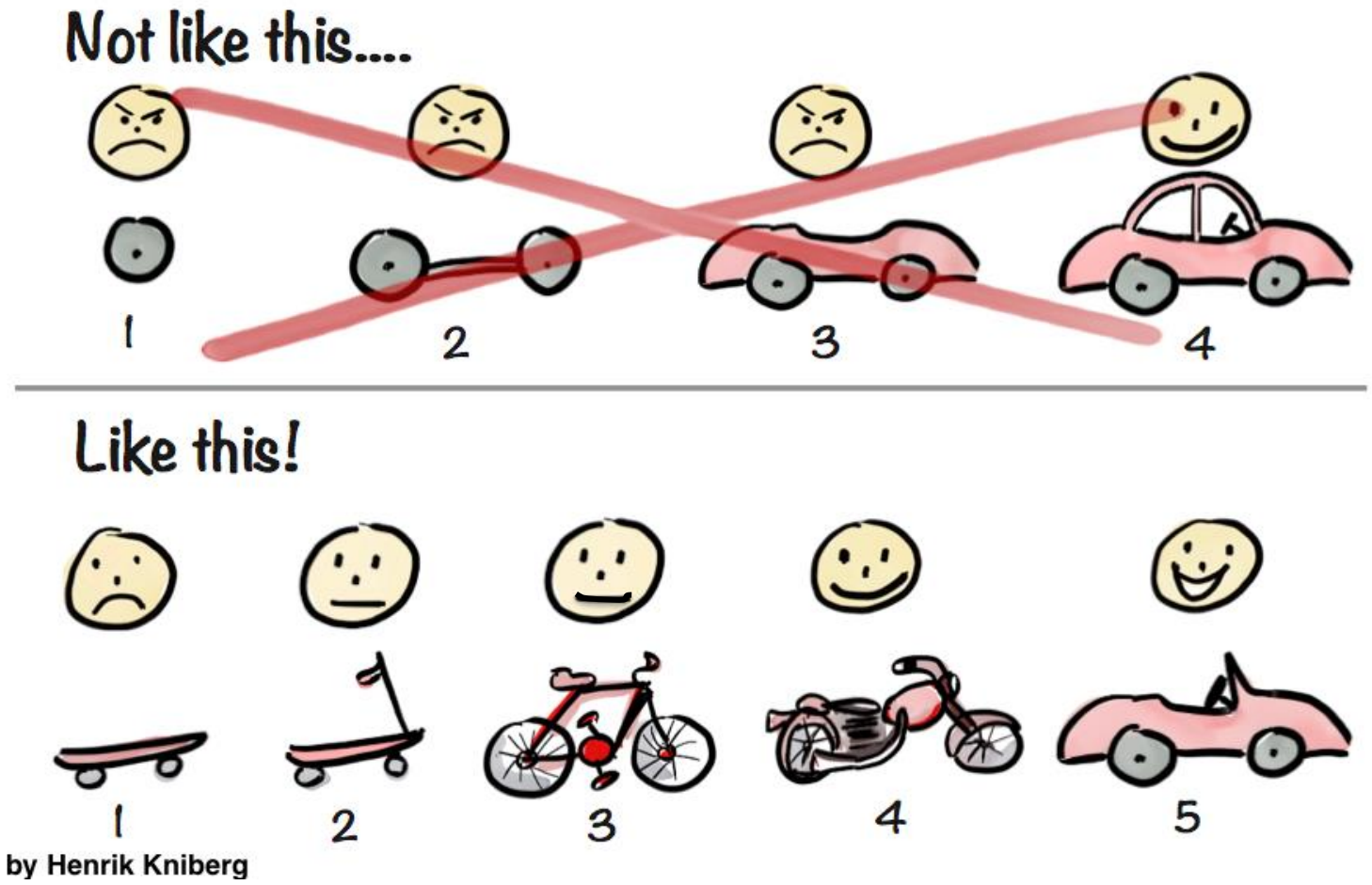
- Valuable
- Useful
- Beneficial

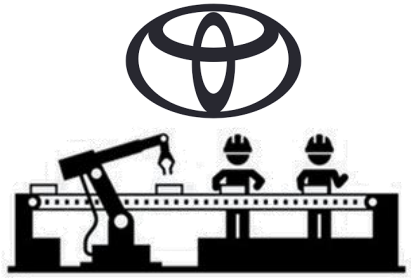
- A “Deliverable”
- Focus on customer

Maximise flow of **value** (to customer)

Lean

Minimum Viable Product (MVP)





7 Lean Manufacturing Principles

Toyota (2001)

1. Eliminate waste *(overproduction)*
2. Amplify learning *(feedback)*
3. Defer Commitment
4. Deliver Fast *(early feedback)*
5. Empower the team
6. Build integrity in *(spot faults)*
7. See the whole *(big picture)*

Toyota's "lean manufacturing principles" (2001)

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of "traditional software development"

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Waterfall

waterfall

/ˈwɔːtəfɔːl/

noun

1. a cascade of water falling from a height

   "Ruth kayaked steadily towards the towering waterfall, with a feeling of impending doom”

A bit more History

Commodore Personal Electronic Transactor - 1977

A bit more History

A bit more History

Commodore Personal Electronic Transactor - 1977

History of Waterfall

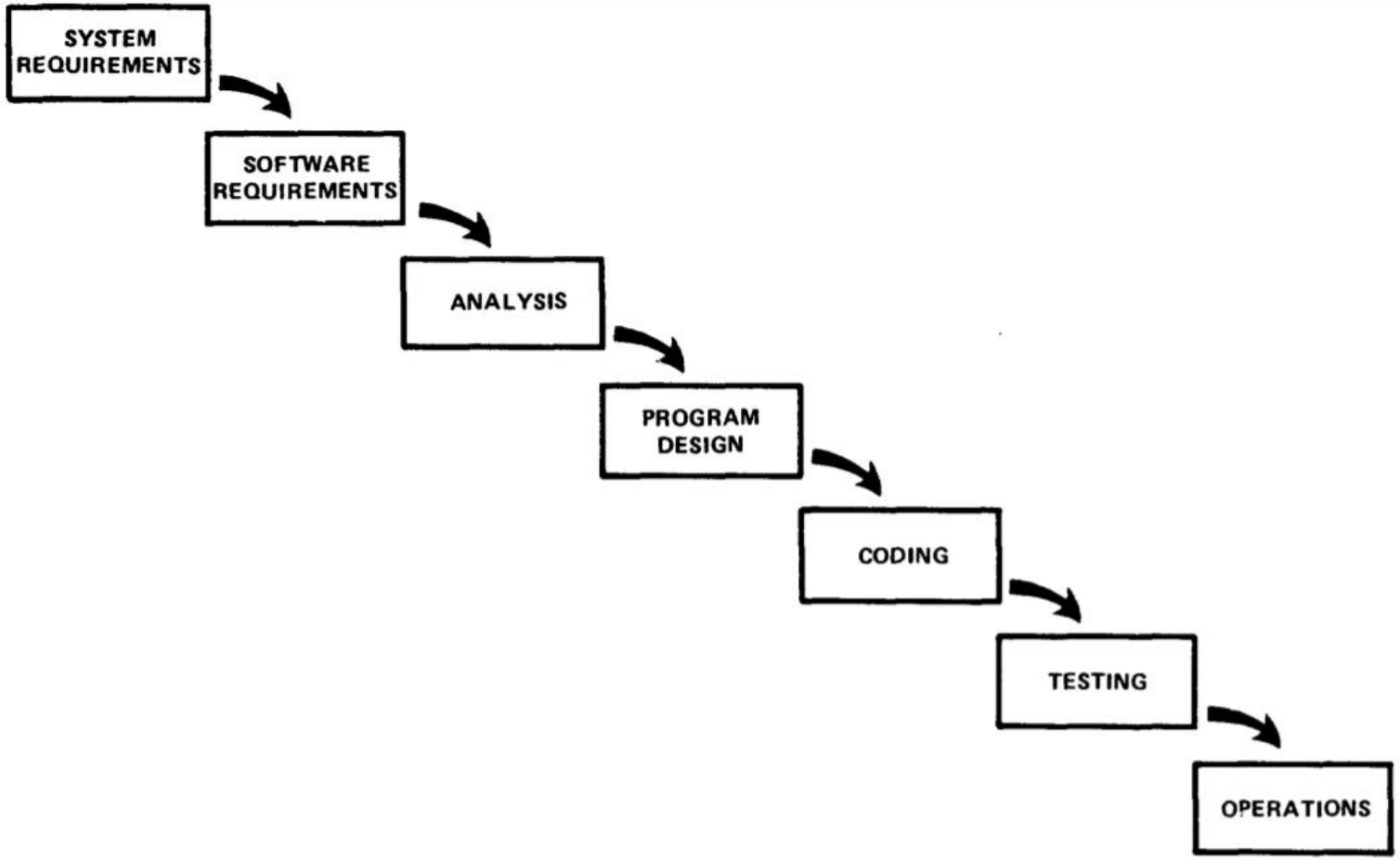


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall

Pros

- Repeatable – low management overhead
- Clear about expectations – explicitly collected
- Clear responsibilities and interfaces

Cons

- High risk – Late detection of problems leads to major redesign (and 100% overrun!)
- One long critical path
- Can we plan/analyse that well?
- Lots of documentation required
- No support for **change**

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall

Ironically, **Winston Royce** first described the Waterfall Method saying:

- "the implementation described above is **risky** and **invites failure"**
- “fail to satisfy the various external constraints, then invariably a **major redesign** is required ... so **disruptive** that the software requirements … which provides the rationale for everything are violated.“
- “In effect the development process has returned to the **origin** and one can expect up to a **100-percent overrun** in schedule and/or costs."

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall

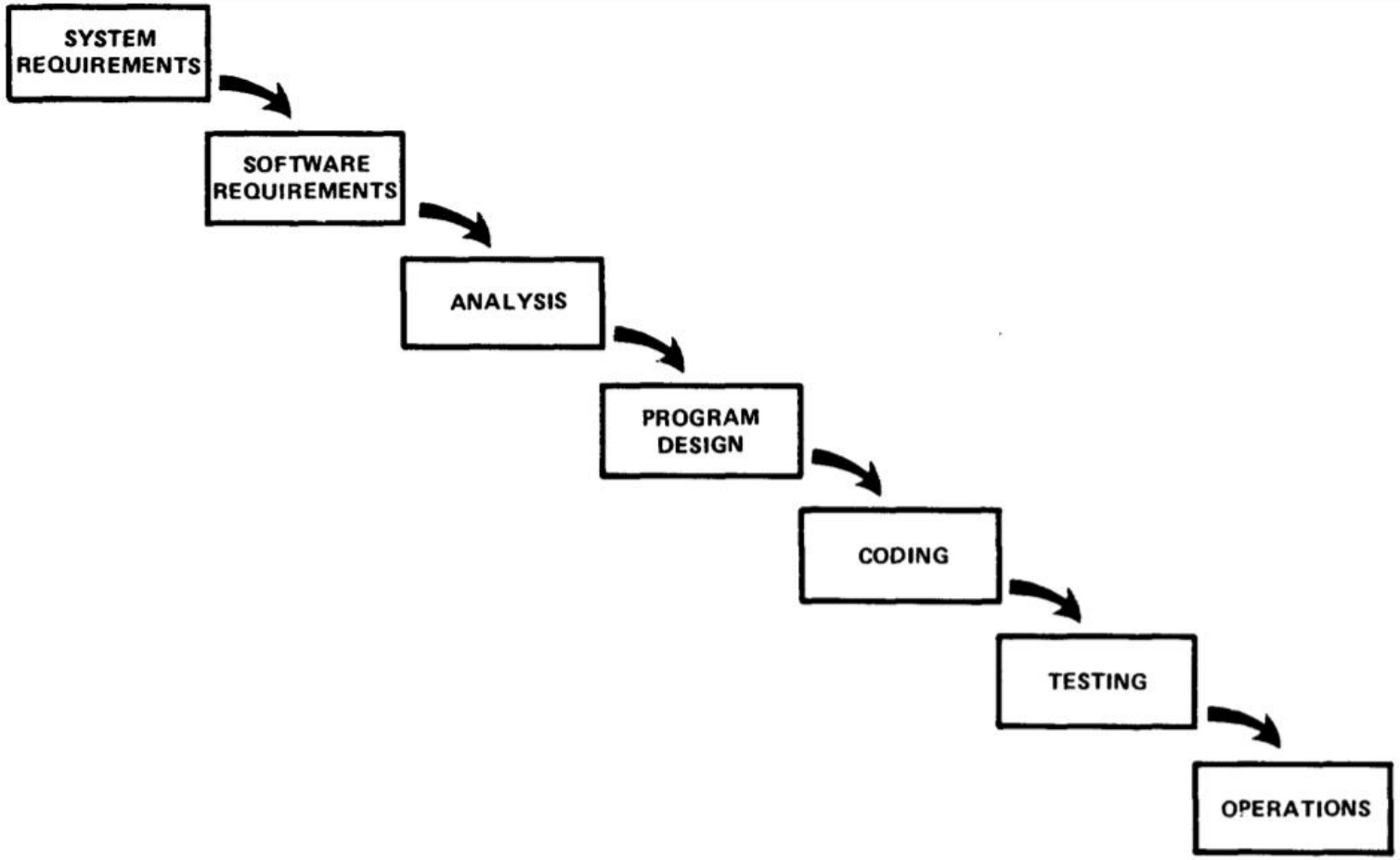


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall

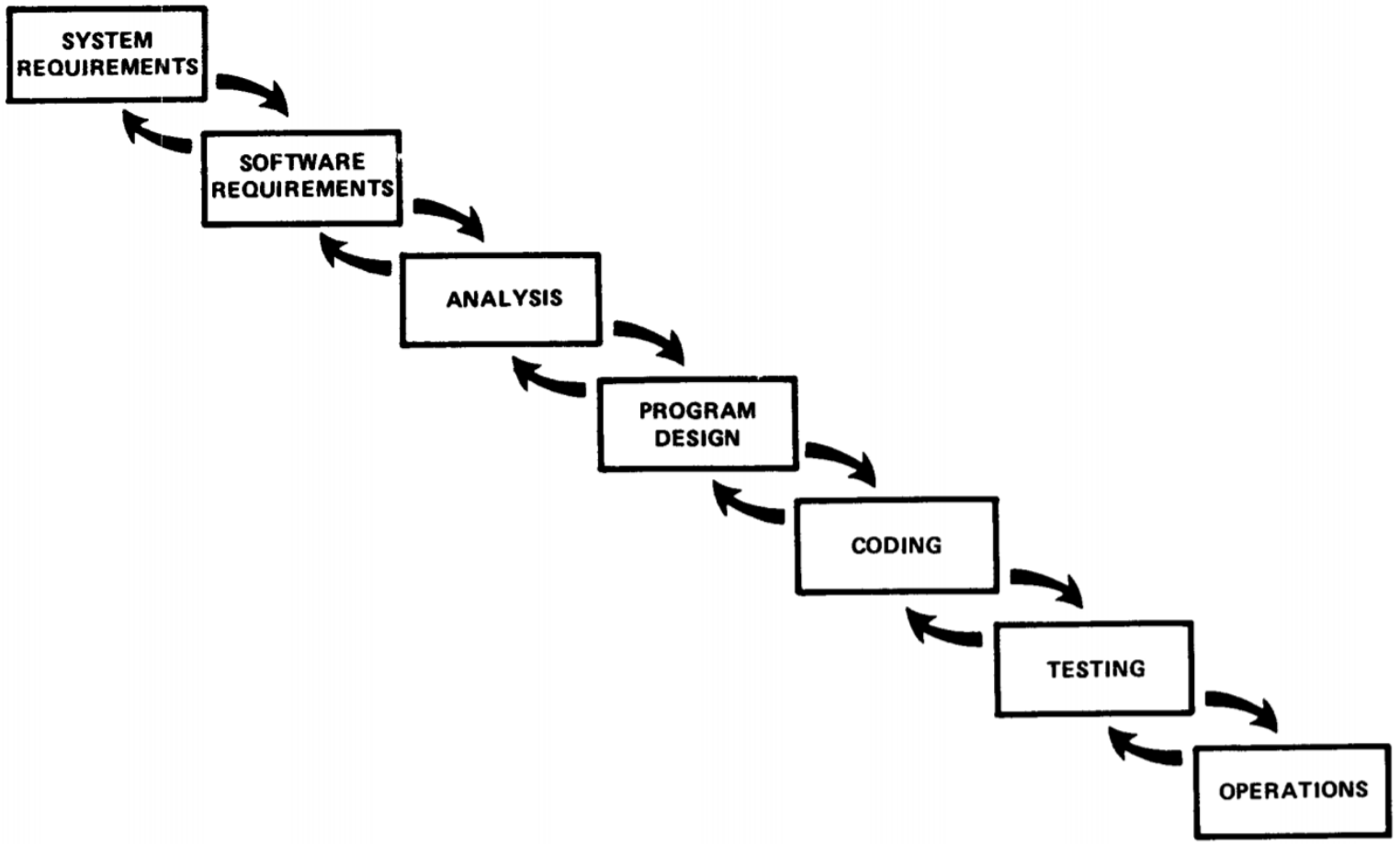


Figure 3. Hopefully, the iterative interaction between the various phases is confined to successive steps.

