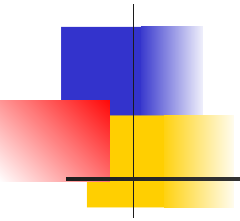


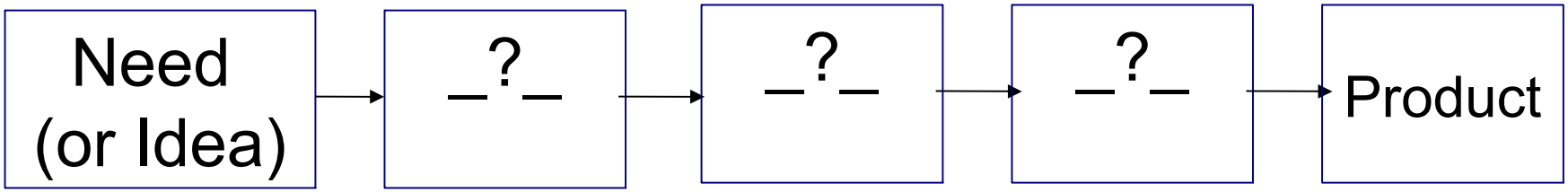
Intro to Software Processes

and the Software Development Life Cycle

Goal of Software Development

Produce a software product that fulfills a need or realizes an idea.

What are the Steps?



What are the major steps or activities you would need to do?

List major activities that would apply to almost any software project.

Activities

Creating software involves

- elicit requirements
- analysis & specification
- design
- construction & testing
- validation
- documentation
- maintenance
- improvement

Managing the project involves

- planning
- obtaining resources
- tracking progress
- resolving problems
- analyzing results
- closing the project

Process

Process -

a [systematic] series of actions to achieve a particular result

Software process - a method for producing software

Software Process according to experts

A software process is a sequence of activities that leads to production of a software product.

-- Ian Summerville, *Software Engineering*, 9 Ed.

...a collection of activities, actions, and tasks that are performed to create [software].

-- Roger Pressman, *Software Engineering: A Practitioner's Approach*, 7 Ed.

Do You Have a Software Process?

What is your software process?

(discussion)

What did you do to create:

- Programming 2 project?
- Exceed Camp project?

Do You Have a Software Process?

Yes!

Everyone who develops software uses a process.

"Never thought about it" *...*

process is *implicit* or *informal*

"***It's different for each project***" ...

ad hoc process

Why Define a Software Process?

Why not *just do it?*

Realities of Software

Software is plagued by defects, over-budget, schedule overrun, and complete failure of projects to deliver.

1. Change can be required almost anytime during a project.
2. Useful software is complex.
3. Useful software must *evolve* (more change)
4. Communication problems plague software
   - between devs and customer
   - within development team
   - implicit assumptions are often not true

Common Project Outcomes (failures)

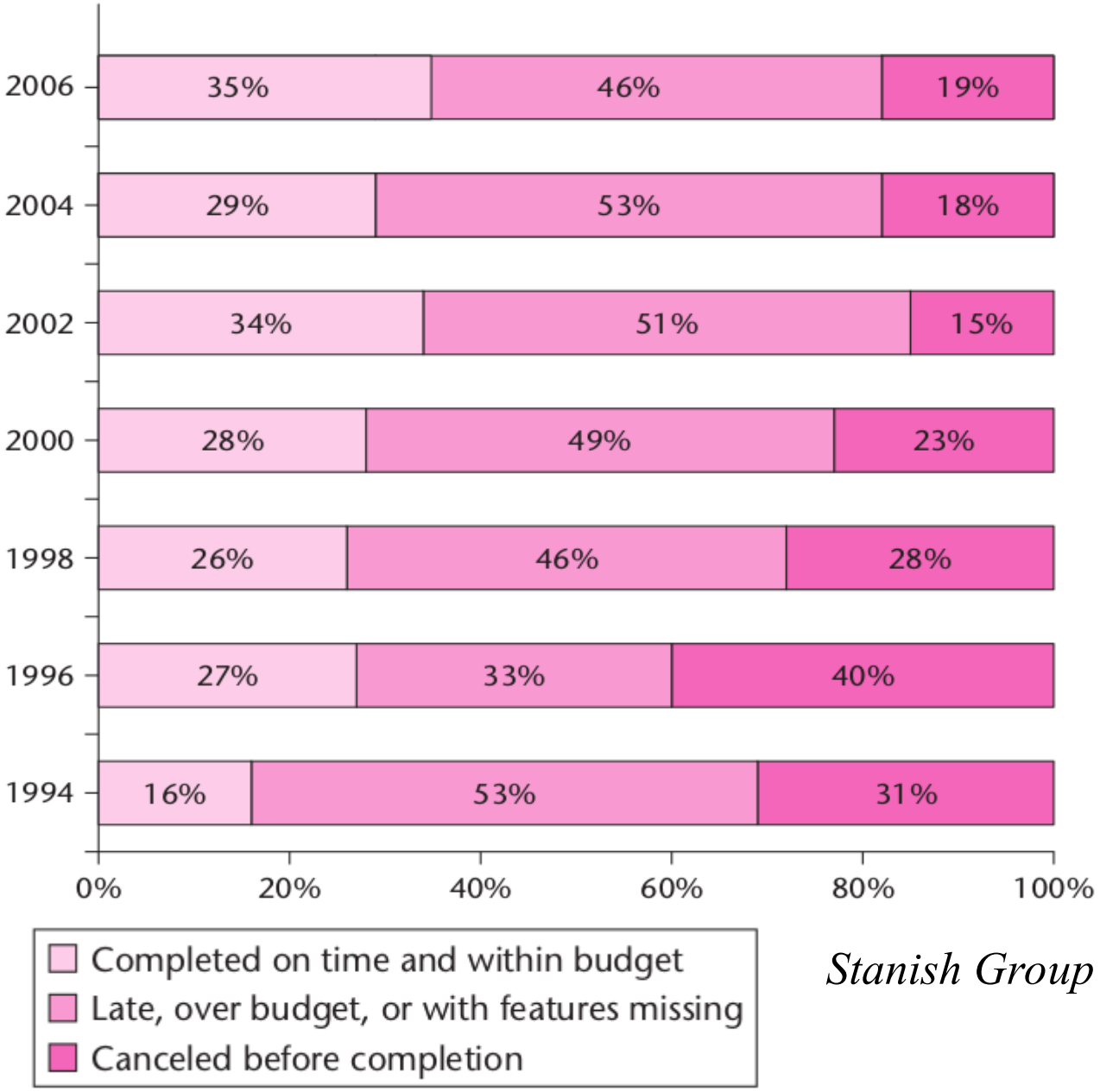
1. Project is late and over-budget.

