

Lecture 2 — Software Processes

CPSC310 — Software Engineering

Learning Goals

By the end of this unit, you will be able to:

- Describe **benefits** of using a software process
- Describe the **waterfall** and **spiral** model including advantages, drawbacks
- Describe the importance of **agile** methods

Why do software projects fail?

- Unrealistic project goals
- Inaccurate estimates of needed resources
- Unmanaged risks
- Poor communication
- ... *so many reasons!!!*

When things go wrong...

Denver Baggage (mis)Handling



detailed report:

<http://calleam.com/WTPF/wp-content/uploads/articles/DIABaggage.pdf>

Denver Baggage (mis)Handling

summary: http://calleam.com/WTPF/?page_id=2086



System at a glance:

1. 88 airport gates in 3 concourses
2. 17 miles of track and 5 miles of conveyor belts
3. 3,100 standard carts + 450 oversized carts
4. 14 million feet of wiring
5. Network of more than 100 PC's to control flow of carts
6. 5,000 electric motors
7. 2,700 photo cells, 400 radio receivers and 59 laser arrays

Underestimation of complexity. Complex architecture. Changes in requirements. Underestimation of schedule and budget. Dismissal of advice from experts. Failure to build in backup or recovery process to handle situations in which part of the system failed. The tendency of the system to enjoy eating people's baggage.

The Beginning...

Nov 1989	Work starts on the construction of the airport	
Oct 1990	City of Denver engages Breier Neidle Patrone Associates to analyse feasibility of building an integrated baggage system. Reports advises that complexity makes the proposition unfeasible	risk flagged; ignored.
Feb 1991	Continental Airlines signs on and plans on using Denver as a hub	
Jun 1991	United Airlines signs on and plans on using Concourse A as a hub	scale changed
Jun 1991	United Airlines engages BAE Systems to build an automated baggage system for Concourse A. BAE was a world leader in the supply, installation and operation of baggage handling equipment	
Summer 1991	Airport's Project Management team recognizes that a baggage handling solution for the complete airport was required. Bids for an airport wide solution are requested	
Fall 1991	Of the 16 companies included in the bidding process only 3 respond and review of proposals indicate none could be ready in time for the Oct 1993 opening. The 3 bids are all rejected	risks flagged; ignored
Early 1992	Denver Airport Project Management team approach BAE directly requesting a bid for the project	
Apr 1992	Denver Airport contracts with BAE to expand the United Airlines baggage handling system into an integrated system handling all 3 concourses, all airlines, departing as well as arriving flights. In addition system is to handle transfer baggage automatically. Contract is hammered out in 3 intense working sessions	hasty contract
Aug 1992	United Airlines changes their plans and cuts out plans for the system to transfer bags between aircraft. Resulting changes save \$20m, but result in a major redesign of the United Airlines portion of the system. Change requests are raised to add automated handling of oversized baggage and for the creation of a dedicated ski equipment handling area	requirements change
Sep 1992	Continental requests ski equipment handling facilities be added to their concourse as well	requirements change
Oct 1992	Chief Airport Engineer, Walter Singer dies. Mr Singer had been one of the driving forces behind the creation of the automated baggage system	guru dies

...The End

Jan 1993	Change orders raised altering size of ski equipment claim area and adding maintenance tracks so carts could be serviced without having to be removed from the rails	requirements change
Feb 1993	Target opening date shifted from 31 Oct 93 to 19 Dec 93 and soon thereafter to 9 Mar 94	
Sep 1993	Target opening date is shifted again, new target date is 15 May 1994	
31 Oct 1993	Original target for opening	
19 Dec 1993	Second target for opening	delays
Jan 1994	United Airlines requests further changes to the oversize baggage input area	
9 Mar 1994	Third target for opening	
Mar 1994	Problems establishing a clean electrical supply results in continual power outages that disrupt testing and development. Solution requires installation of industrial filters into the electrical system. Ordering and installation of the filters takes several months	technical challenges
Apr 1994	Airport authorities arrange a demonstration for the system for the media (without first informing BAE). Demonstration is a disaster as clothes are disgorged from crushed bags	
Apr 1994	Denver Mayor cancels 15 May target date and announces an indefinite delay in opening	more delays
May 1994	Logplan Consulting engaged to evaluate the project	
15 May 1994	Fourth target for opening	
May 1994	BAE Systems denies system is malfunctioning. Instead they say many of the issues reported to date had been caused by the airport staff using the system incorrectly	BAE blames the user
Aug 1994	System testing continues to flounder. Scope of work is radically trimmed back a Logplan's recommendation airport builds a manual tug and trolley system instead	manual system used instead
Aug 1994	City of Denver starts fining BAE \$12K per day for further delays	BAE fined for overtime
28 Feb 1995	Actual opening	done! but badly
Aug 2005	In order to save costs the system is scrapped in favour of a fully manual system. Maintenance costs were running at \$1M per month at the time.	scrapped

So what happened?

- **bad planning:** They left it late! The software production started only 17 months before the scheduled opening
 - In Munich, engineers spent **two years testing** a similar but much smaller system. And even that system is not glitch free (nothing ever is - but at least it's 24/7 operational)
- **physical problems:** Most buildings were built before the baggage system was designed, meaning the baggage system had to adapt to an architecture that wasn't a good fit (sharp turns, narrow corridors)
- **change of management:** Death of the driving force of the project (you'd think this was atypical, but gurus often leave a company mid-way through a project, leaving the project somewhat stranded if planning isn't good)
- **accepting changing requirements:** alterations to baggage sizes, types, paths, etc — and the contractor said a firm “yes” to all these changes!
- **lack of experience:** this was the first time BAE had built a system like this. But for some reason they didn't choose to ask for outside advice from the Munich baggage system engineers who might have provided insights and helped mitigate risk.