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall

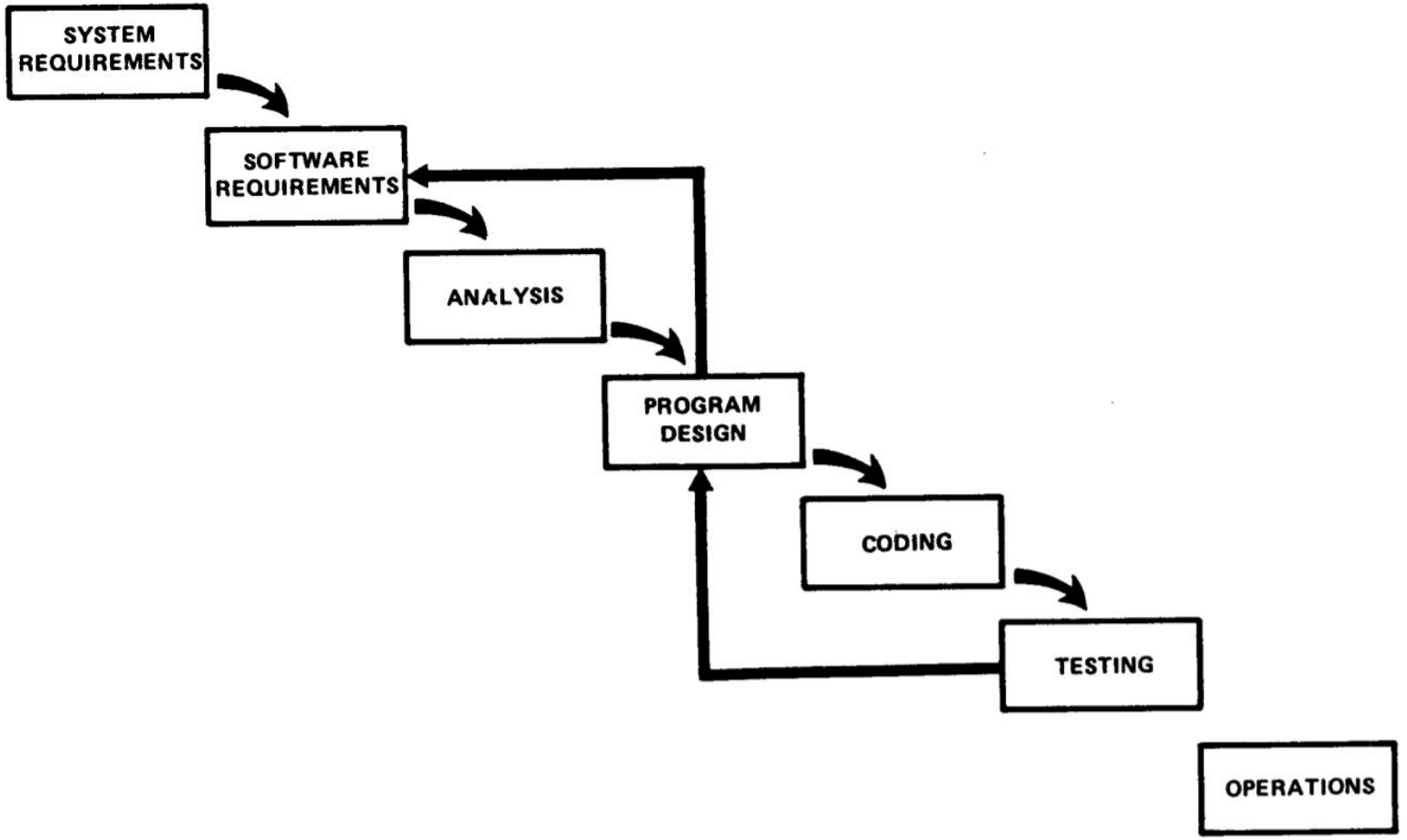
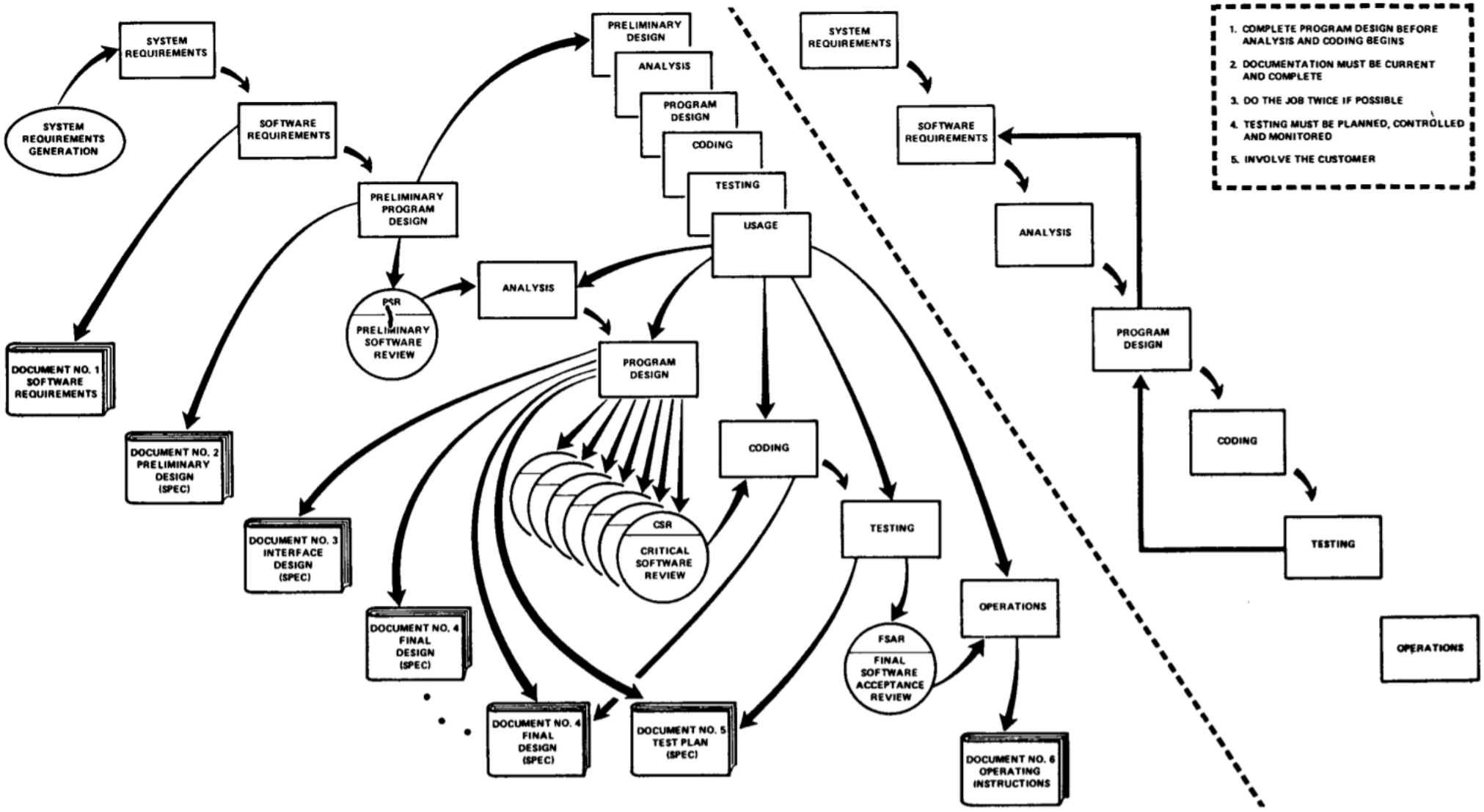


Figure 4. Unfortunately, for the process illustrated, the design iterations are never confined to the successive steps.

[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

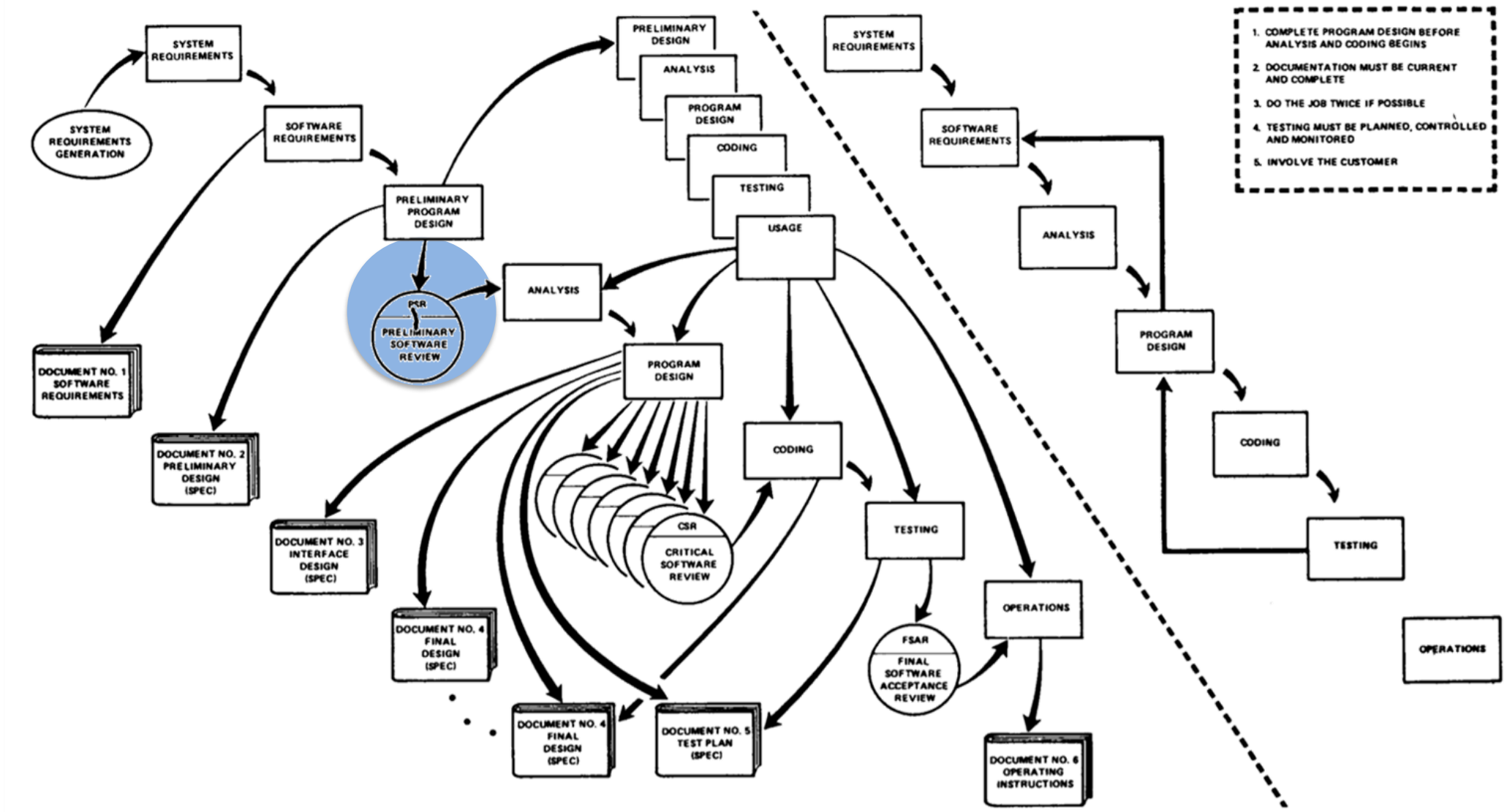
[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

History of Waterfall



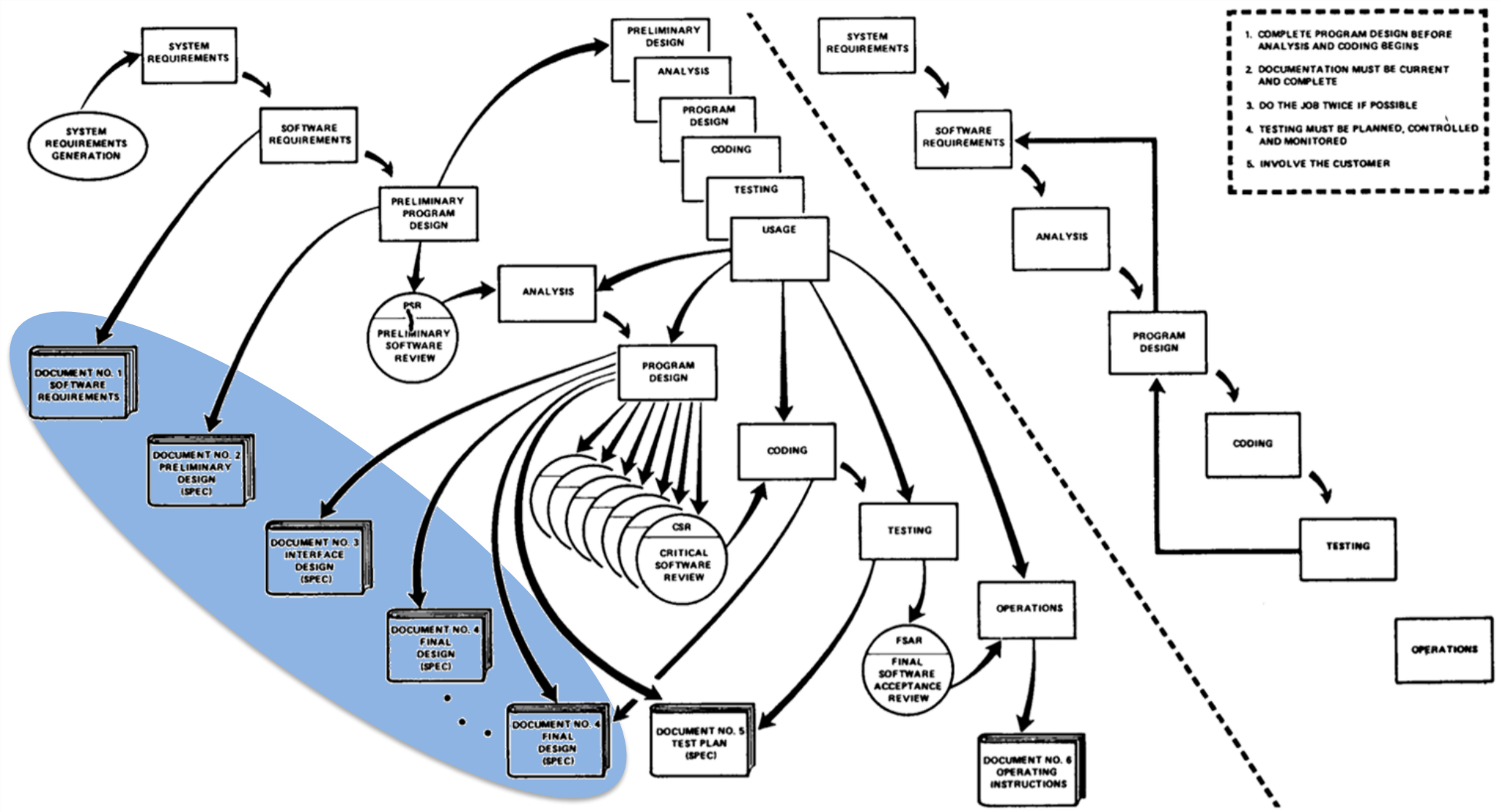
[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