2. Software does not do what the customer wants.

3. Excessive defects.

4. Project is canceled.

Software Project Failure over Time



Stanish Group annual CHAOS report

Britain Abandons NHS IT Project

After 10 years and 11 Billion pounds (450,000,000,000 Bt), the British government abandoned a huge IT project for the National Health System (NHS) in 2011.

Some components continue to be developed, but they are all late and over-budget.

Why? What Happened?

https://www.henricodolfing.com/2019/01/case-study-10-billion-it-disaster.html

https://www.computerweekly.com/opinion/Six-reasons-why-the-NHS-National-Programme-for-IT-failed

Microsoft Windows Critical Flaws

Each month in 2020, Microsoft set a new record for the number of critical vulnerabilities disclosed & patched.

Microsoft programmers have been working on Windows code for almost 20 years -- if we take Windows 7 as the starting point.

Yet Windows still contains hundreds or thousands of critical vulnerabilities.

Why?

Causes of Project Failure

1. Poor communication.
2. Unrealistic schedule or budget. Forced deadlines.
3. Unclear requirements.
4. Excessive change in requirements.
5. Unwillingness to accept change.
6. Not monitoring *actual* project progress regularly.
7. Insufficient developer skills.

Benefits of a Defined Process

- Saves Time - don't rediscover how to execute each project
- Enable Planning and Tracking
- Basis for Estimation - you collect data for each activity and task from previous projects and learn
- Repeatable results
- Improve the Process - it must be defined before you can examine and improve it

4 Factors in Development Speed

1. **People**

   ability, knowledge, skills, motivation

2. **Process**

   promotes effective work or hinders it

   helps team stay on track? quality focus?

3. **Product**

   Size and characteristics, nature of requirements

4. **Technology**

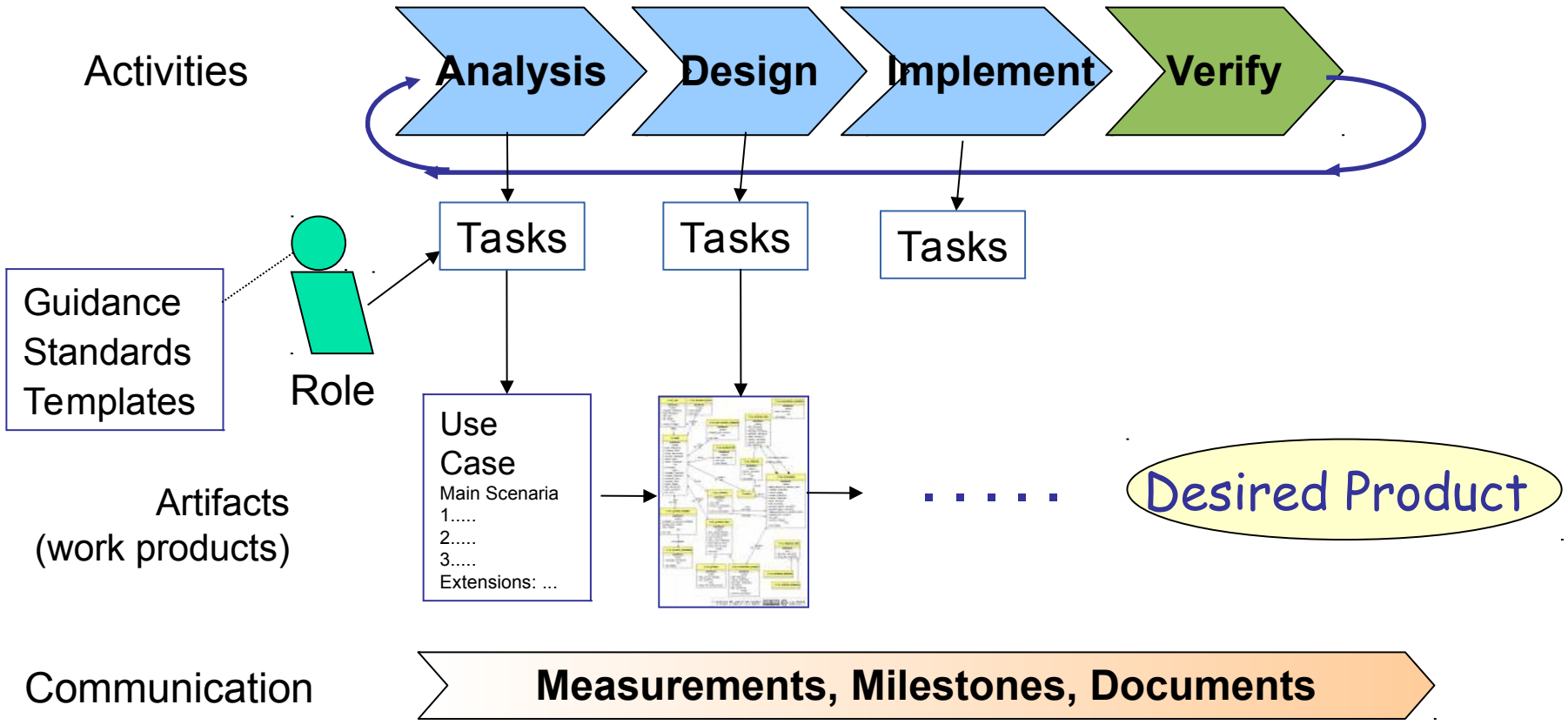
   Language and software frameworks

   Tools

Software Process Model

Process consists of activities

... and a lot of other stuff



Activities

Process consists of several activities.