Software Project Risks

http://www.iaeng.org/publication/IMECS2011/IMECS2011_pp732-737.pdf

- 6 dimensions of risks:
 - **USER:** resistance to change; conflicts between them; negative attitudes towards the project; lack of commitment; lack of cooperation
 - **REQUIREMENTS:** continually changing; inadequately identified; unclear; incorrect
 - **PROJECT COMPLEXITY:** new technology; high technical complexity; immature technology; first use of technology
 - **PLANNING & CONTROL:** poor process oversight; inadequate estimation resources; poor planning; unclear milestones; inexperienced pm's; ineffective communication
 - **TEAM:** lack of experience; lack of training; lack of specialised skill
 - **ORGANIZATIONAL ENVIRONMENT:** change of management during the project; unstable organisation; ongoing restructuring

Projects need a good plan

- A ***software process*** is a structured set of activities to develop a software system.
- Defines who is doing what, when and how to reach a goal.

Software Process

- Processes have descriptions that discuss
 - **Products** (the outcome of a process activity)
 - **Stakeholders** (people who care about the outcome)
 - (managers, developers, customers, testers)

what are some major types of software customers?

Software process

- Many different software process
- All include:
 - requirements elicitation
 - architectural design
 - detailed design
 - implementation
 - integration
 - testing

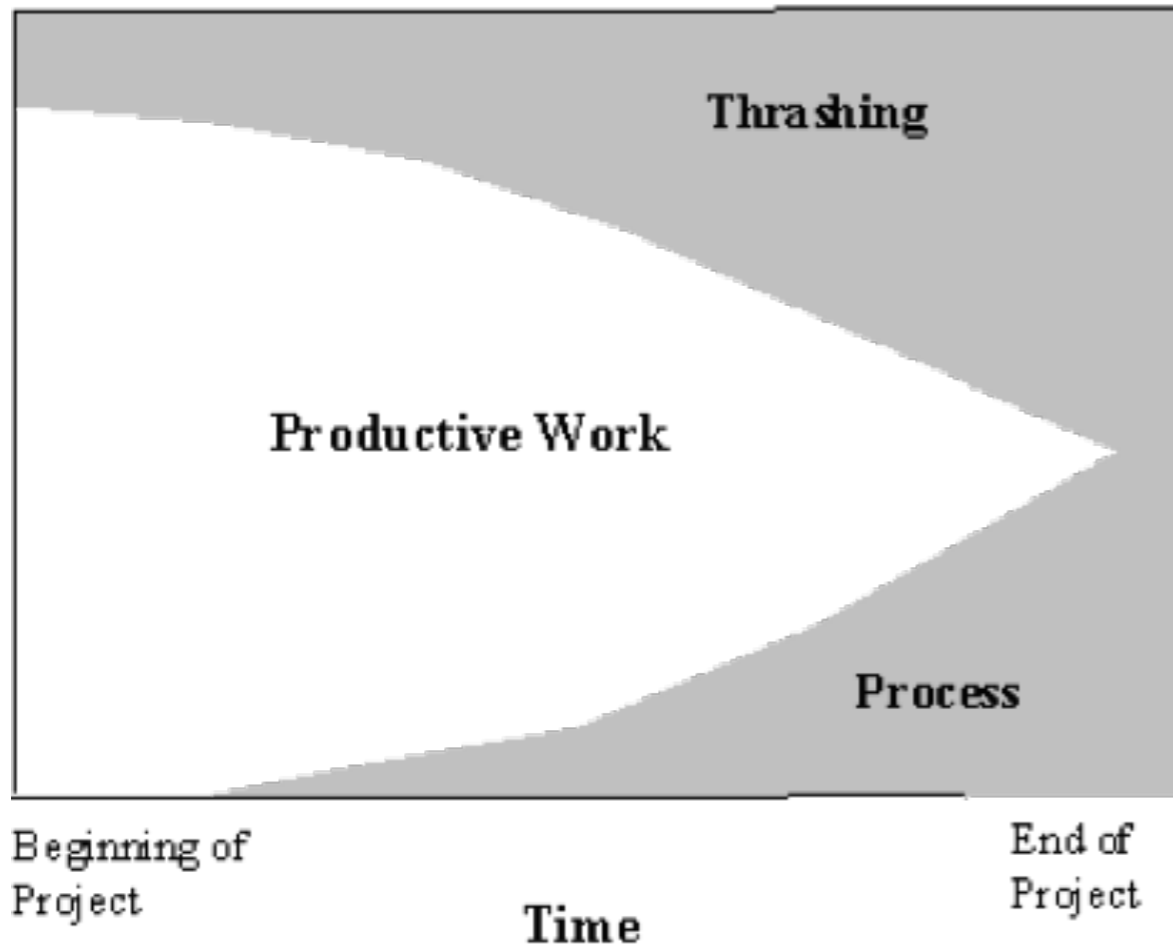
the goal for each of these activities is to:

- mark out a clear set of steps
- produce tangible item(s)
- allow for review of work
- specify actions to perform next

Is Process Worth It?