1. Review Design



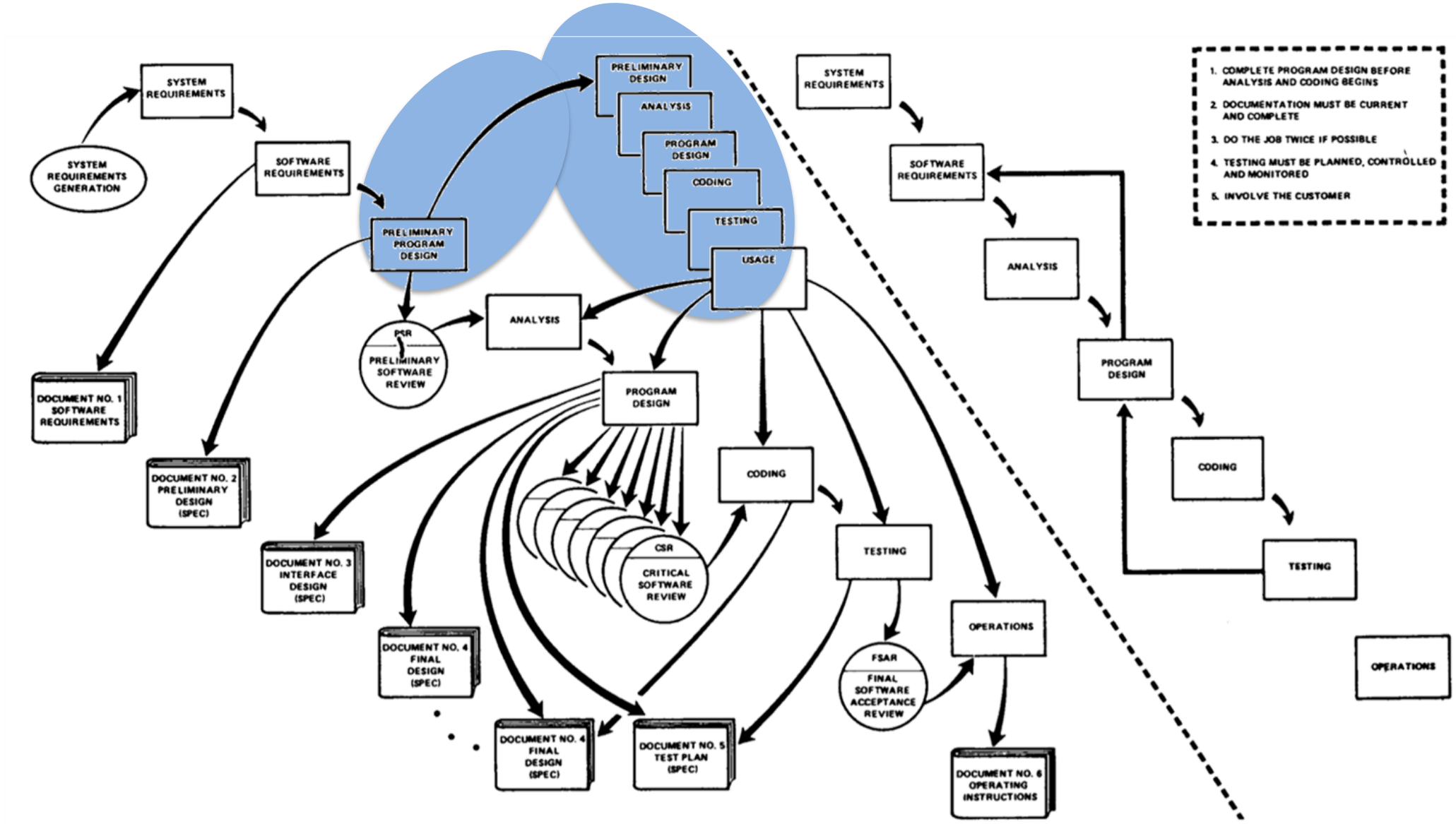
[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

2. Document



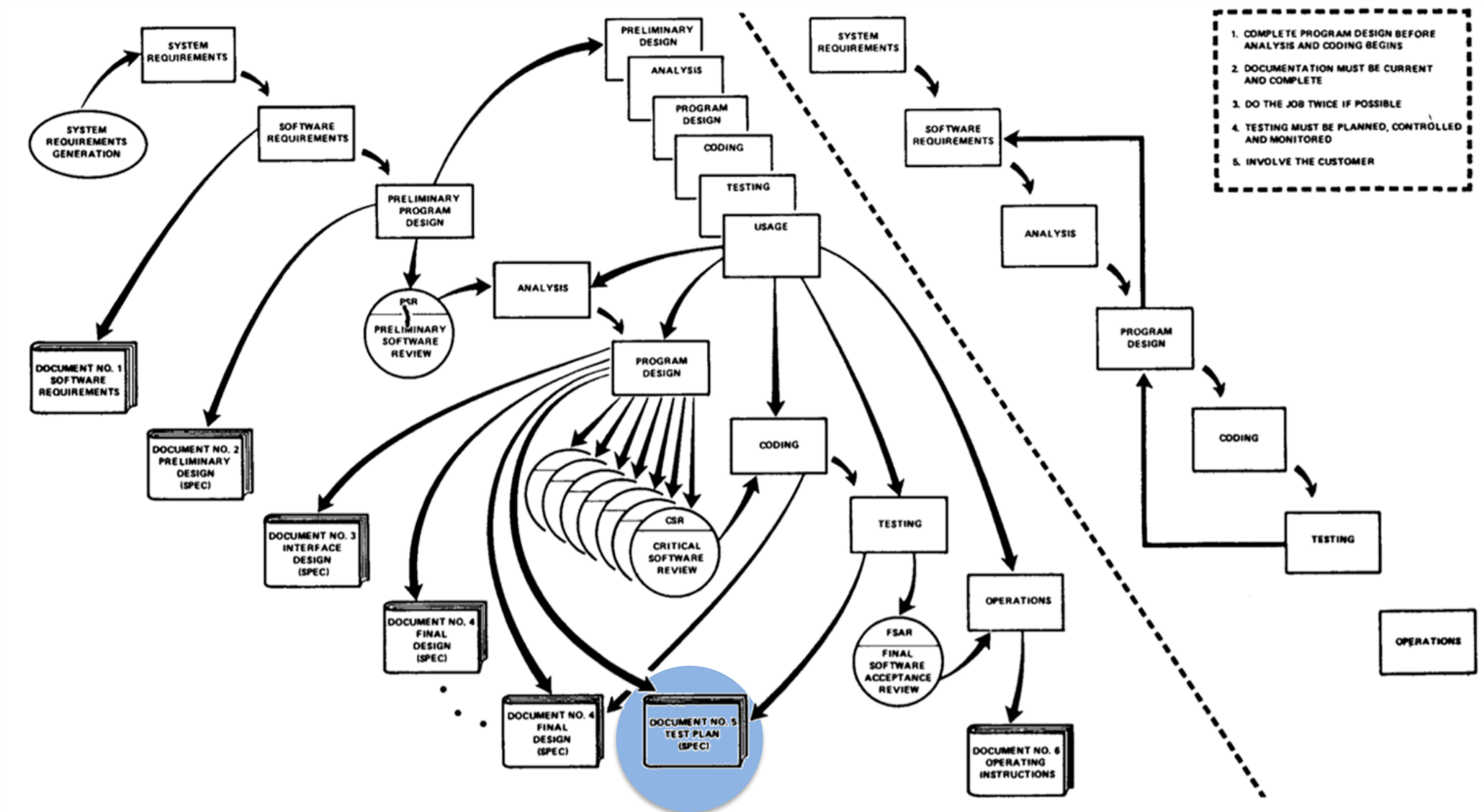
[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

3. Waterfall²



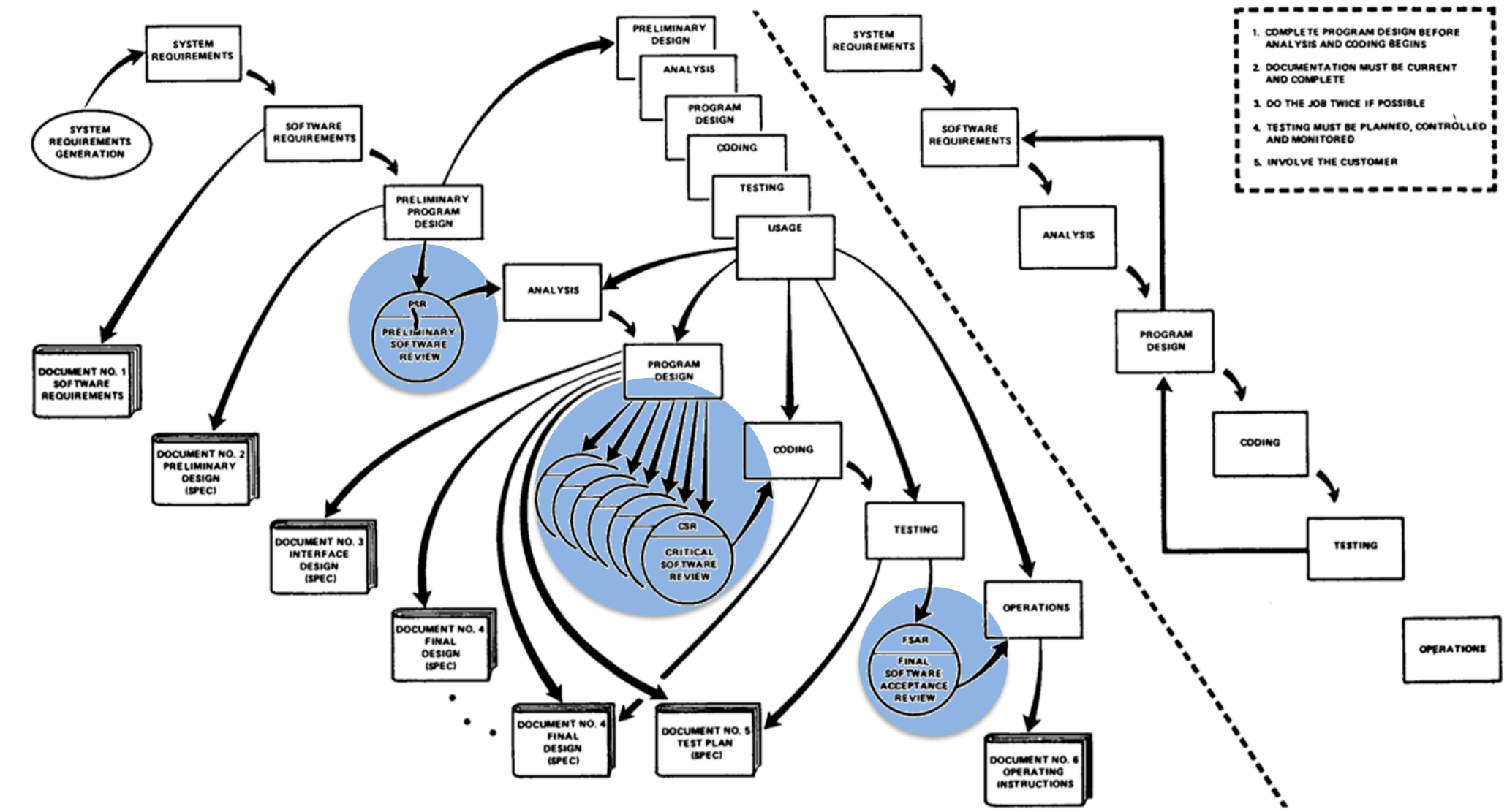
[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

4. Plan/Control Testing



[Royce, Managing the Development of Large Software Systems, 1970]
http://www-scf.usc.edu/~csci201/lectures/Lecture11/royce1970.pdf

5. Involve Customer



The Truth

- Nobody ever thought Naive Waterfall was a good model, (not even its creator)!
- Often called “traditional”, but was invented before computing really had any traditions!

- For all its faults, it’s simple and logical, easy to communicate
- Still widely used (*e.g.* 3rd year projects).

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

Agile

agile

/ˈadʒʌɪl/

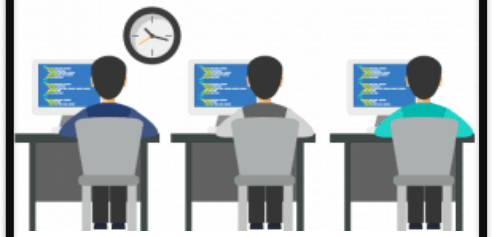
adjective

1. able to move quickly and easily.
   - *"Ruth was as agile as a kitten"*
   - nimble, lithe, spry, supple,
   - limber, sprightly, acrobatic …

“Traditional” Approach

Customer
(Requests it)

Product Owner
(Defines it)



Development Team
(Makes it)

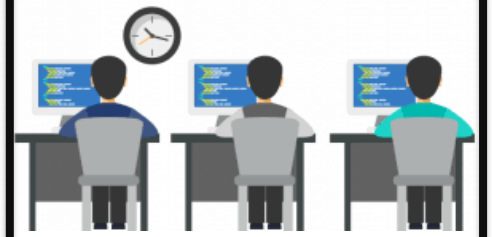
User

(Uses it)

“Agile” Approach

Customer
(Requests it)

Product Owner
(Defines it)



Development Team
(Makes it)

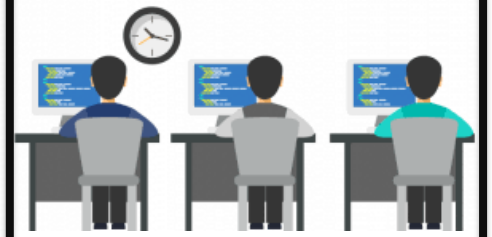
User

(Uses it)

“Agile” Approach

Customer
(Requests it)

Product Owner
(Defines it)



Development Team
(Makes it)

User

(Uses it)

Paradigm Shift

“Traditional”	**Agile**
Plan driven	?
Fix scope → estimate cost + time	
Solve it all in one go	
Progress is binary	
Cost, time and quality are at risk	
Risky until the end	

Paradigm Shift

“Traditional”

Plan driven

Fix scope → estimate cost + time

Solve it all in one go

Progress is binary

Cost, time and quality are at risk

Risky until the end

Agile

Value driven (prioritise requirements)

Paradigm Shift

“Traditional”	**Agile**
Plan driven	Value driven (prioritise requirements)
Fix scope → estimate cost + time	Fix cost + time → estimate scope
Solve it all in one go	
Progress is binary	
Cost, time and quality are at risk	
Risky until the end	

Paradigm Shift

“Traditional”		**Agile**
Plan driven	→	Value driven (prioritise requirements)
Fix scope → estimate cost + time	→	Fix cost + time → estimate scope
Solve it all in one go	→	Solve iteratively
Progress is binary		
Cost, time and quality are at risk		
Risky until the end		

Paradigm Shift

“Traditional”	**Agile**
Plan driven	Value driven (prioritise requirements)
Fix scope → estimate cost + time	Fix cost + time → estimate scope
Solve it all in one go	Solve iteratively
Progress is binary	Progress is gradual
Cost, time and quality are at risk	
Risky until the end	

Paradigm Shift

“Traditional”		**Agile**
Plan driven	→	Value driven (prioritise requirements)
Fix scope → estimate cost + time	→	Fix cost + time → estimate scope
Solve it all in one go	→	Solve iteratively
Progress is binary	→	Progress is gradual
Cost, time and quality are at risk	→	Scope at risk
Risky until the end		

Paradigm Shift

“Traditional”		**Agile**
Plan driven	→	Value driven (prioritise requirements)
Fix scope → estimate cost + time	→	Fix cost + time → estimate scope
Solve it all in one go	→	Solve iteratively
Progress is binary	→	Progress is gradual
Cost, time and quality are at risk	→	Scope at risk
Risky until the end	→	Risk declines with progress

Paradigm Shift

"Iron Triangle paradigm shift"

"Turned the iron triangle on it's head!"

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

The Agile Manifesto (2001)

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.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

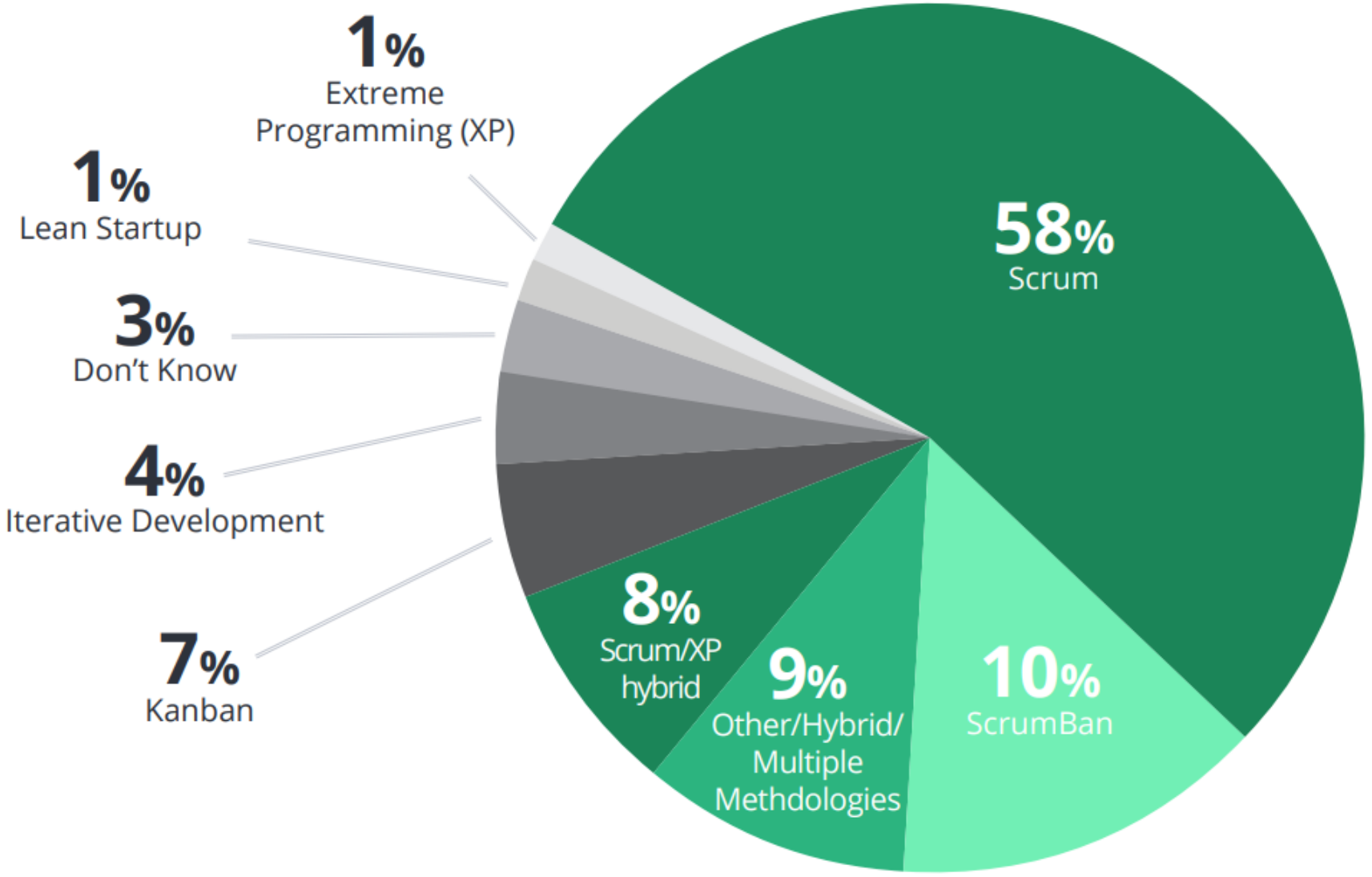
http://agilemanifesto.org/

The Agile Principles

1. **Early** and **continuous delivery** of **valuable** software
2. Welcome **changing requirements**, even late in development
3. Deliver **working software** frequently – e.g. every 2 weeks
4. Business people and developers must **work together daily**
5. Build around **motivated** individuals with **support and trust**
6. **Face-to-face** conversations
7. **Working software** is the primary measure of progress
8. **Sustainable** – able to maintain a constant pace indefinitely
9. **Excellence** – good design enhances agility
10. **Simplicity** – the art of maximizing the amount of work **not** done
11. **Self-organizing** teams
12. **Reflect** on how to become more effective

http://agilemanifesto.org/principles.html

Agile Approaches



Annual State of Agile Report #14 (2020)

Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

Manifesto and Principles

Scrum

The Daily Stand-Up

Part 2: Guest Lecture

User Stories

User Story:

```
As a <type of user>, I want <some goal> so that <some reason>.
```

ToDo List

ID	Story
7	As an unauthorized User I want to create a new account
1	As an unauthorized User I want to login
10	As an authorized User I want to logout
9	Create script to purge database
2	As an authorized User I want to see the list of items so that I can select one
4	As an authorized User I want to add a new item so that it appears in the list
3	As an authorized User I want to delete the selected item
5	As an authorized User I want to edit the selected item
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due
8	As an administrator I want to see the list of accounts on login

A "*Product Backlog*" built by the *Product Owner*

Epic: a group of related user stories.