Activities may be performed differently on different projects, and sometimes not at all.

Major activities:

- requirements specification
- modeling & design
- construction
- validation
- deployment

[Major activities listed by Summerville & Pressman.]

Tasks

Activities are large and general.

An activity is broken down into (concrete) tasks.

Some tasks during Construction:

- iteration planning
- backlog selection & estimation
- detail design
- coding
- unit testing
- integration testing

Activity May Subdivide into 2 Levels

In Pressman, an activity consists of actions divided into tasks.

Construction Activity

Action: *iteration planning meeting*

Tasks:

- review & prioritize items in product backlog
- select items for this iteration (sprint)
- estimate items
- assign "done" (test) criteria to each item
- design software for this iteration

How to do it? What to produce?

"Activities", "actions", and "tasks" should make *progress toward finishing* the project.

What to do? How to do it?

Need a task description and guidance

What to produce?

Every task should have an output -- a work product

Is the work correct?

Need a way to evaluate the work product

Common Process Models

Code and Fix

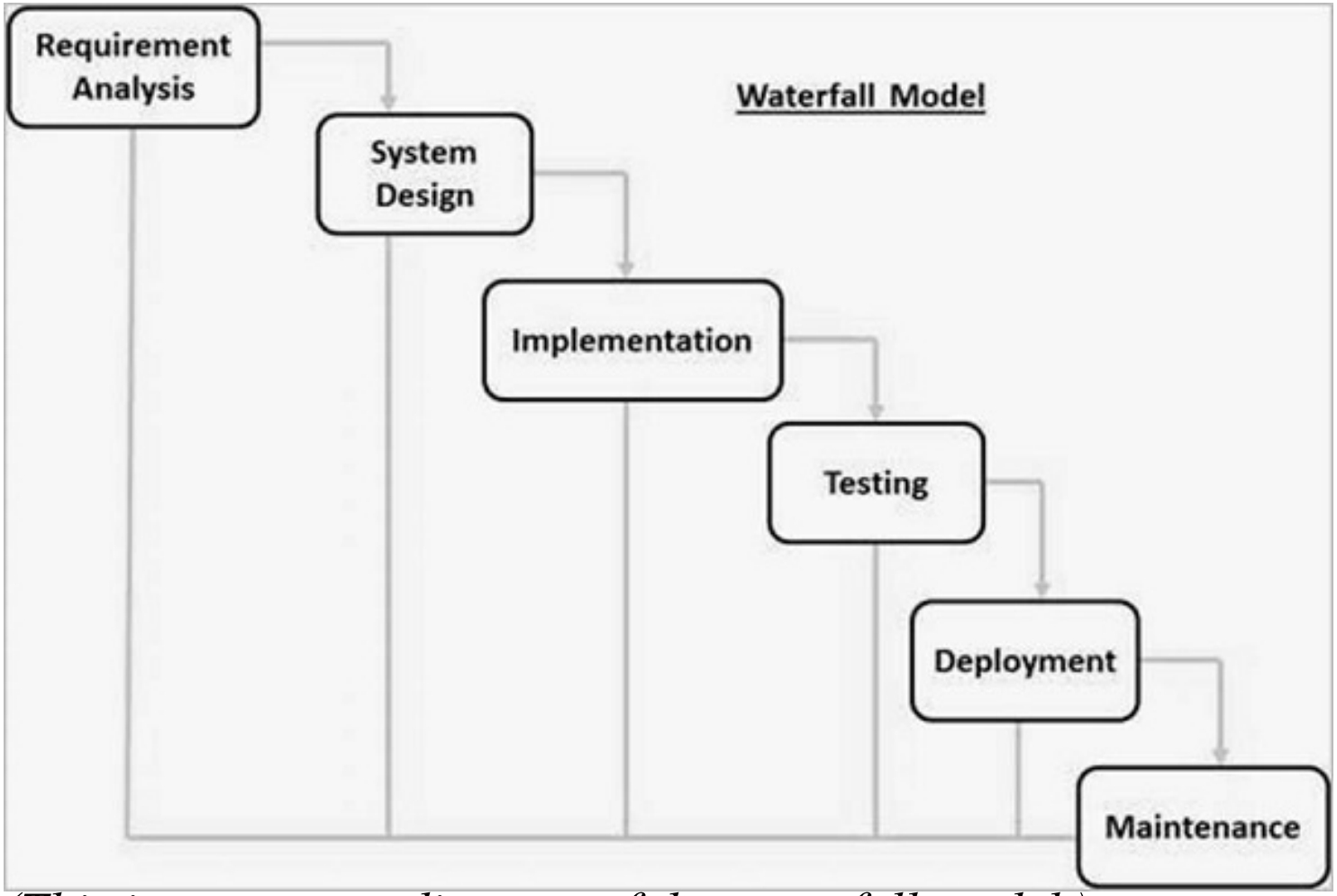
- The most common software development process
- Little or no planning and design.

1. think about the problem, write ideas on paper
2. start coding
3. run it. fix the code.
4. add another feature. As code grows I need to rewrite some parts to support each new feature.
    - modify the code for new feature
    - goto step 2.