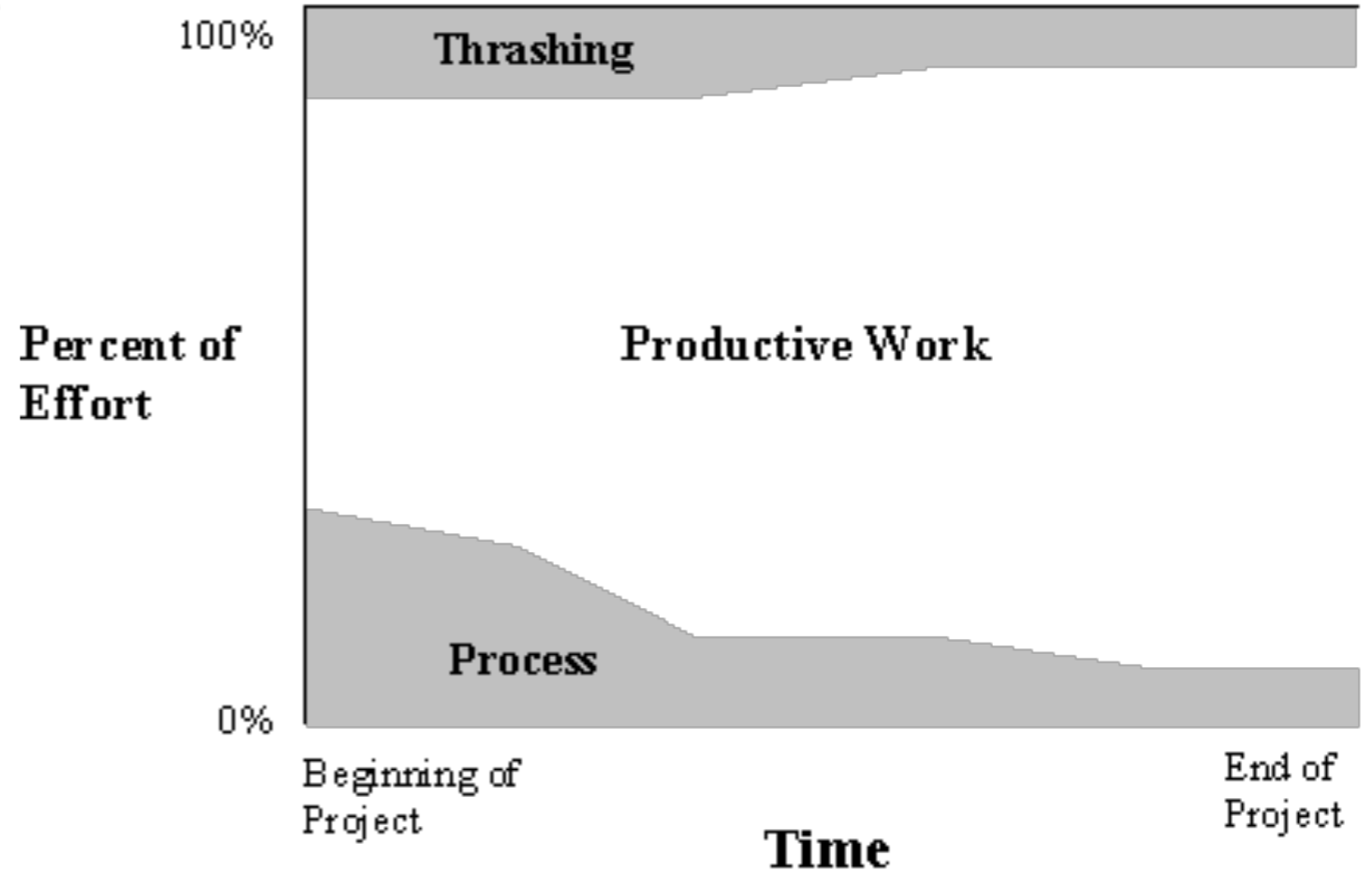
<http://www.stevemcconnell.com/articles/art09.htm>

Process Phobic Team




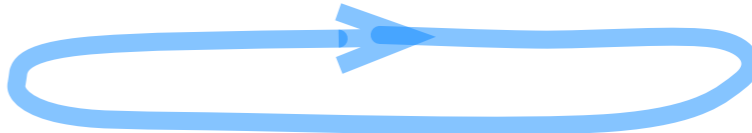
“When a project has paid too little early attention to the processes it will use, by the end of a project developers feel they are spending all of their time in meetings and correcting defects and little or no time extending the software.”

Process-Oriented Team



“During the first few weeks of the project, the process-oriented team will seem less productive than the process-phobic team... By the end of the project, the process-oriented team will be operating at a high-speed hum, with little thrashing, and performing its processes with little conscious effort.”