Theme: top-level objective of the product/project.

Captures **scope / requirements:**

- *who, what, why* (not *how*)

User Stories

User Story:

```
As a <type of user>, I want <some goal> so that <some reason>.
```

Add:

- ***Priority***
- ***Estimated size / effort***
- ***Definition of Done***

Epic: a group of related user stories.

Theme: top-level objective of the product/project.

Captures **scope / requirements:**

- *who, what, why* (not *how*)

ToDo List

ID	Story	Estimation	Priority
7	As an unauthorized User I want to create a new account	3	1
1	As an unauthorized User I want to login	1	2
10	As an authorized User I want to logout	1	3
9	Create script to purge database	1	4
2	As an authorized User I want to see the list of items so that I can select one	2	5
4	As an authorized User I want to add a new item so that it appears in the list	5	6
3	As an authorized User I want to delete the selected item	2	7
5	As an authorized User I want to edit the selected item	5	8
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9
8	As an administrator I want to see the list of accounts on login	2	10
Total		**30**	

A "*Product Backlog*" built by the *Product Owner*

Scrum Roles

- *Product Owner*
  - Represents **customer's** view
  - Communicates the **vision** (via User Stories) to the team
  - Prioritizes User Stories, and accepts/rejects them at the end of sprint.
  - Secures funding, Manages the ROI

- *Development Team*
  - Developers, testers, designers, analysts...
  - 3 to 9 members (when 9 not enough, split into two)

- *Scrum Master*
  - **Not** a manager, but a "servant leader" – removes impediments
  - Works across teams to improve communication
  - Hosts meetings (Sprint Planning, Daily Scrum, Retrospective)

https://www.youtube.com/watch?v=9TycLR0TqFA

Scrum Values

➔ Empower the team

Commitment (to goals in the sprint)

Courage (to do what you think is right)

Focus (on the work items in the current sprint)

Openness (about any challenges you face)

Respect (trust that everyone is doing their best)

Scrum

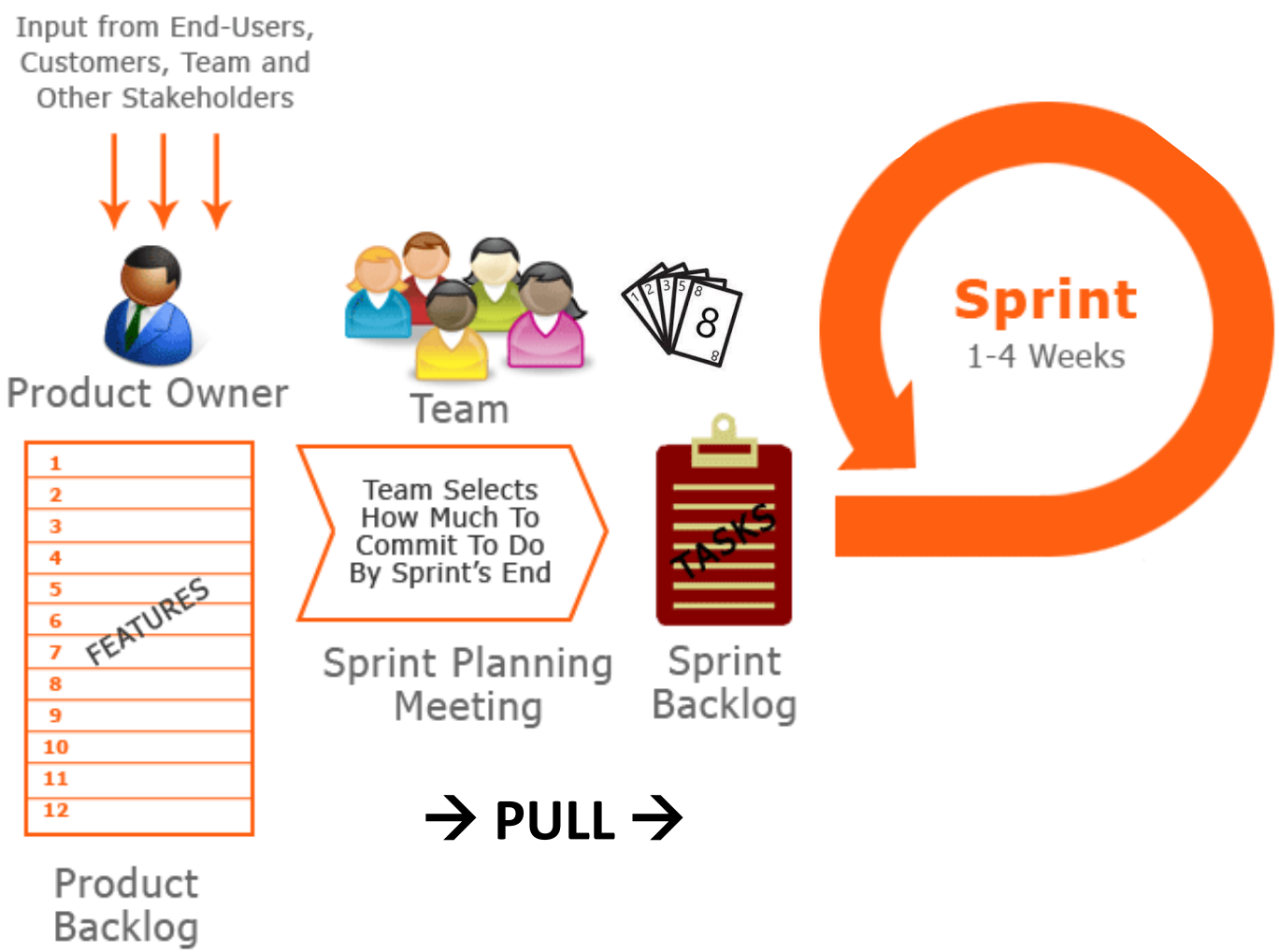

Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Product Backlog

Scrum

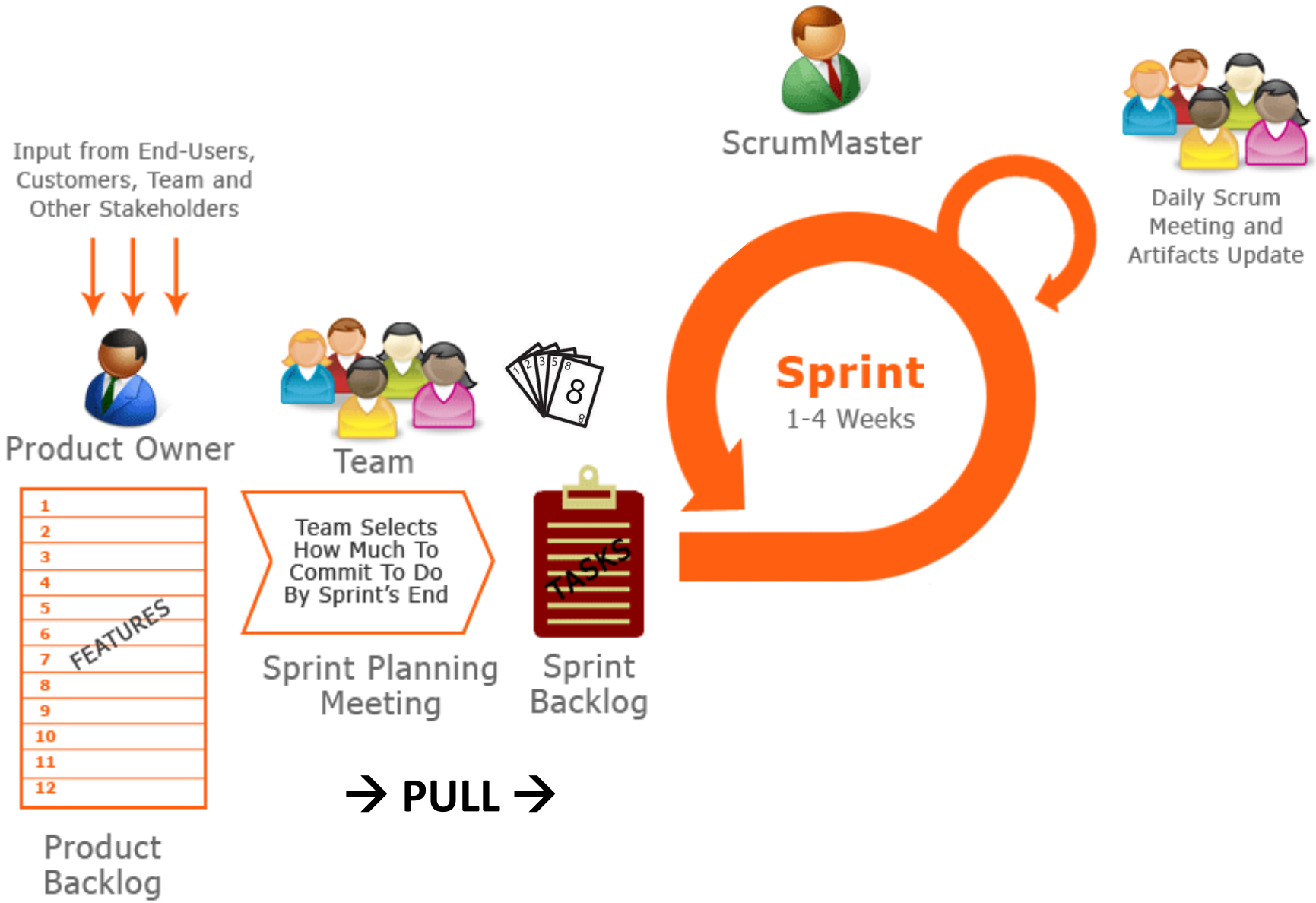

Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
1
2
3
4
5
6
7
FEATURES
8
9
10
11
12
Product Backlog


Sprint
1-4 Weeks

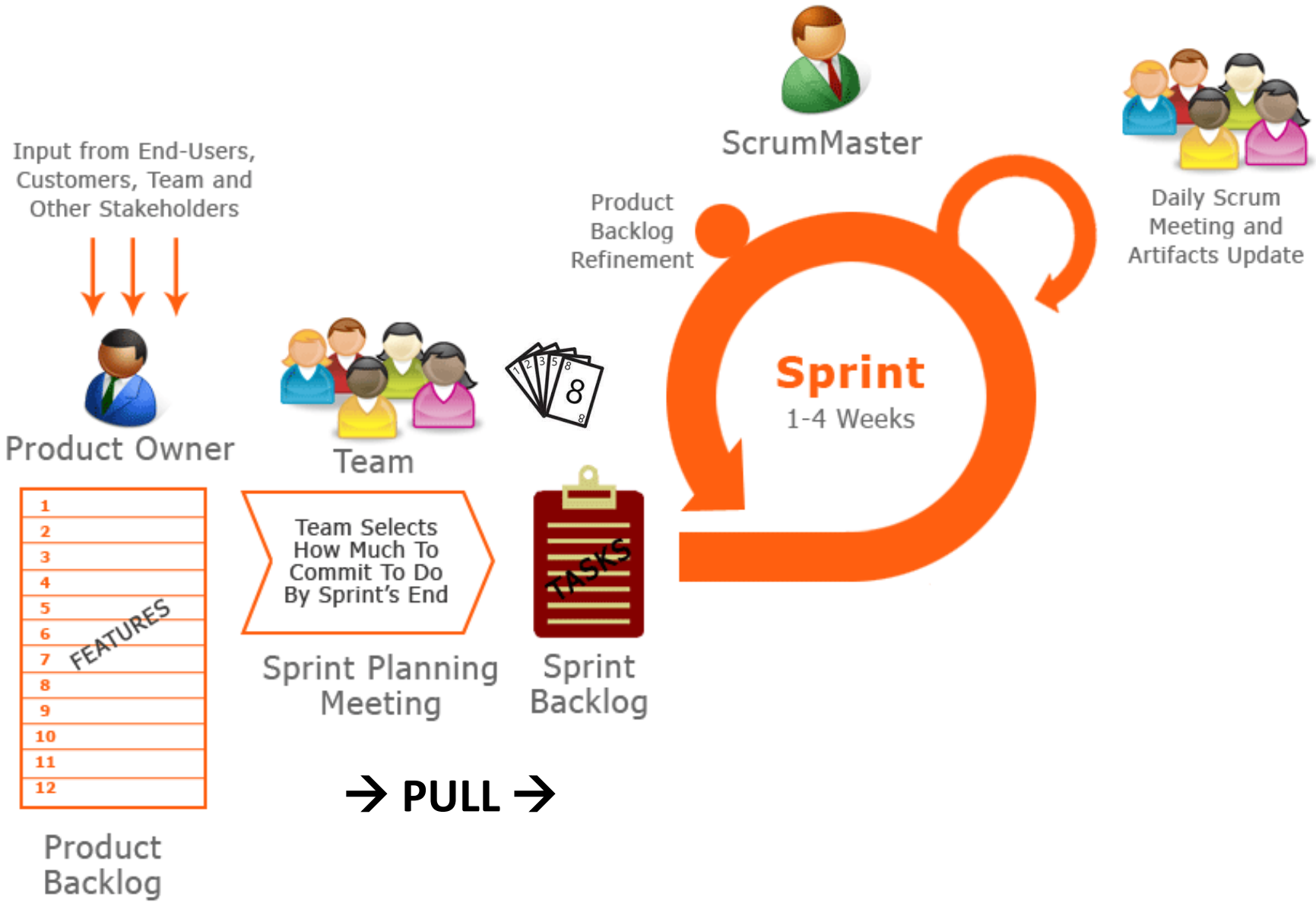
Scrum


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
Sprint
1-4 Weeks
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Sprint Planning Meeting
Sprint Backlog
→ PULL →
Product Backlog