*My software process since high school (*Fortran*)*

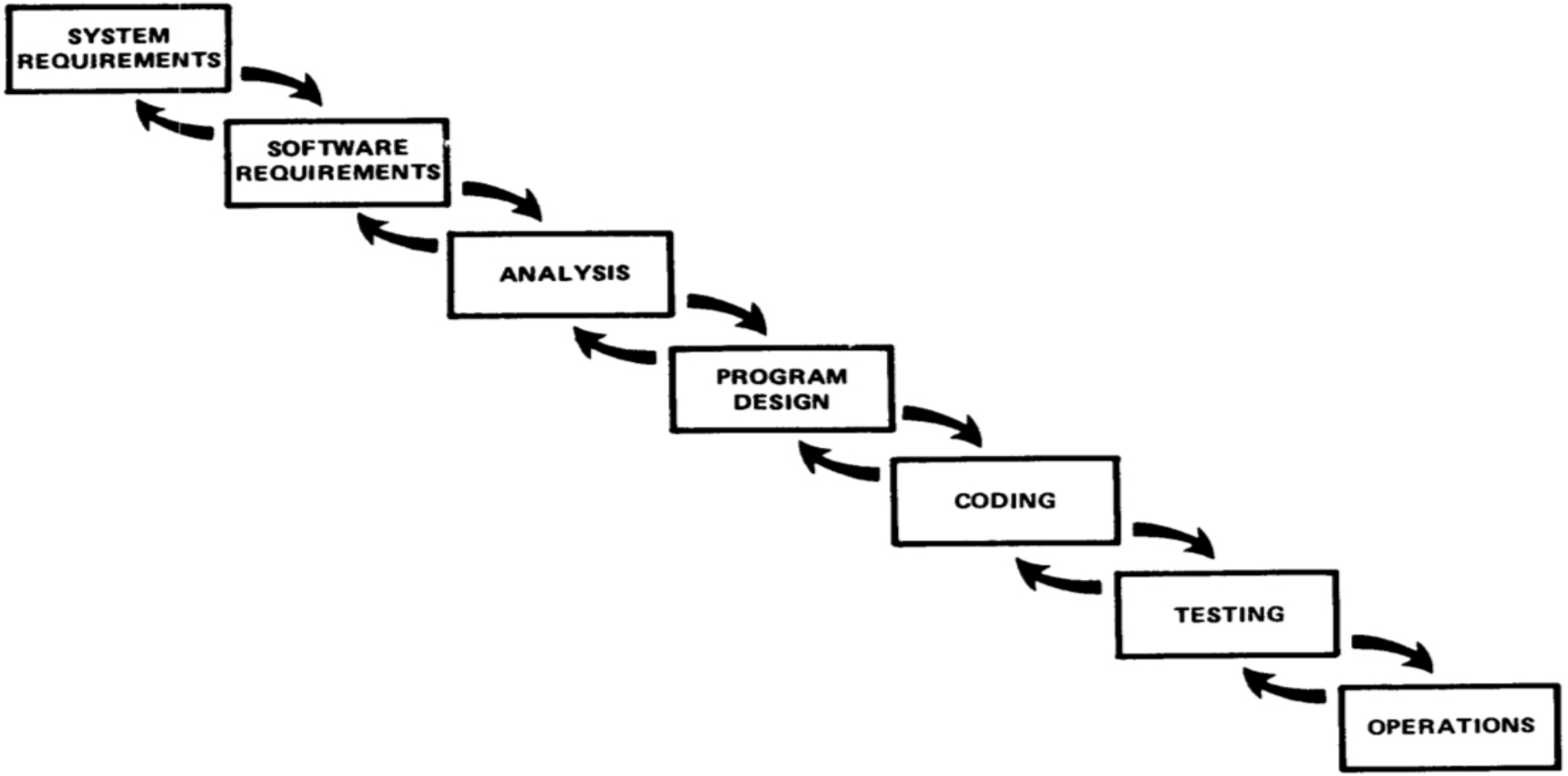
What if we do the activities in order?

Similar to a civil engineering project.



(This is a common diagram of the waterfall model.)

The Original Waterfall Model



Winston Royce, *Managing the Development of Large Software Systems* (1970)

Waterfall is still widely used.

What Could Go Wrong?

Common Problems with Waterfall

What would be effect on project if ...

1. You miss some requirement(s).

2. You misunderstand requirements, so the design is not what the customer wants.

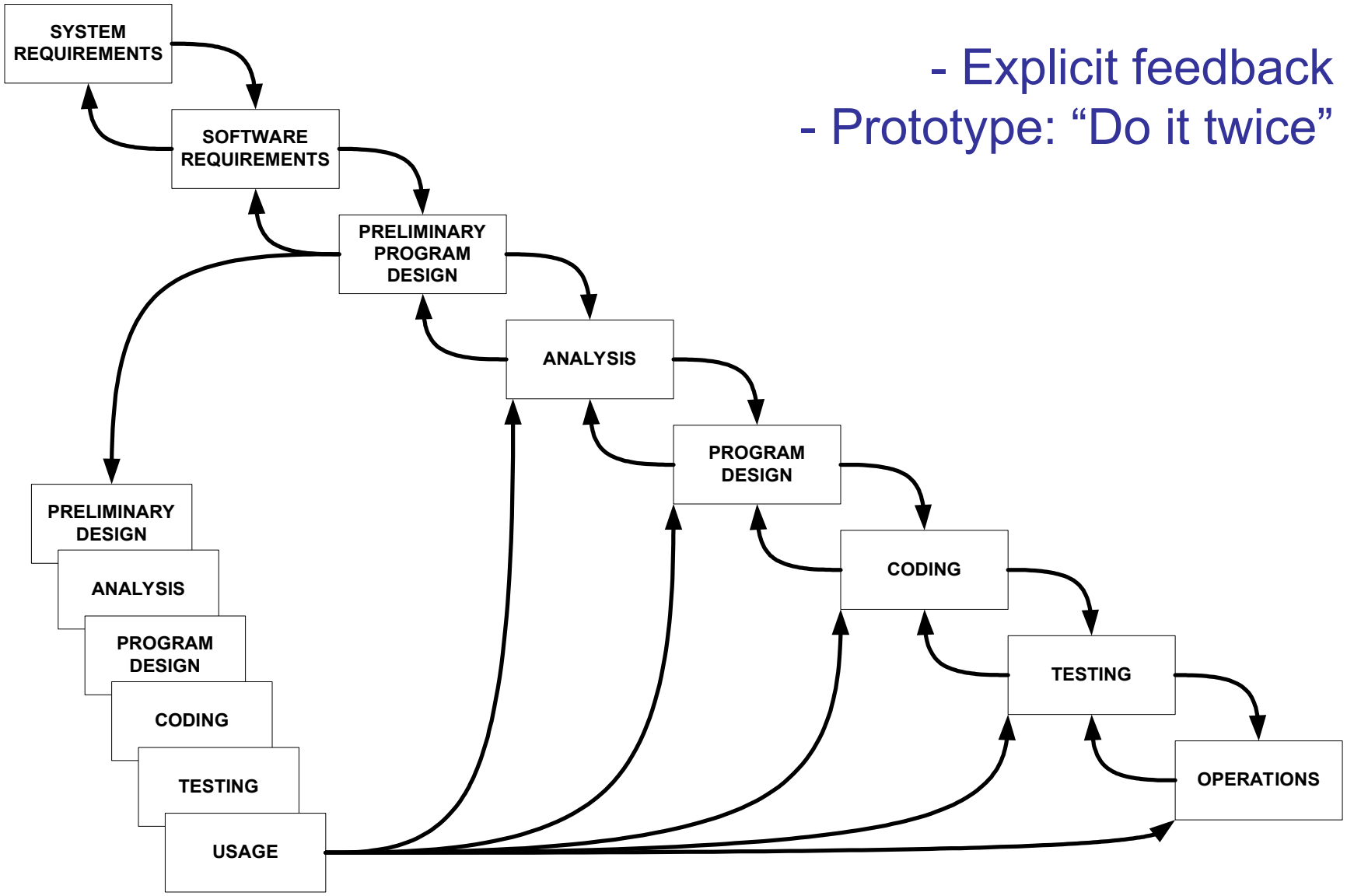
3. The solution you chose can't handle the requirements.