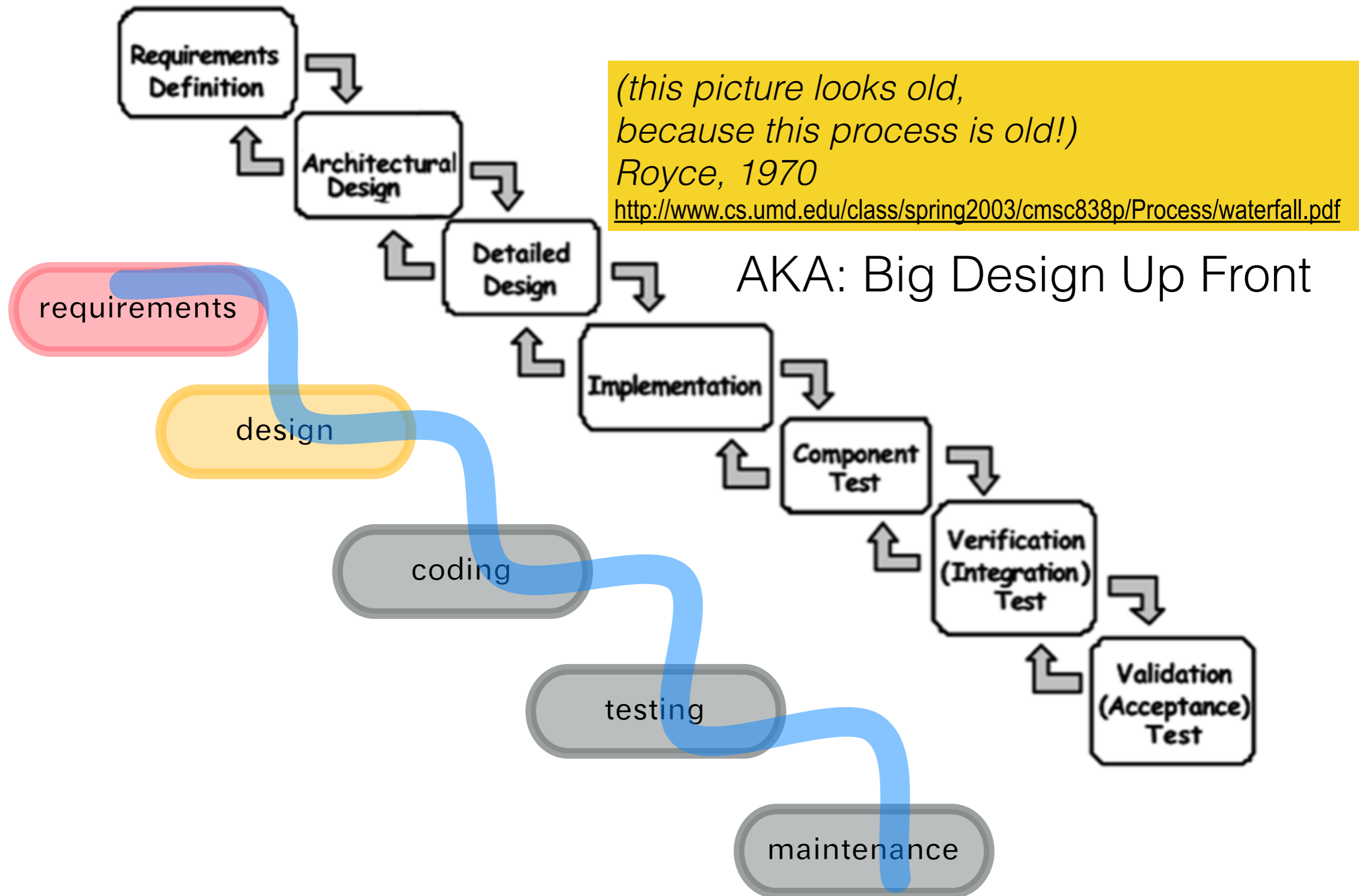
Software Process Models

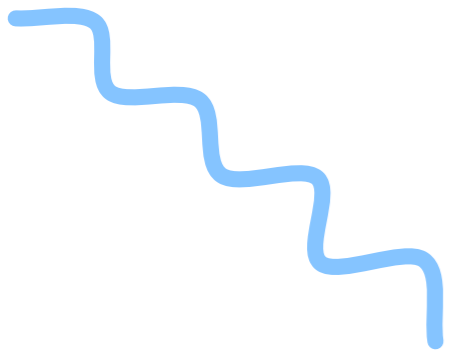
- Waterfall model (sequential)
 - separate and distinct phases of specification and development
- Spiral model 
 - incremental prototyping with risk management
- Agile models / principles 
 - iterative approach with high customer communication

Some models are better for some types of projects than others. Often models are combined to build a tailored process for building a certain type of product.

Project leads need to be able to choose and tailor a model and assess risk
Developers need to understand processes and work within them

Waterfall Model



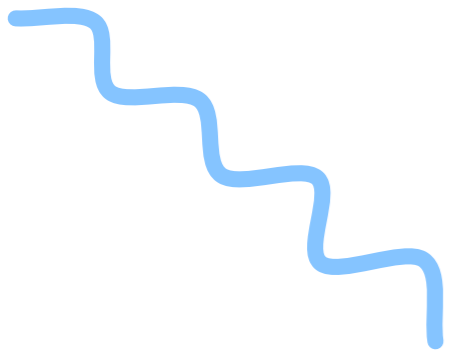


Waterfall Model

AKA: Big Design Up Front

Benefits:

- Good for well-understood, complex projects
 - Tackles all planning up front
 - No midstream changes = efficient process
- Provides support for an inexperienced team
 - Orderly, sequential, easy-to-follow model
 - Relatively slow progress
 - Reviews at each stage



Waterfall Model

AKA: Big Design Up Front

Drawbacks:

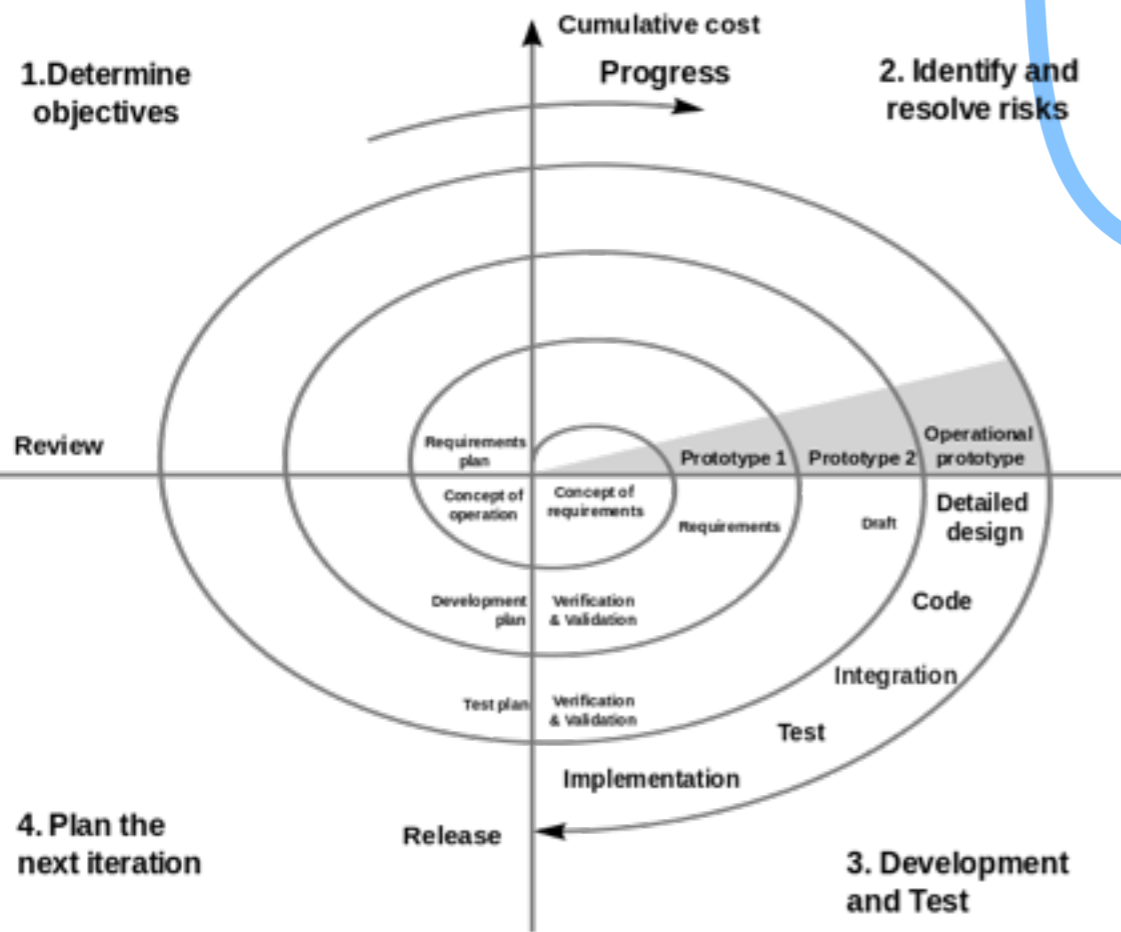
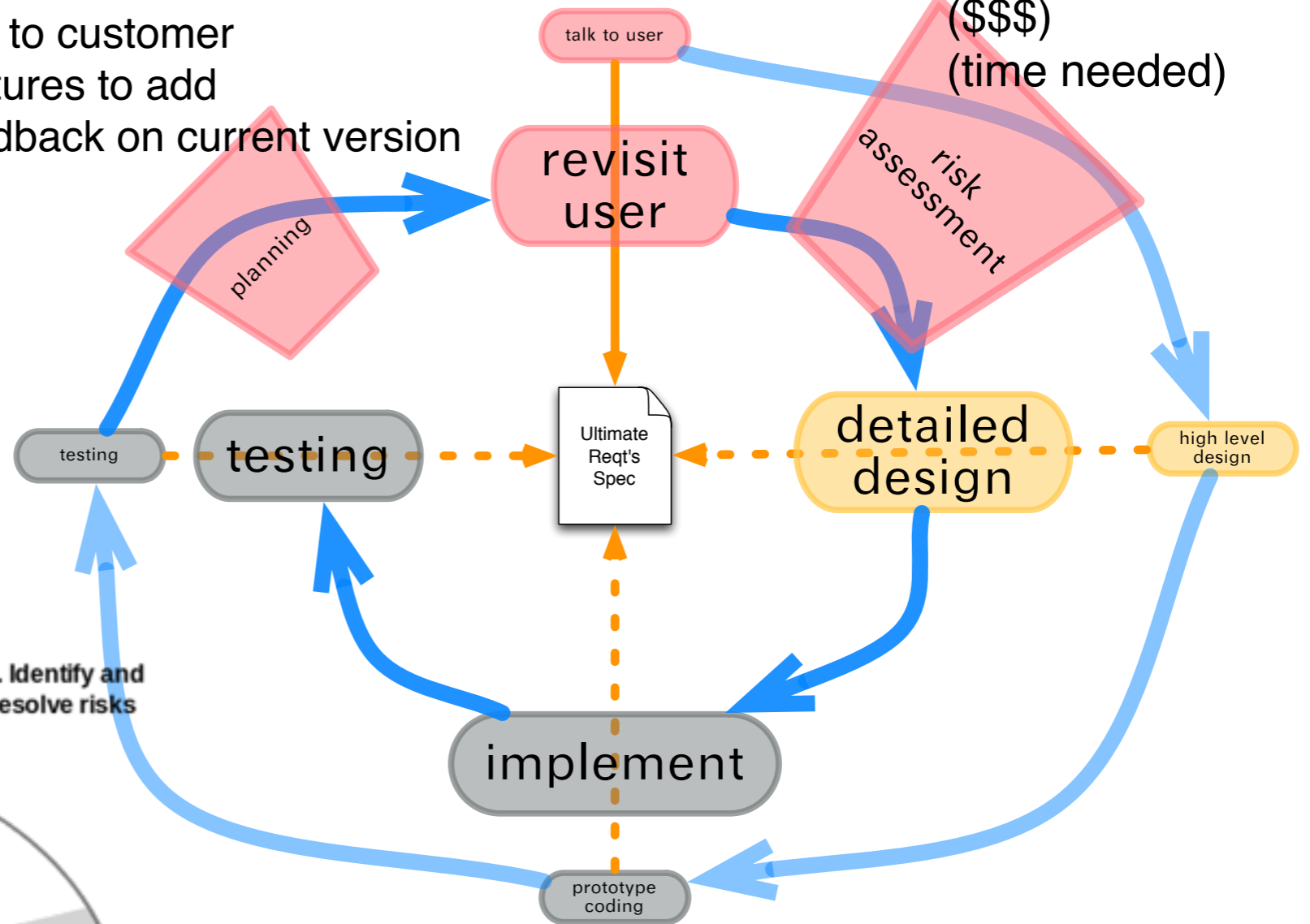
- Getting requirements right up front.
- Hard to manage risks (a bit of a gamble)
- Reactive to errors but not to changing requirements