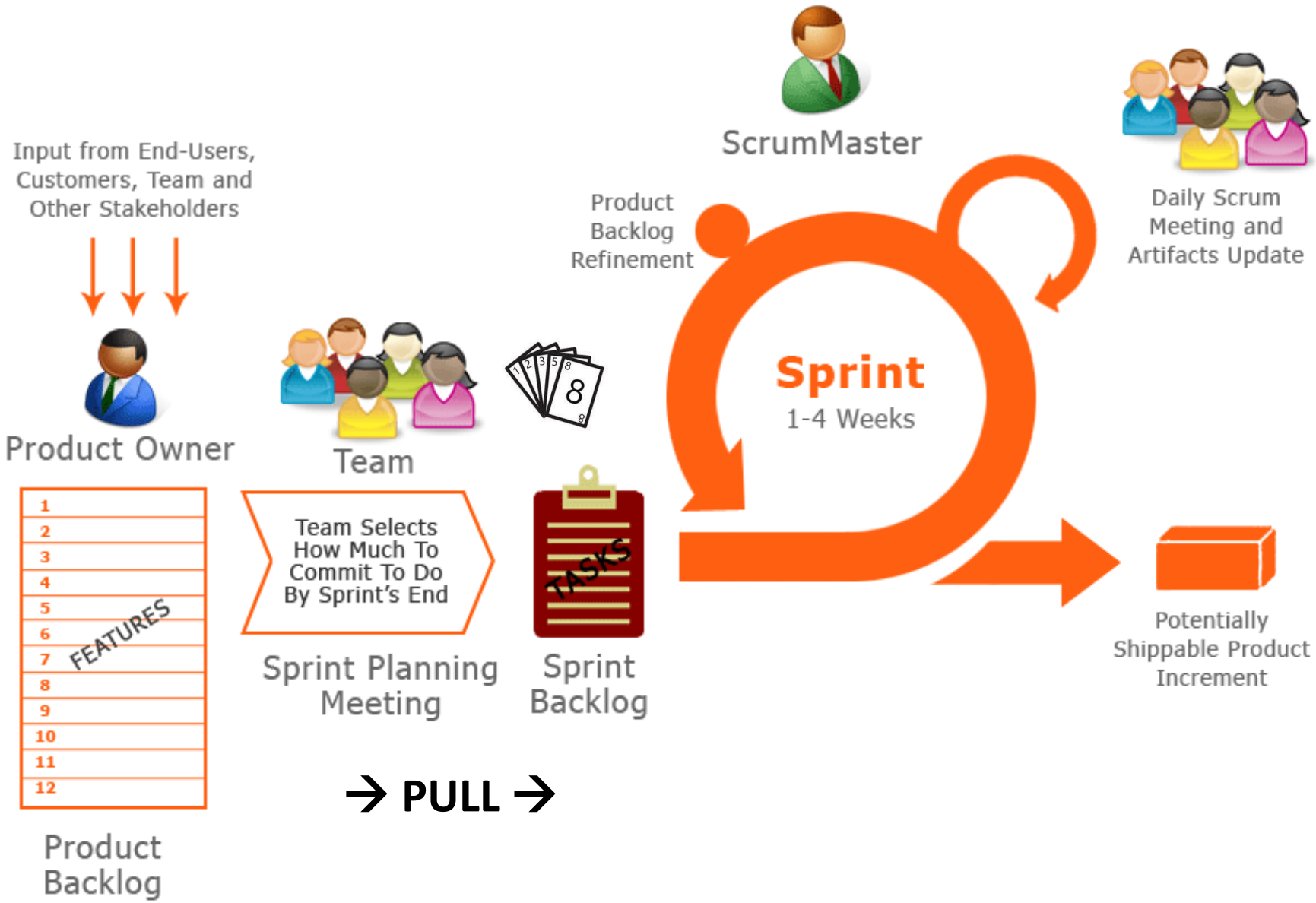
Scrum


ScrumMaster
Input from End-Users, Customers, Team and Other Stakeholders
Daily Scrum Meeting and Artifacts Update
Sprint
1-4 Weeks
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Product Backlog
Sprint Planning Meeting
Sprint Backlog
→ PULL →

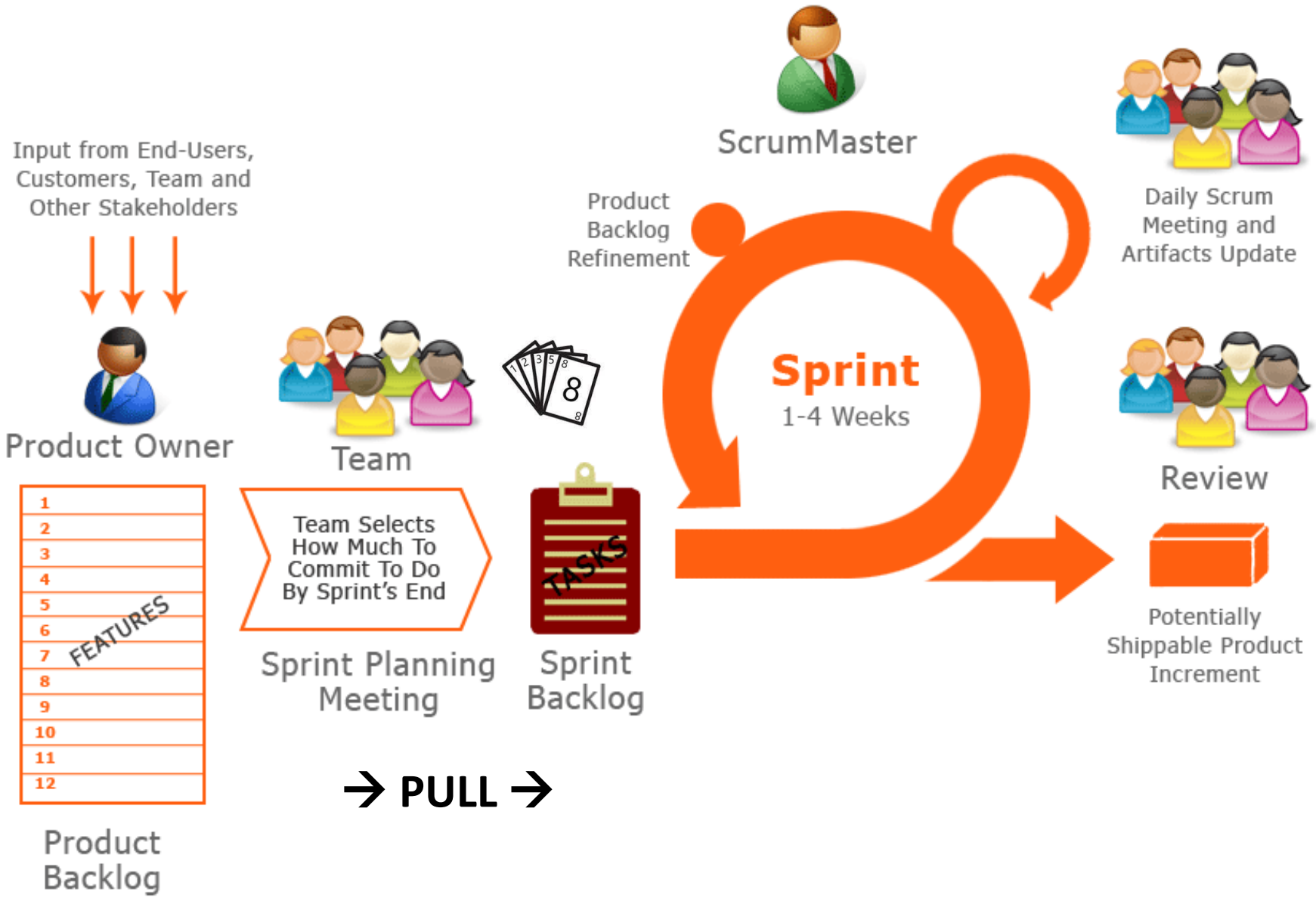
Scrum


Input from End-Users, Customers, Team and Other Stakeholders
ScrumMaster
Daily Scrum Meeting and Artifacts Update
Product Backlog Refinement
Sprint
1-4 Weeks
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Product Backlog
Sprint Planning Meeting
Sprint Backlog
→ PULL →

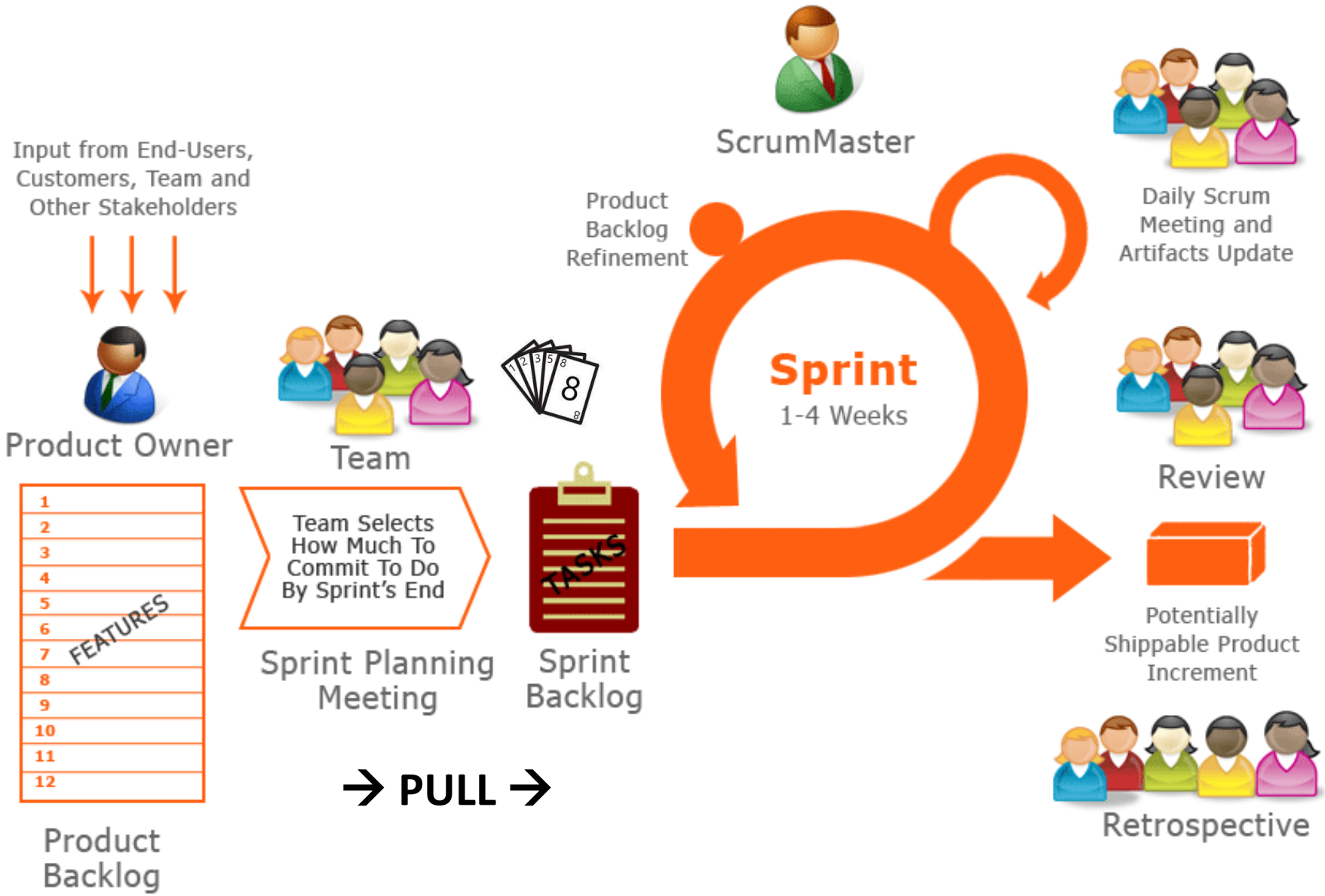
Scrum


Input from End-Users, Customers, Team and Other Stakeholders
ScrumMaster
Daily Scrum Meeting and Artifacts Update
Product Backlog Refinement
Product Owner
Team
Sprint
1-4 Weeks
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Potentially Shippable Product Increment
Sprint Planning Meeting
Sprint Backlog
→ PULL →
Product Backlog

Scrum


Input from End-Users, Customers, Team and Other Stakeholders
ScrumMaster
Product Backlog Refinement
Daily Scrum Meeting and Artifacts Update
Product Owner
Team
Sprint
1-4 Weeks
Review
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Potentially Shippable Product Increment
Sprint Planning Meeting
Sprint Backlog
→ PULL →
Product Backlog

Scrum

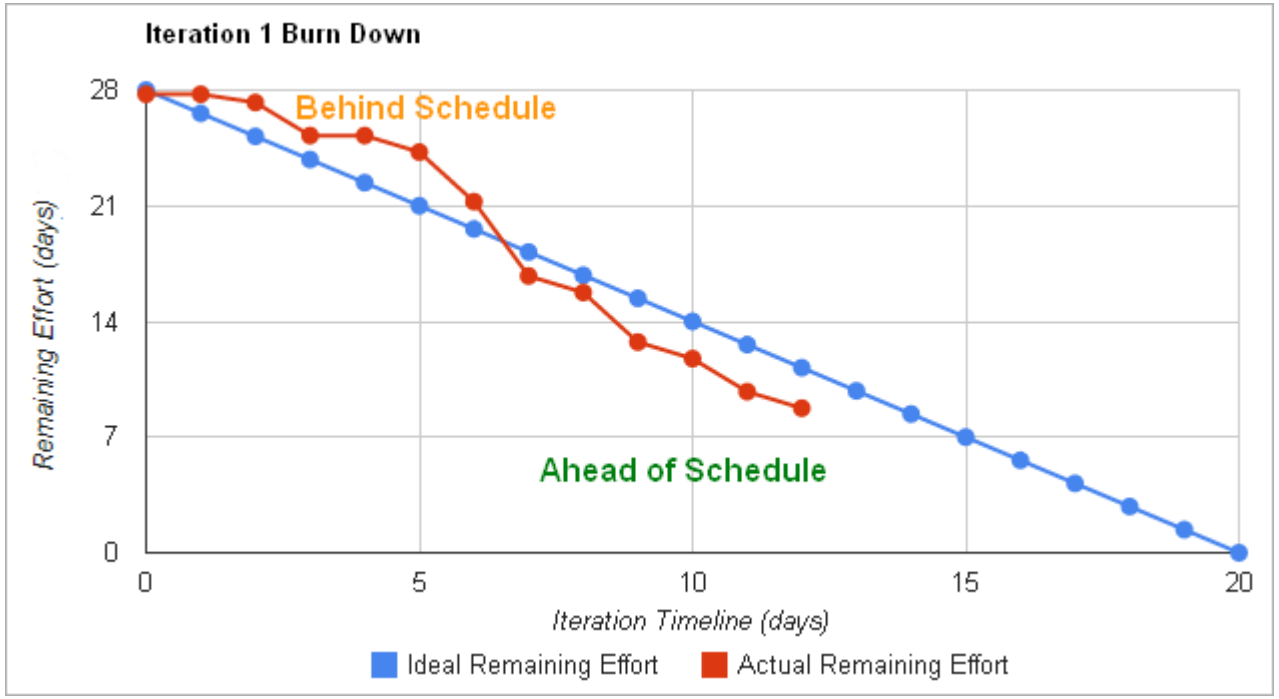

Input from End-Users, Customers, Team and Other Stakeholders
ScrumMaster
Product Backlog Refinement
Daily Scrum Meeting and Artifacts Update
Product Owner
Team
Sprint
1-4 Weeks
Review
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Team Selects How Much To Commit To Do By Sprint's End
TASKS
Potentially Shippable Product Increment
Sprint Planning Meeting
Sprint Backlog
→ PULL →
Retrospective
Product Backlog

Scrum *Artifacts*

- ***User Story***
  - A small piece of functionality to work on during a Sprint.
- ***Task***
  - May be unrelated to a User Story. For example, investigating a memory issue
- ***Product Backlog***
  - A list of User Stories and Tasks for future Sprints.
- ***Sprint backlog***
  - A list of User Stories and Tasks picked from Backlog for the current Sprint.
- ***Product increment***
  - A potentially shippable piece of functionality delivered at the end of the Sprint.
- ***Extensions***
  - Reports like Burndown Chart, Velocity, etc. used to keep track of the team's progress.