4. Lots of defects during development, discovered only late during testing.

How to Avoid These Problems?

- *Early* Feedback
- *Early* Testing
- *Continuously* review actual versus planned progress
- *Involve customer* at key points during project
- *Incremental delivery* of functionality.
- *Analyze* results and take corrective action

Royce Waterfall Model with Prototype



- Explicit feedback
- Prototype: “Do it twice”

Project Phase = Process Activity

In Waterfall, major activities are *phases* of project...

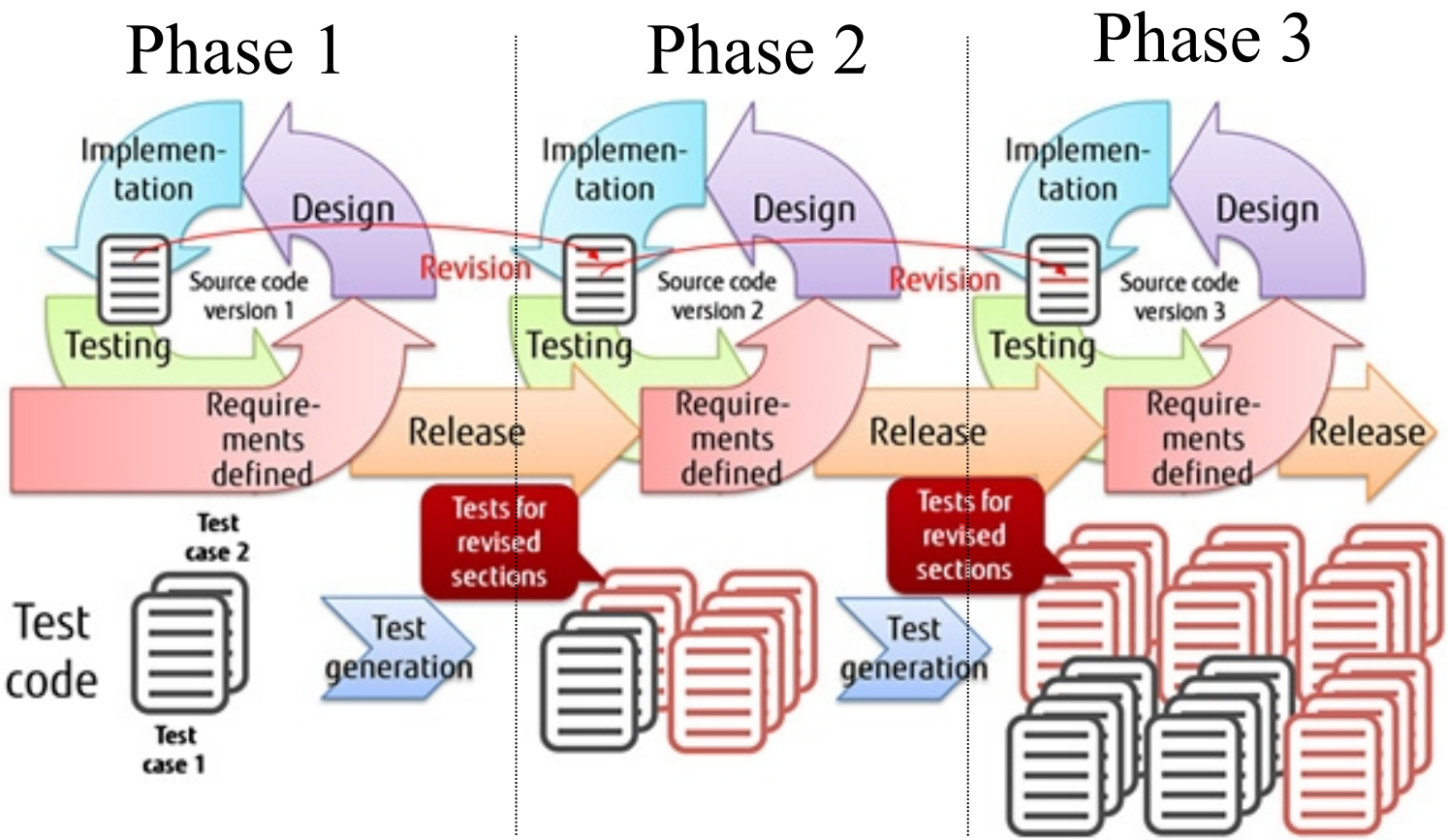
- Requirements phase
- Analysis phase
- Design phase
- Construction phase

...

Iterative and Incremental

Let's not try to build the whole product at once.
Build a useful part and evaluate it, then repeat.