Spiral Model

talk to customer
features to add
feedback on current version

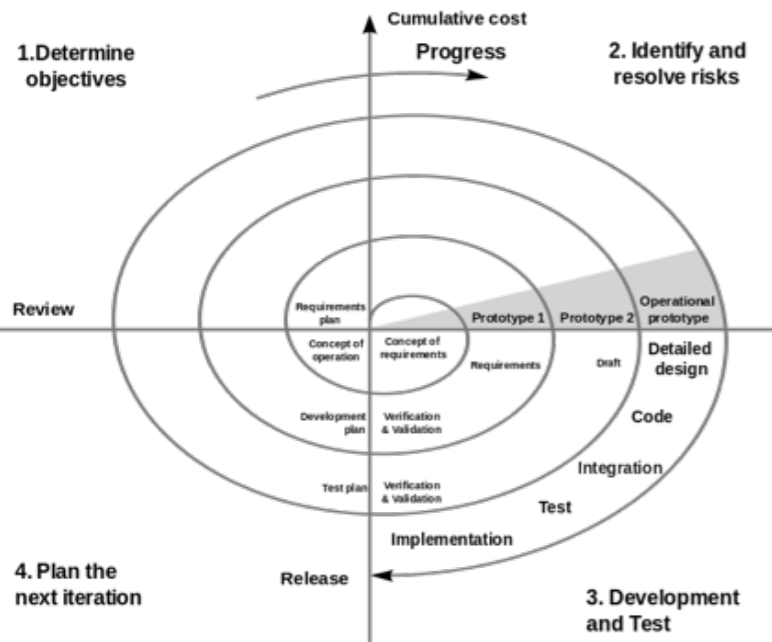
risks/problems identified
(\$\$\$)
(time needed)



Boehm B, "A Spiral Model of Software Development and Enhancement",
ACM SIGSOFT Software Engineering Notes, "ACM", 11(4):14-24, August 1986



Spiral Model



- Each cycle
 - Ends with demoable prototype
 - Lessons from performing the cycle are reviewed and incorporated for next cycles
- For each cycle
 - Sectors encompass more of a complete Waterfall
 - Prototype becomes more of a complete product
- In the last cycle
 - A complete Waterfall emerges