Burndown Chart

- Displays remaining effort for the sprint
- Allows estimation of the *velocity* of the team.
- Scrum Master must:
  - Get the team to fix estimation
  - Ensure additional stories ready to be added into the sprint



Today

Part 1: Lecture

Lean

History

Eliminating Waste

Kanban

Minimum Viable Product (MVP)

Waterfall

The myth of “traditional software development”

Agile

Paradigm Shift

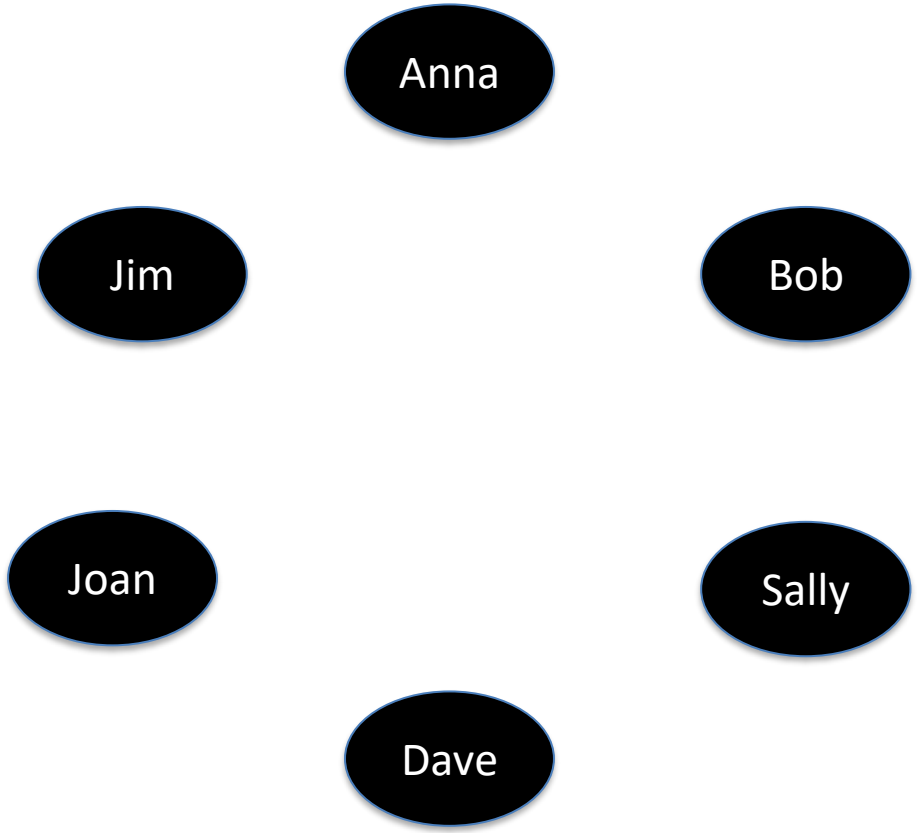
Manifesto and Principles

Scrum

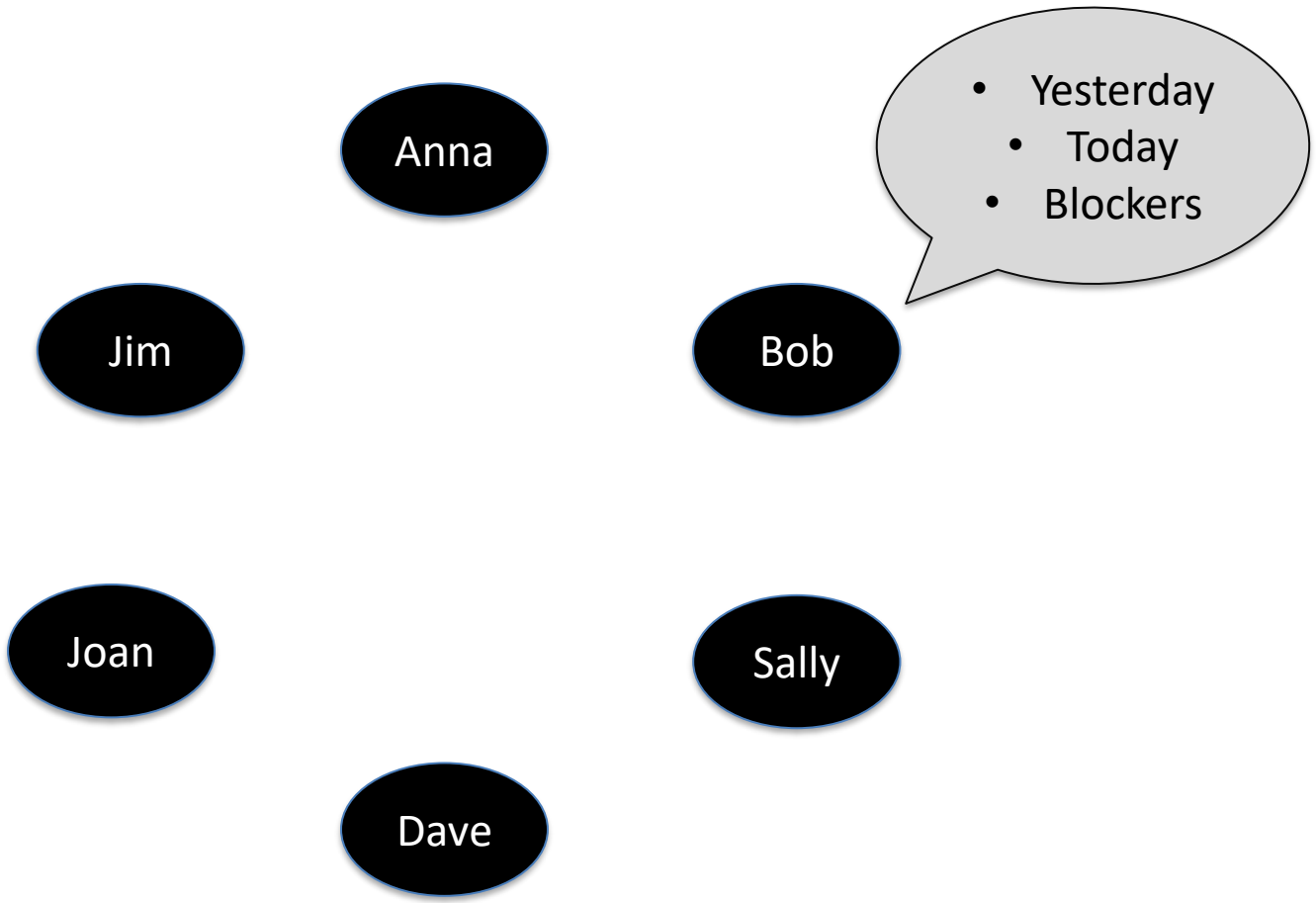
The Daily Stand-Up

Part 2: Guest Lecture

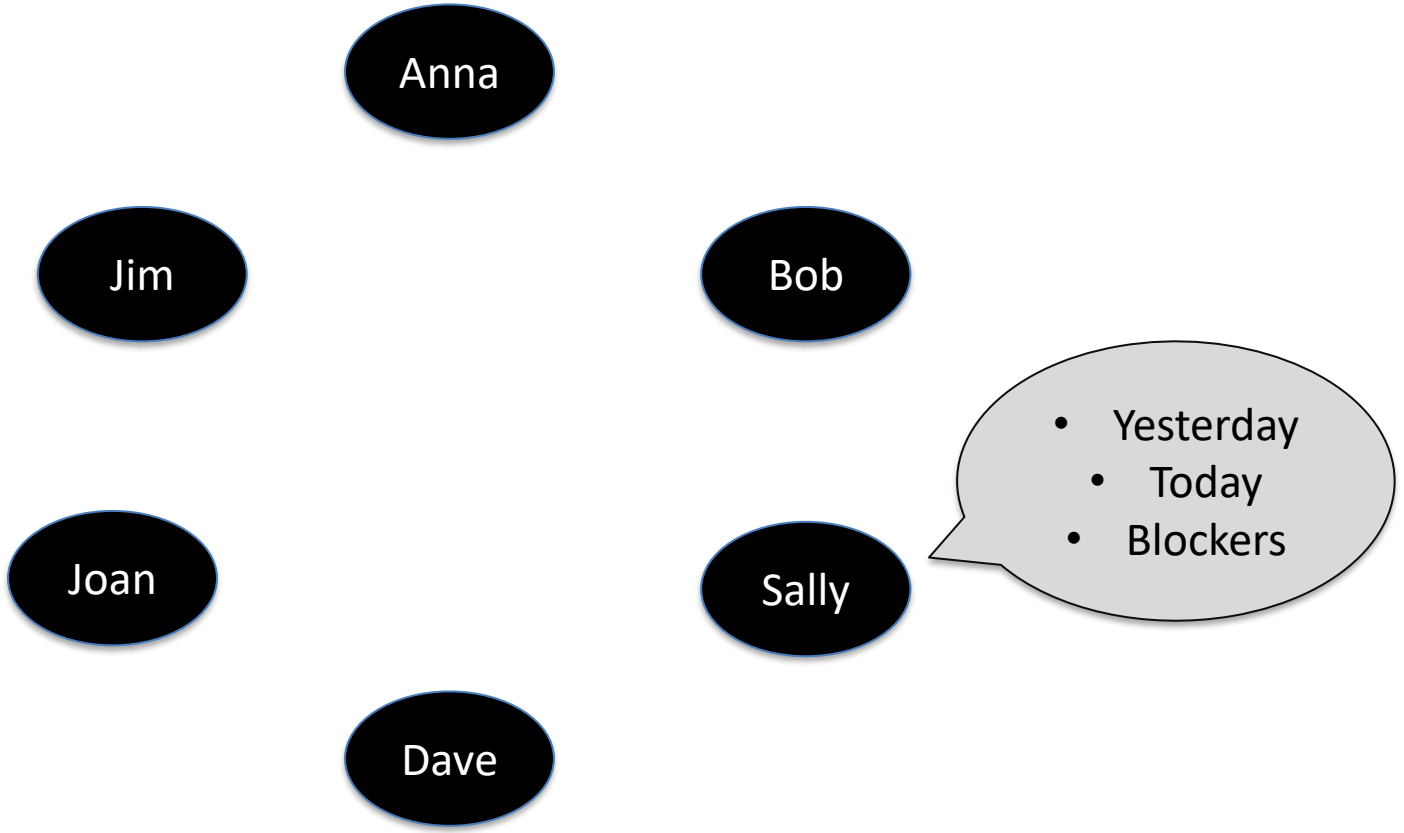
The Daily Stand Up



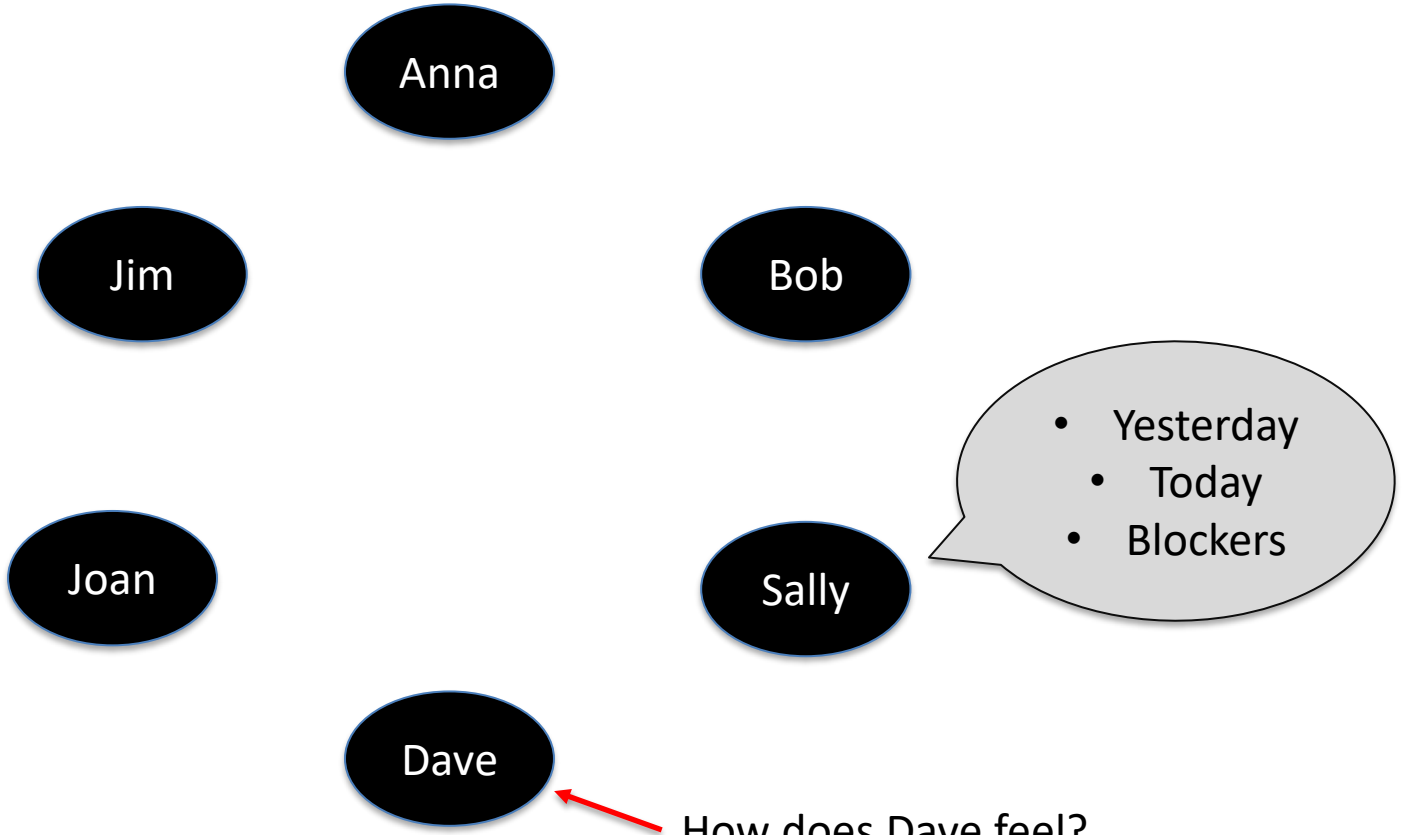
The Daily Stand Up



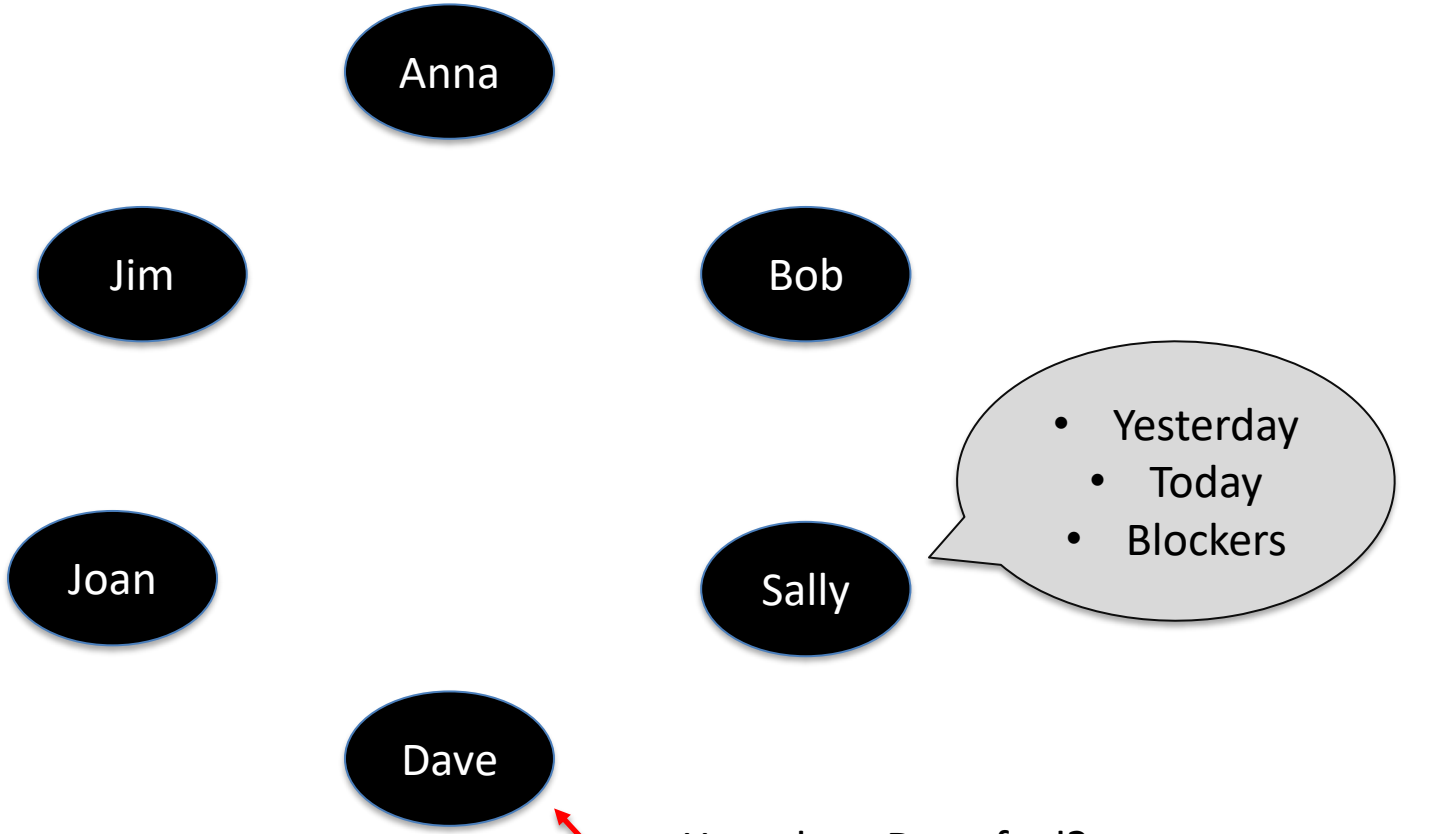
The Daily Stand Up



The Daily Stand Up



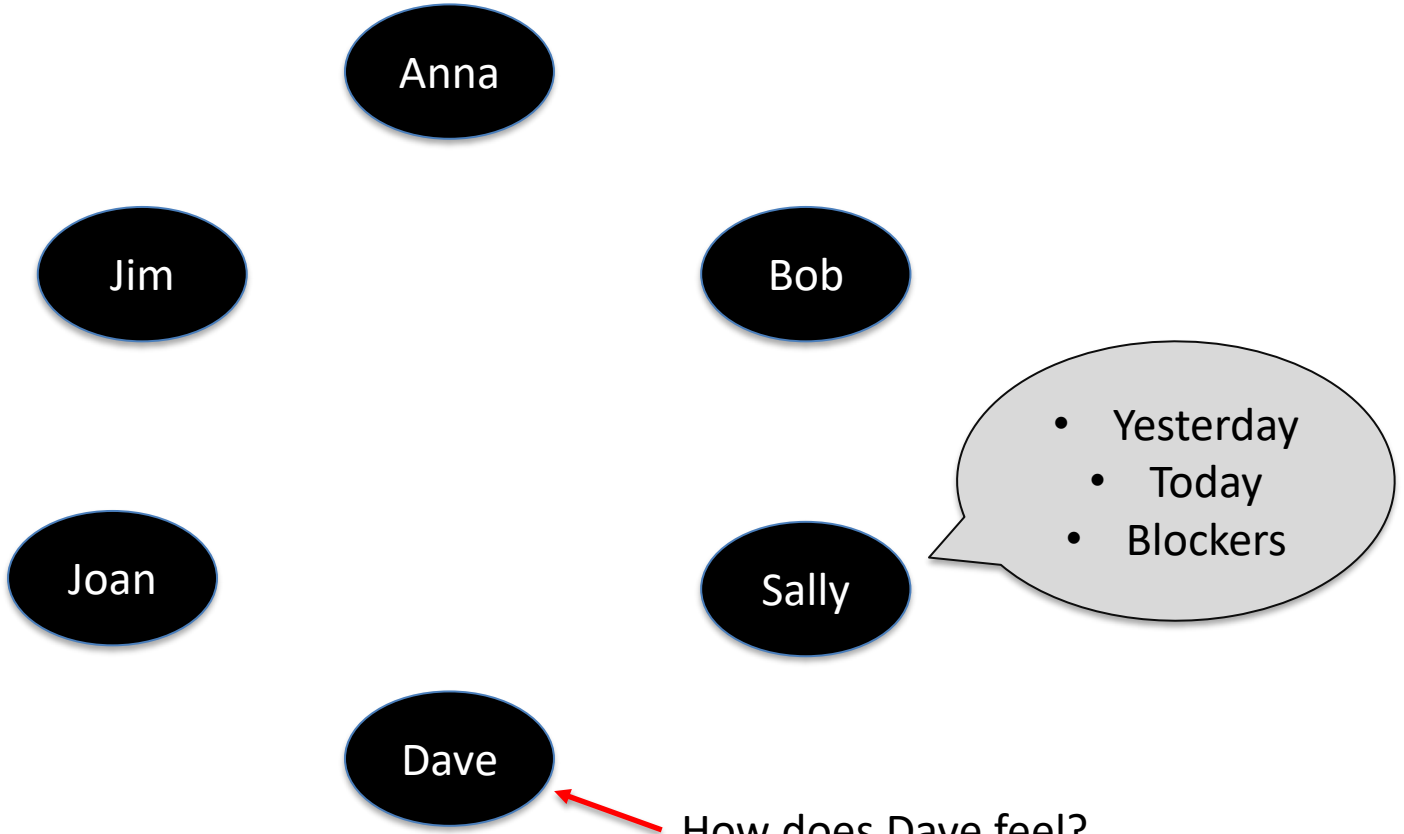
The Daily Stand Up



How does Dave feel?