Activities ≠ Phases



Iterative and Incremental

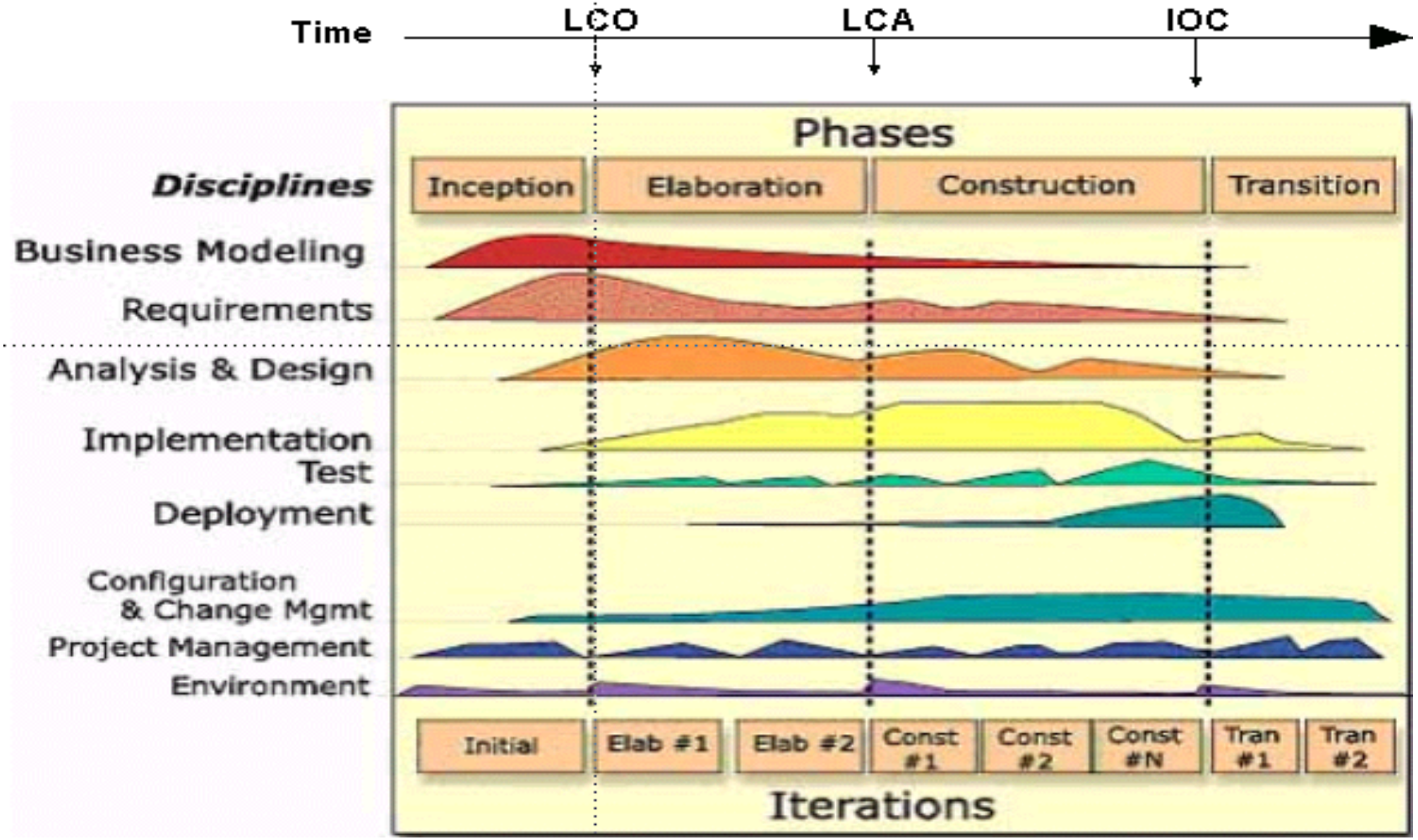
Incremental - product divided into increments.
Each increment adds **new features** and produces a **usable product**.

Iterative - iterate over the (almost) same activities for each phase or increment.

Unified Software Dev't Process (U.P.)

Workflows (disciplines) for different kinds of activities.

Phases: major divisions of project. Each has iterations.



UP is an Iterative Process

The diagram *conveys a lot* about the UP…

- workflows (disciplines) are done in parallel
- "phases" for major evolutions of the project
- iterations within each phase, as needed

In U.P. what are "activities"?

The definition of software process refers to "activities".

What word does U.P. use for these?

Characteristics of UP

- Time-boxed iterations
- Plan based, but adapts to change
- "Architecture centric"
- Identify & address risks early
- Implement requirements based on business value, architecture, or risk
  - handle risky requirements early
  - choose requirements that have big impact on the architecture
- UP is a "framework" for a process -- tailor to your project

UP is covered in *Software Spec and Design* course.

Agile

Agile is not a software process

Agile & Scrum is a separate topic

What About Individual Process?

Many software processes.

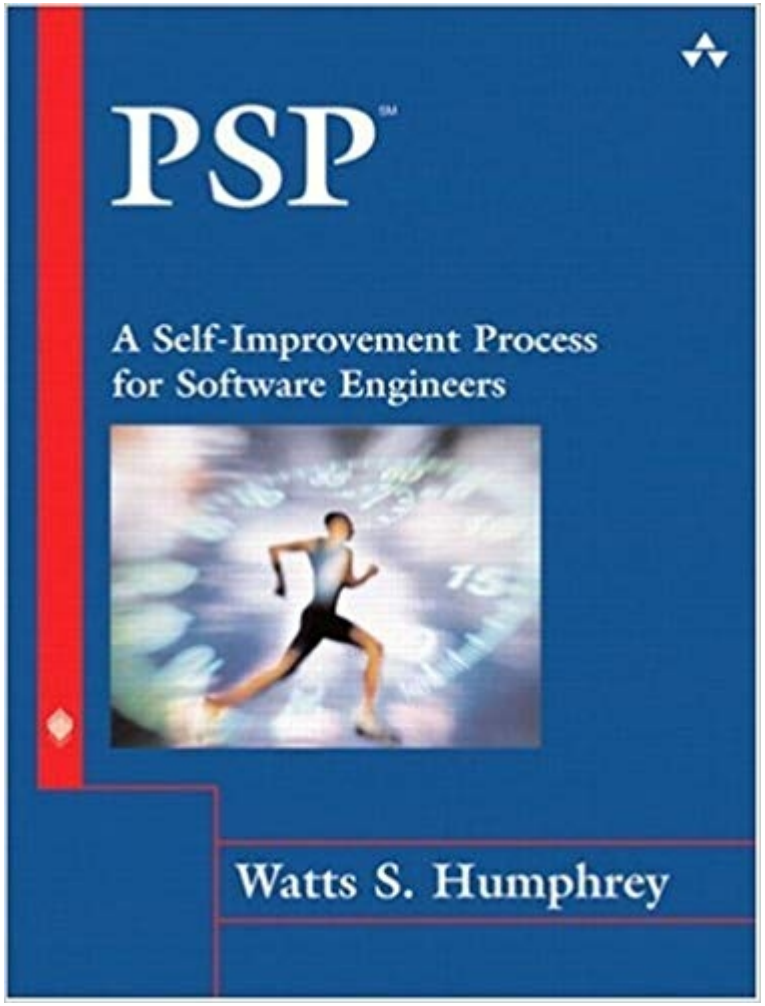
So what?