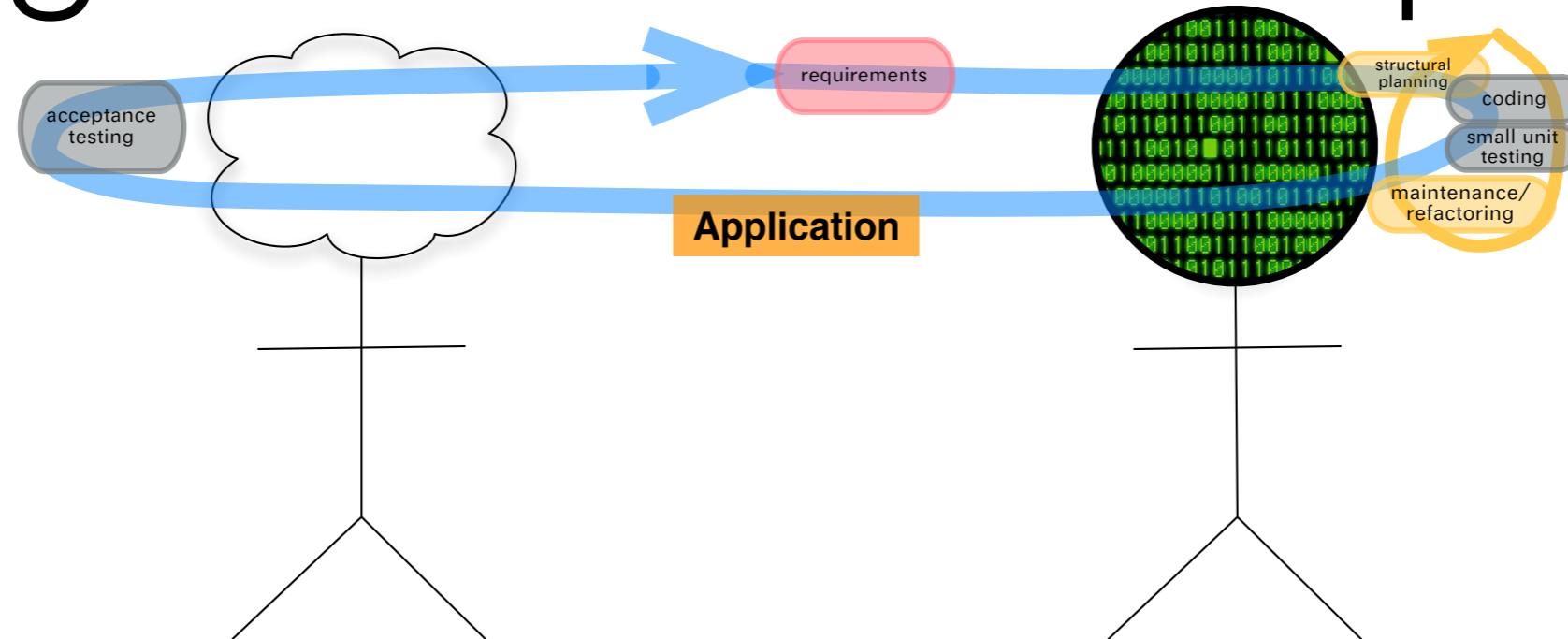
Benefits:

Manages risk by repeated prototyping
Requirements changes incorporated iteratively

Drawbacks:

High administrative overhead
Can be overly conservative if you have high confidence in the project outcome