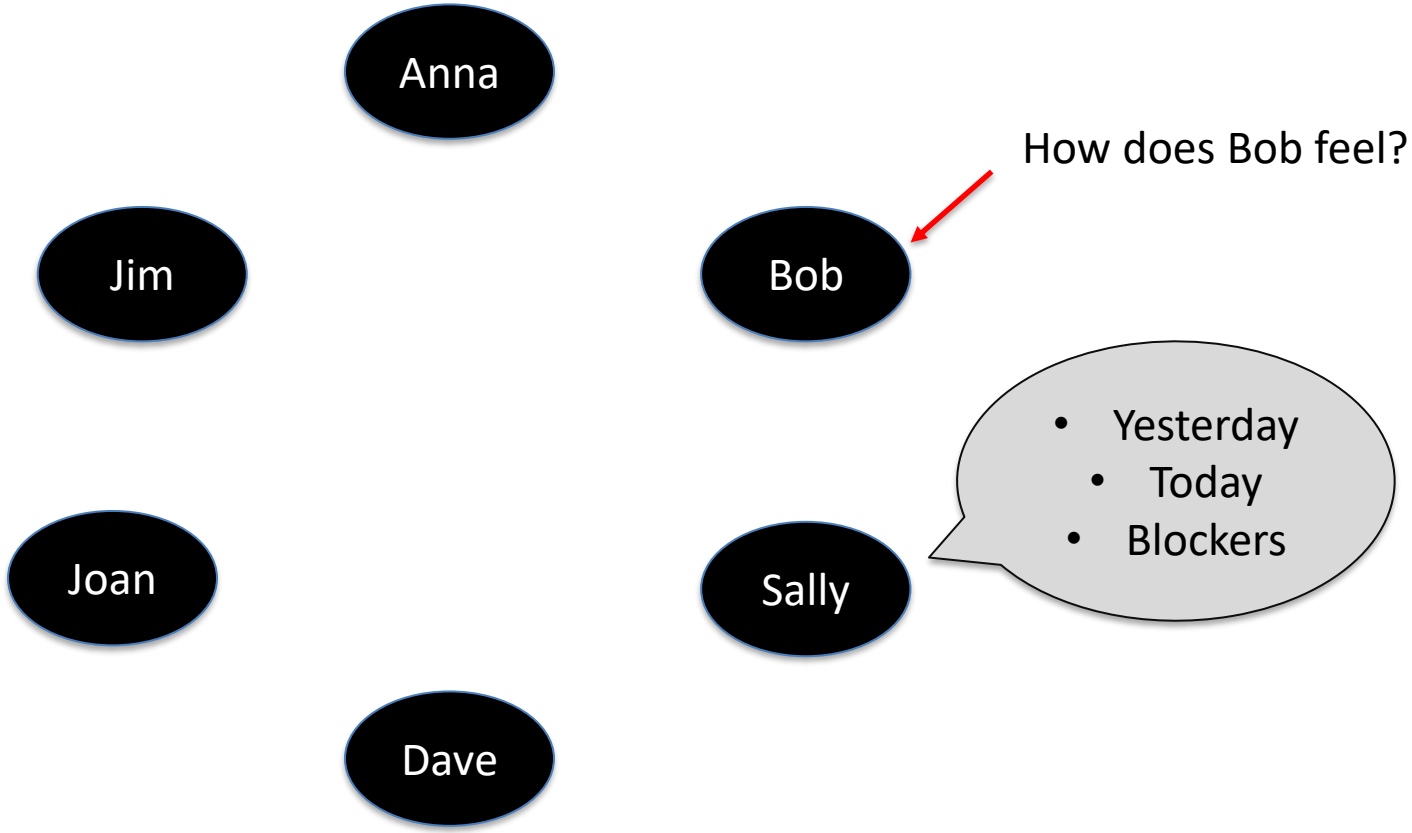
- Not listening - thinking about what to say
- Wants to impress colleagues
- Wants to avoid embarrassment

The Daily Stand Up

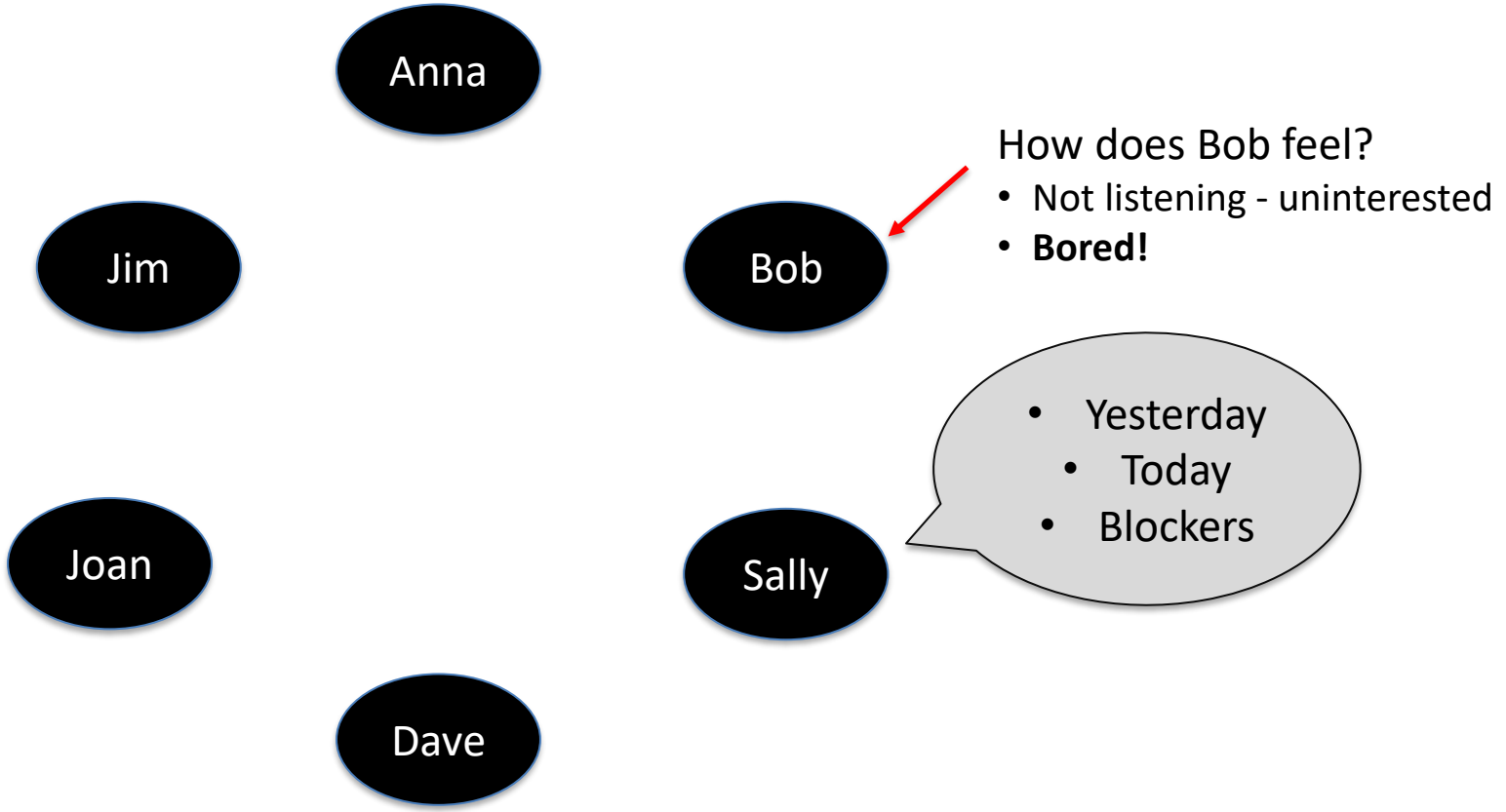


•STRESSED!

The Daily Stand Up



The Daily Stand Up

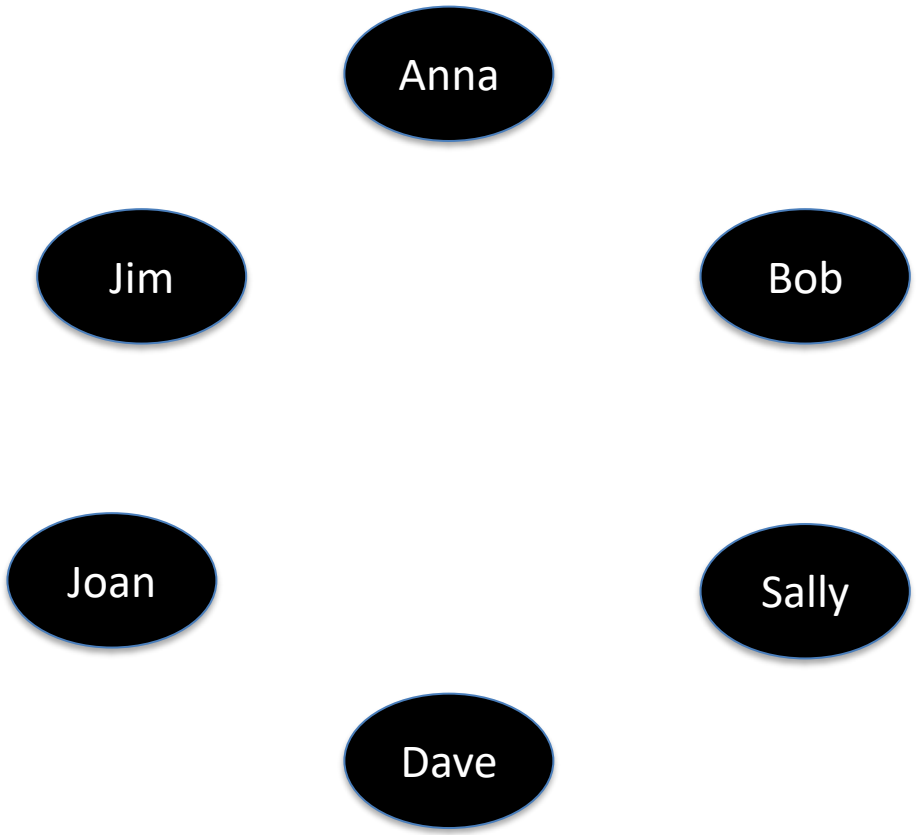


The Daily Stand Up

Task 1 - blocked

Task 2 - in progress

Task 3 - not started

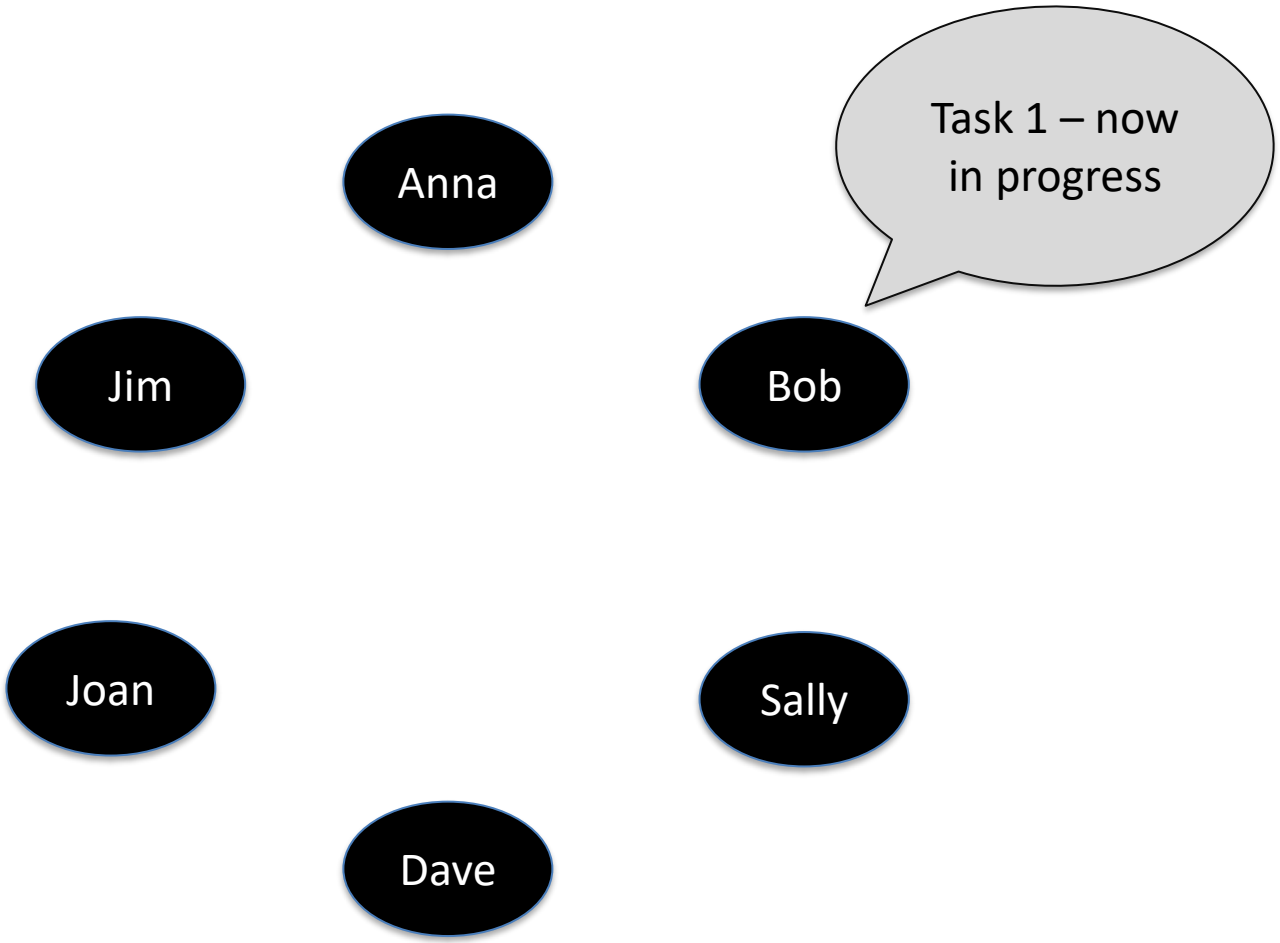


The Daily Stand Up

Task 1 - **in progress** - (Bob)