... This is a course about individual process.

The Role of Individual Process

See "Task-model" slide (1 slide).

Original Syllabus: Personal Software Process



Step-by-step course to build a personal process for:

planning

defect tracking

estimation

measuring quality & efficiency

evaluation

process improvement

Personal Software Process (PSP)

Objective: provide a disciplined process for SEs to manage their own work

- improve estimation and planning skills
- reduce defects in their products
- manage their own schedule & work quality
- improve their own software process

PSP progress through levels

PSP0: [baseline] measure time you spend on planning, design, coding, test, and *post mortem* (retrospective)

PSP0.1: measure output LOC. Add a coding standard and process improvement proposal (PIP).

PSP 1.0: Estimate program size using level 0 data. Make a test plan.

PSP 1.1: Add planning. Estimate time from program size.

PSP 2.0: Add design & code review. Emphasis on defect removal and prevention.

PSP 2.1: Add design specification.

PSP 3: Apply an iterative process to PSP2.1.

PSP Tools and Support

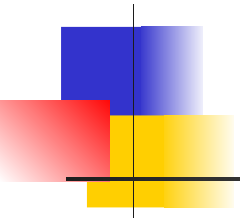
PSP emphasizes use of scripts, forms, and checklists to guide the user. These are included in course.

A useful tool is Process Dashboard (Sourceforge).

- performs time tracking. Automates some reporting.
- includes the PSP scripts and forms, and generates reports
- *can be used for other processes*!

Problem of Teaching Software Process

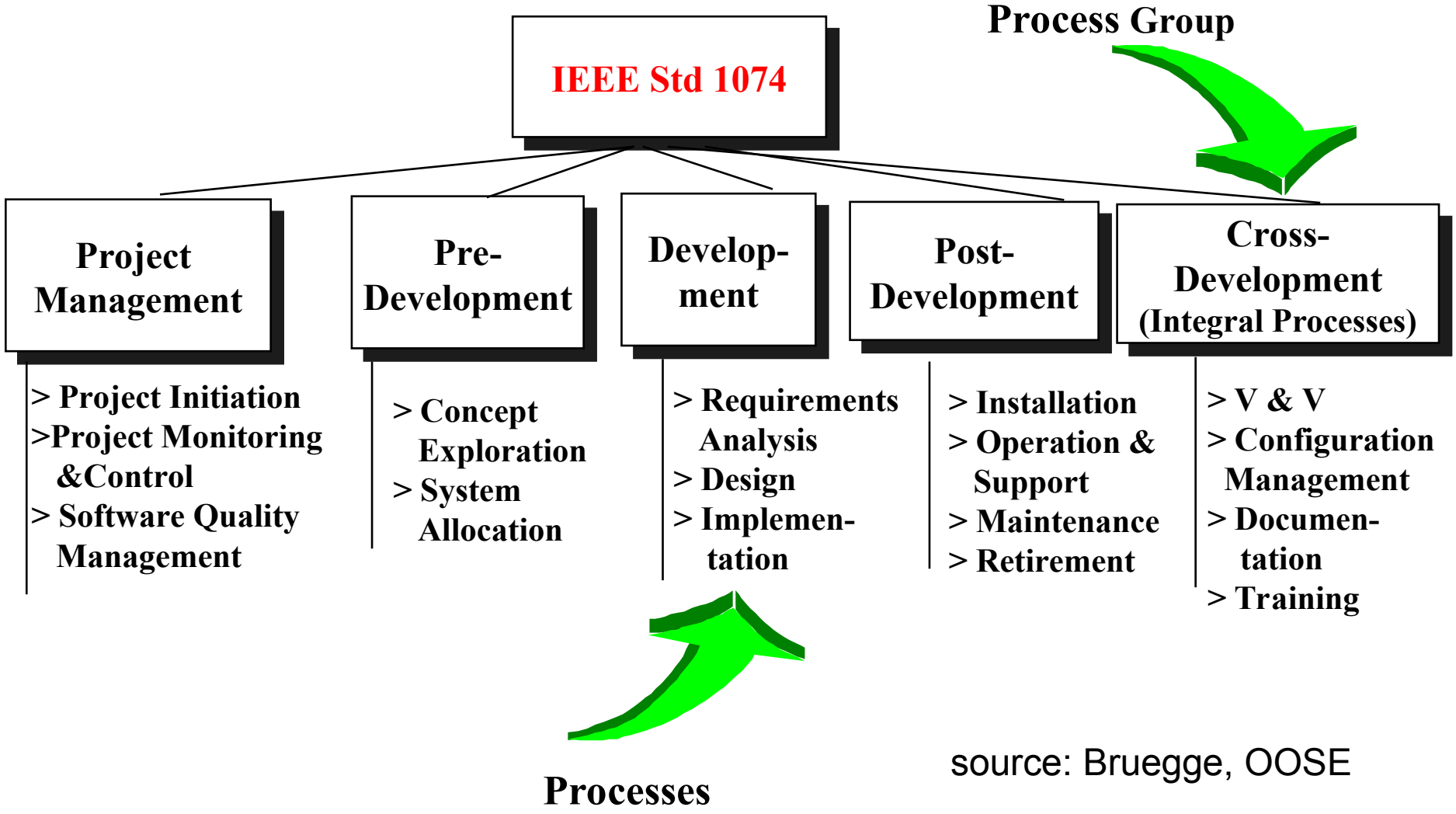
1. We learn on *small, one-semester* projects.
2. Projects often succeed based on heroic effort or super-programmers.
3. Programs aren't deployed or supported.
4. We are still learning, so process seems awkward.
5. We have many courses -- different environment from full-time developers
6. Outcome is a grade, not a paycheck or bonus



21934x
Software Process & ...