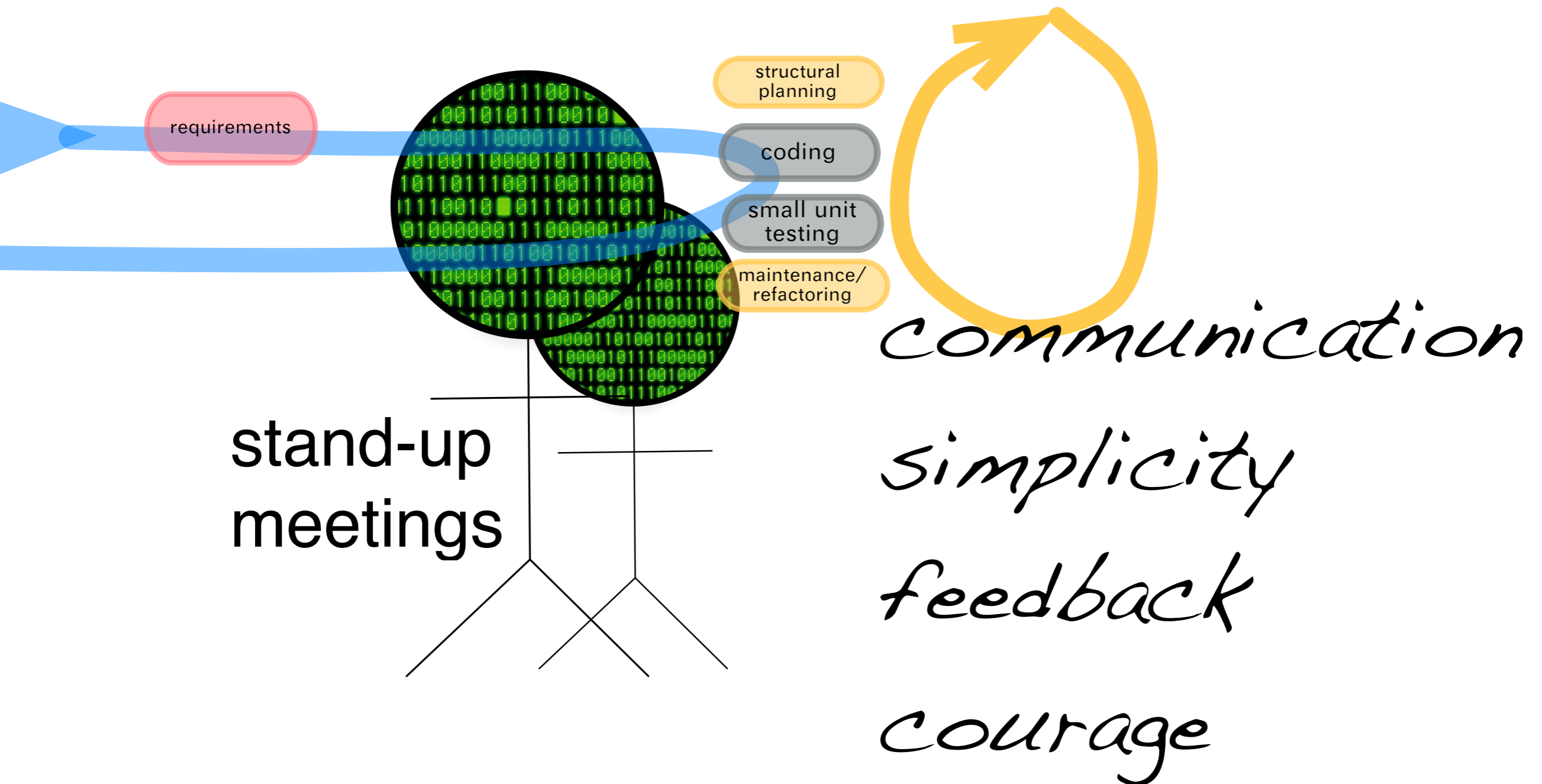
Agile Models/Principles



- The goal of agility: to develop software in the face of ***changing environment*** and ***constrained resources***
- More a **set of principles** than a fixed model; many variations of agile processes
- Developers are indoctrinated with the principles and then trusted to act with less oversight

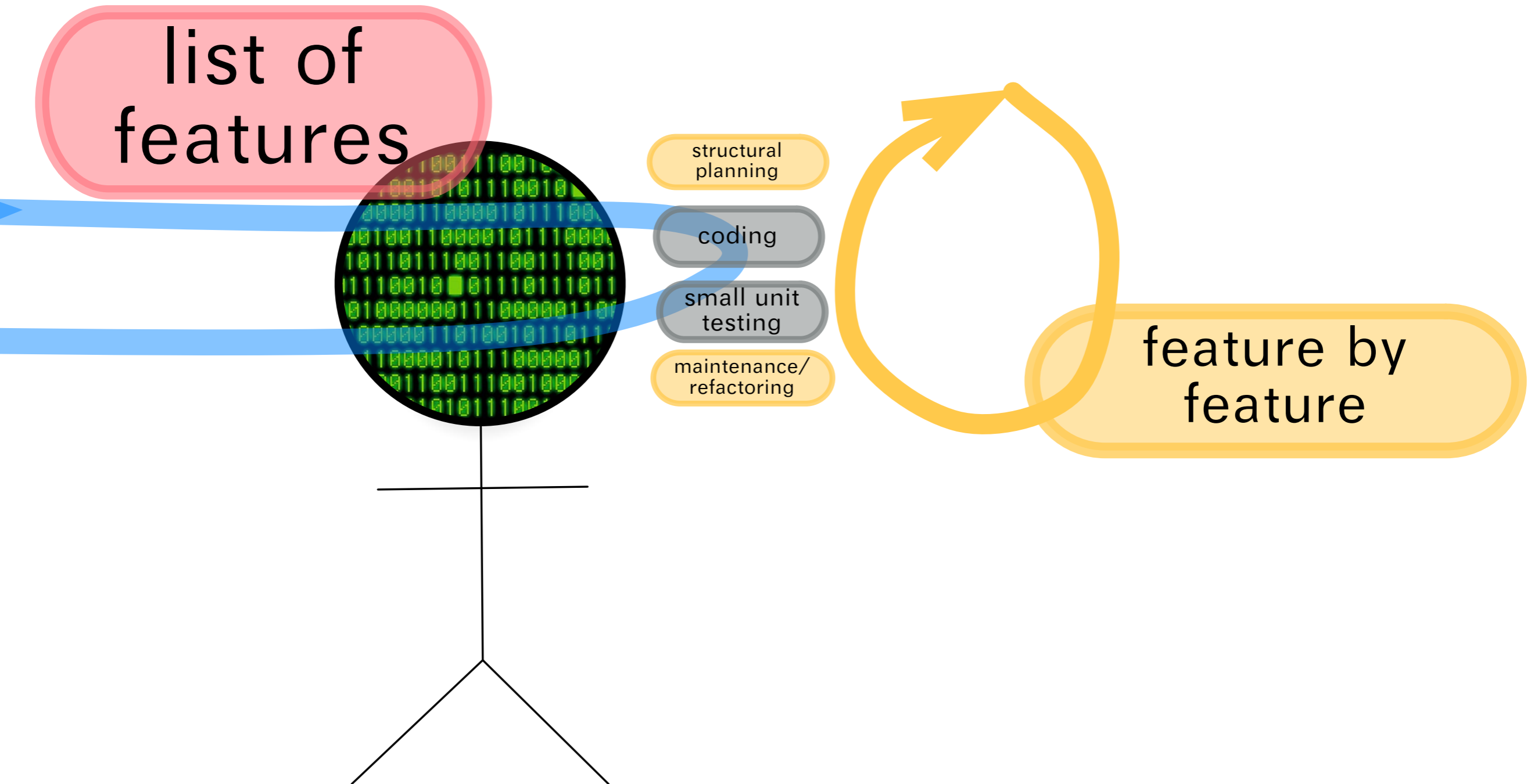
Flavours of Agile...

eXtreme Programming



Flavours of Agile...

Feature Driven Development (FDD)



Summary

- Why a software process can be important
- Definition of a software process
- There exist many different process models
- Basic models: Waterfall, Spiral
- Agile models: incremental and iterative

More on Agile...

We will be getting more into agile principles and models in the next lecture...