Task 2 - in progress

Task 3 - not started

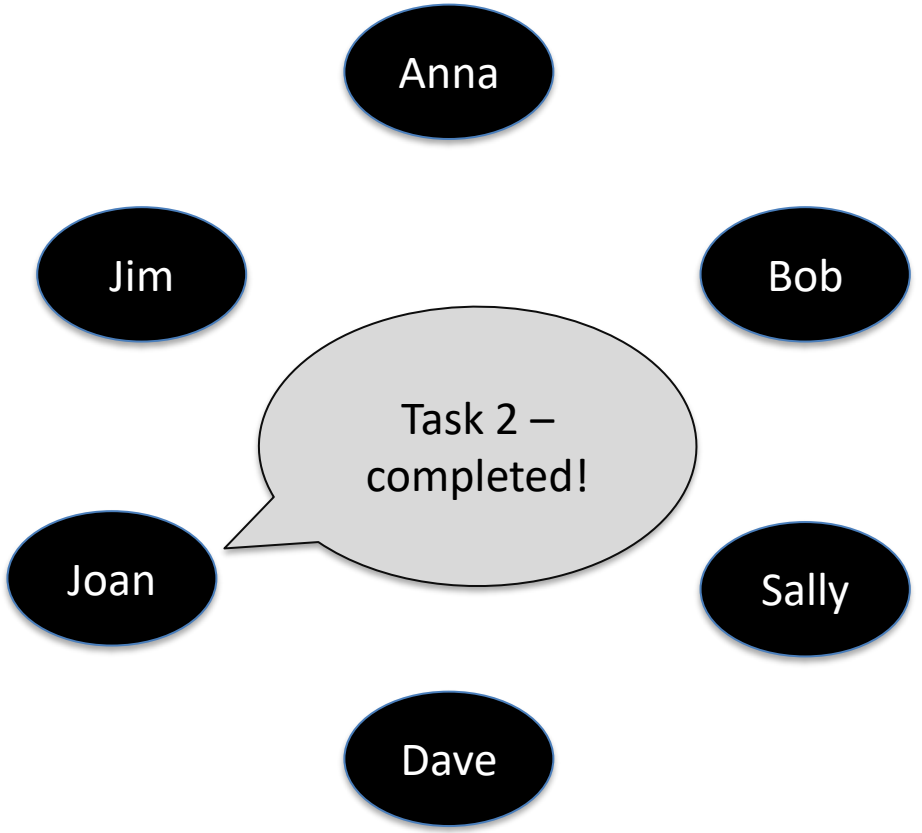


The Daily Stand Up

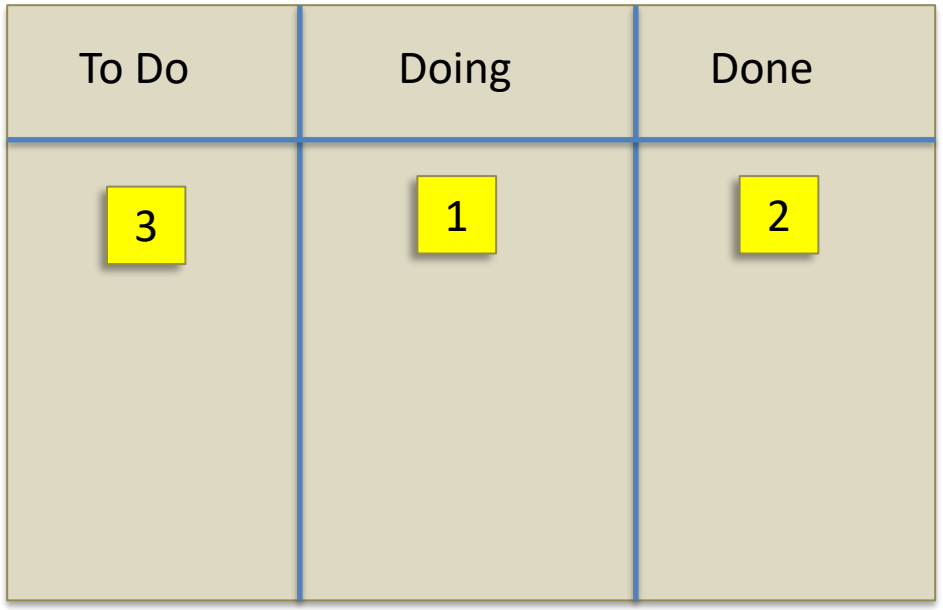
Task 1 - in progress - (Bob)

Task 2 - **completed** - (Joan)

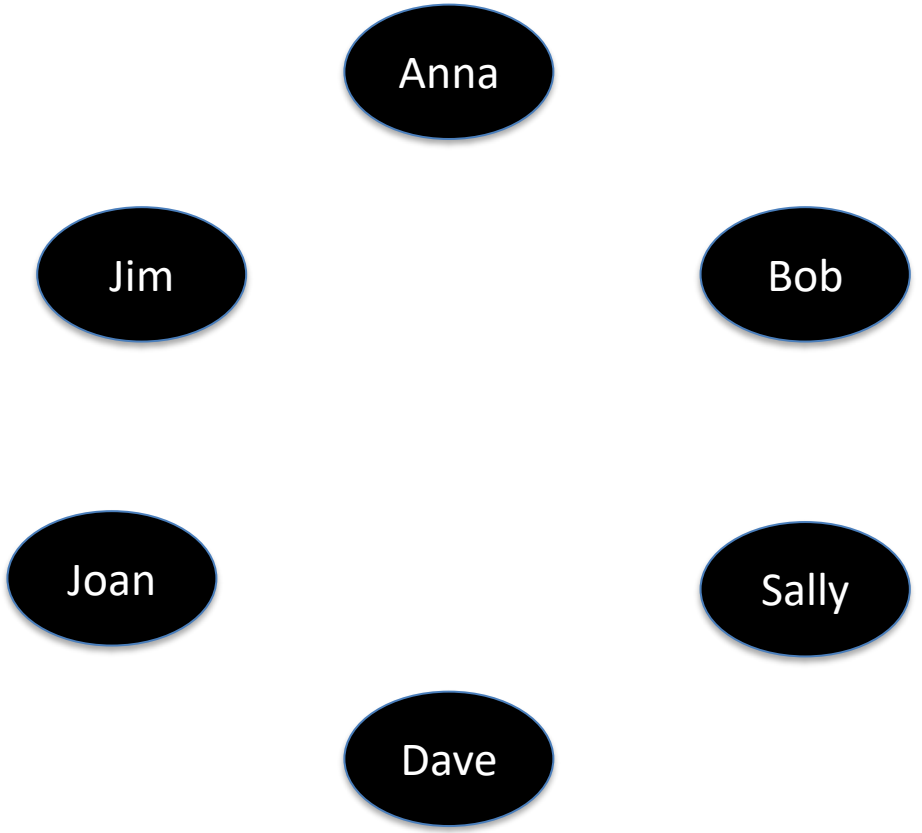
Task 3 - not started



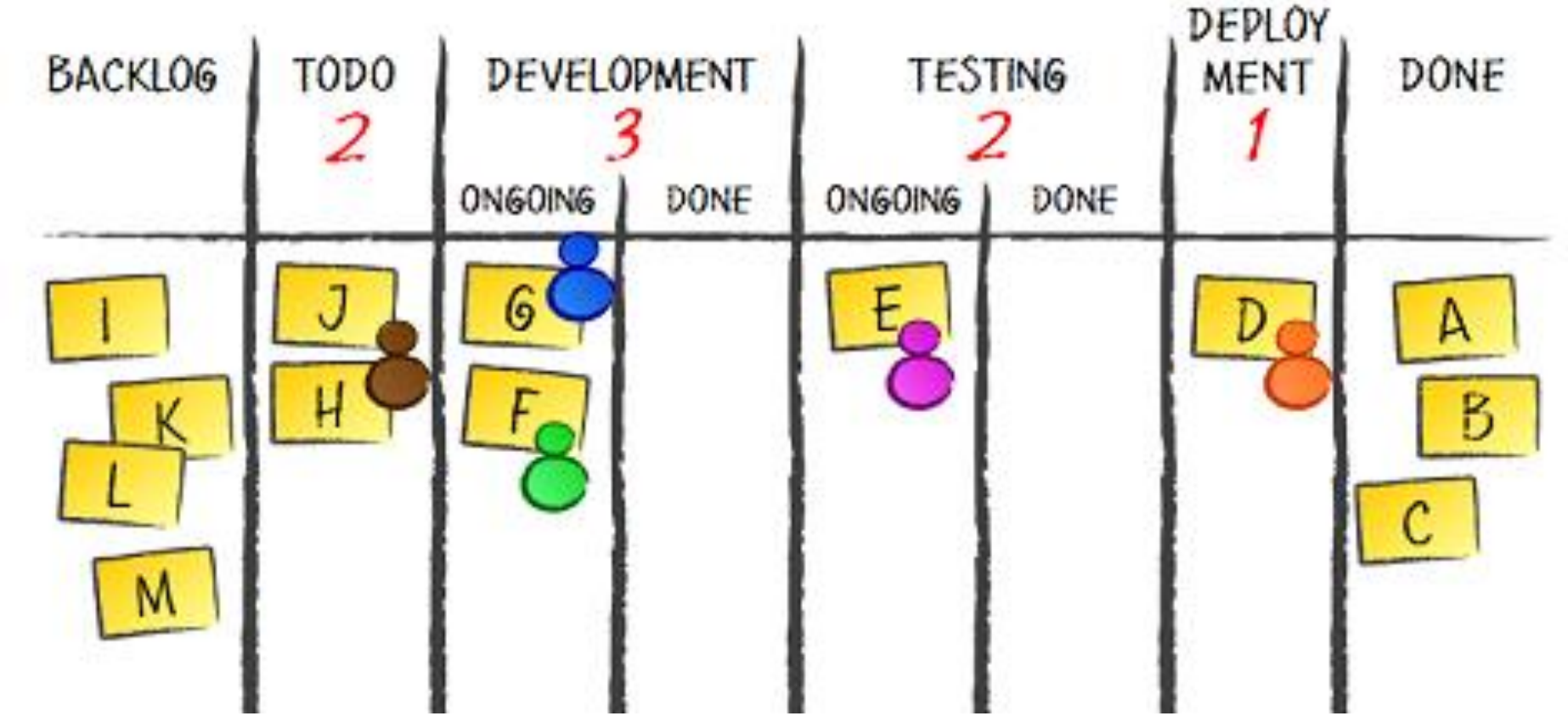
The Daily Stand Up



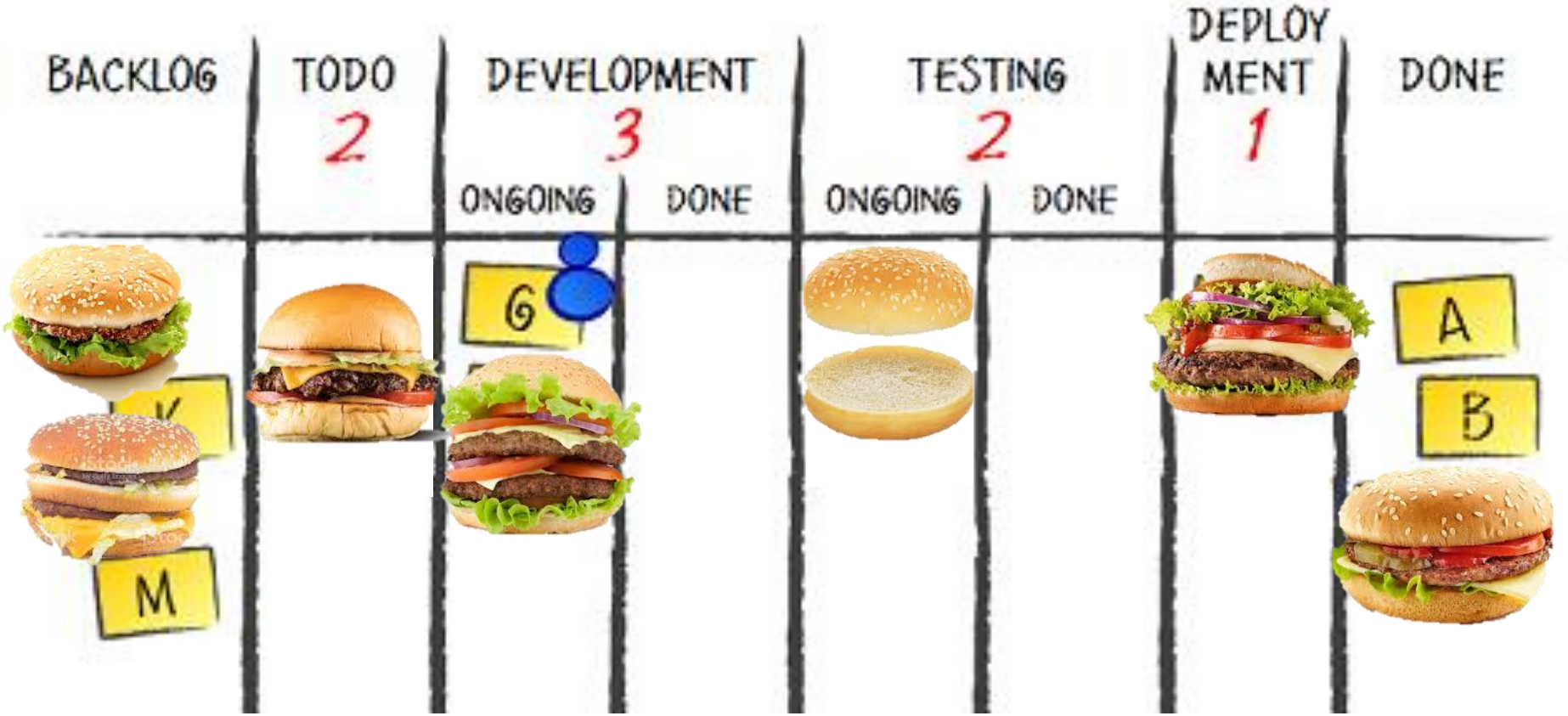
A Kanban Board!



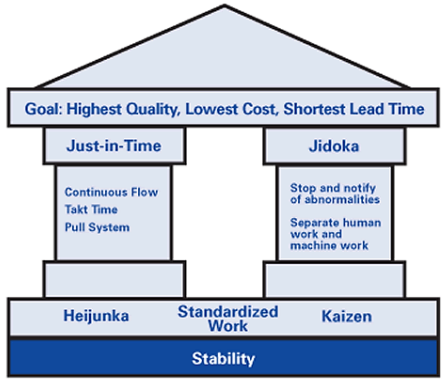
Kanban Board



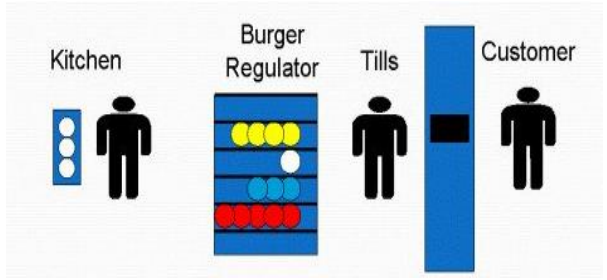
Kanban Board

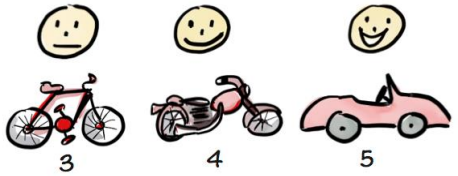


Summary

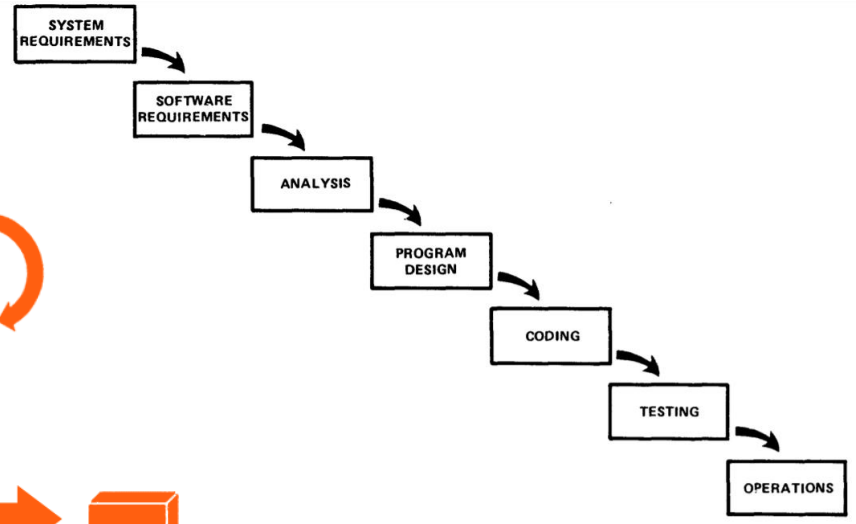


- **Lean**
  - Eliminate waste
  - Push vs Pull
  - Minimum Viable Product
  - Kanban





- **Agile**
  - Paradigm shift over “traditional” approach
  - Focus on flexible scope, continuous delivery, collaboration and trust
  - Scrum
  - Kanban (again)



Next: Guest Lecture

Practical Project Management

Jon Artus

Client Director, Softwire

Softwire

Week	Lectures		Seminars			Individual Report		
	Topic	Guest	Case Study	Exercises	Submission	Chapter	Submission	Marking
1	Specification			Specification				
2	Initiation		Selection		Pitch			
3	Scope / Time			Scope/Time				
4	PRINCE2	PRINCE2	Initiation			Ch.1 Initiation		
5	Budgeting			Budgeting				Self-assess
6	Lean/Agile 1	Waterfall / Agile	Planning			Ch.2 Planning		
7	Lean/Agile 2	Lean		Scrum/Kanban			Ch.1-2	
8	Risk	Risk / Finance	Monitoring			Ch.3 Execution		Review Ch.1-2
9	Teamwork	Large Projects	Prepare Presentation	Risk				
10	Revision				Presentation	Ch.4 Monitoring		
11							Ch.1-4	
Term 2								Review Ch. 1-4