A preview of some boring stuff
from a later course.

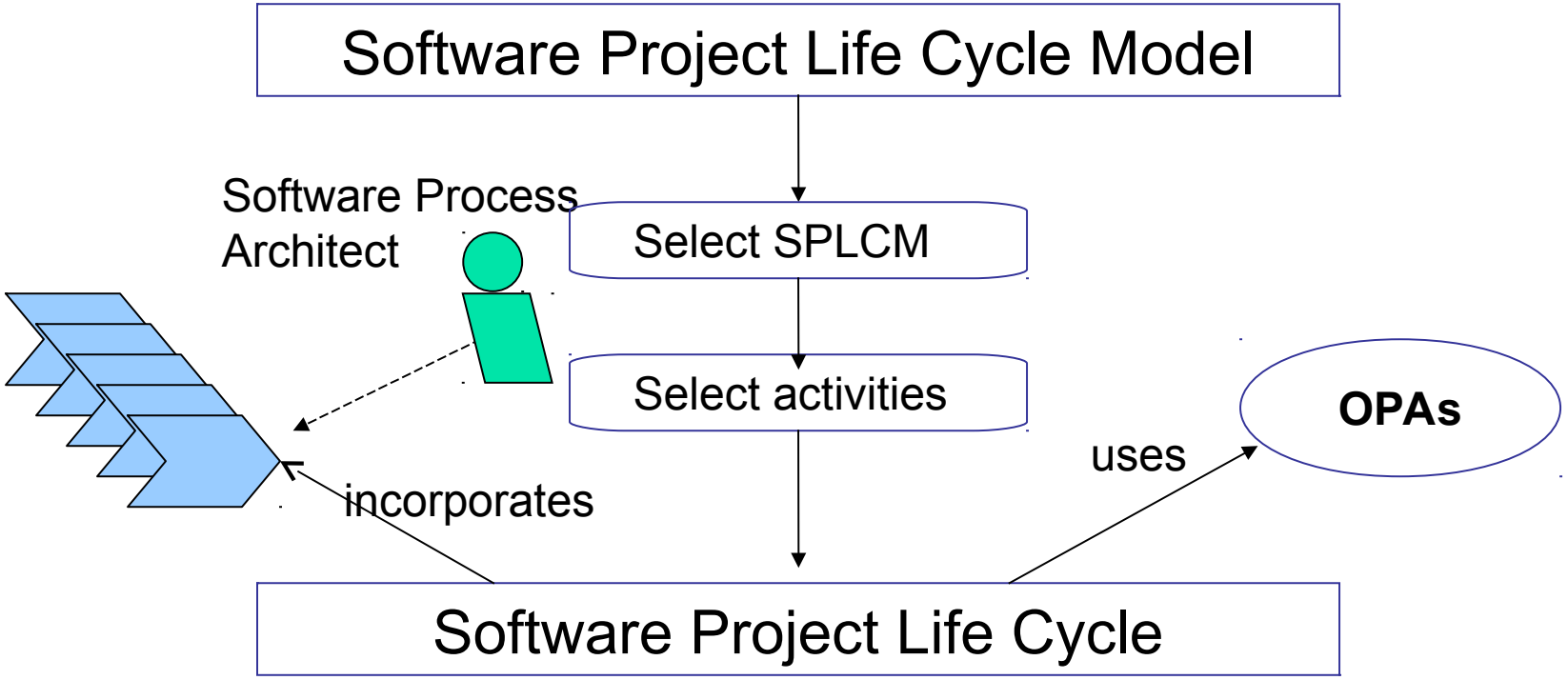
IEEE Std 1074: Standard for Software Lifecycle



source: Bruegge, OOSE

IEEE 1074

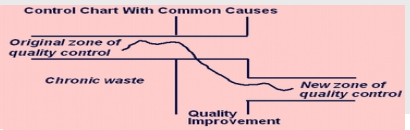
- IEEE Standard for Developing a Software Project Life Cycle Process



Overview of CMMI - Maturity Levels

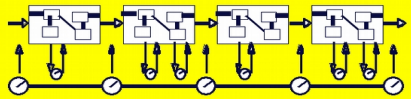
5 Focus on process improvement;

"Optimizing"



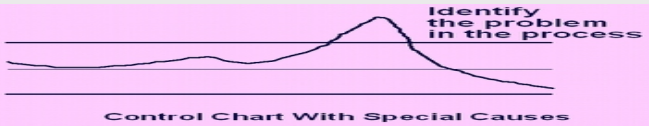
4 Process measurements; adapt to problems to reduce variance; predictable performance

"Quantitatively Managed"



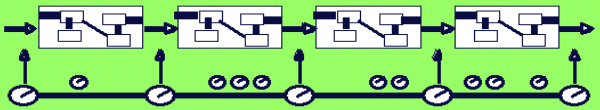
3 Process and procedures are standard, managed, improved over time

"Defined"



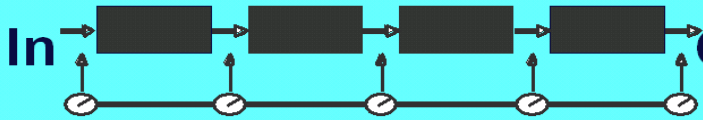
2 has process model but varies project to project; repeatable

"Managed"



1 ad hoc not repeatable

"Performed"



0 Outcome depends on "heroes"

"Incomplete