Grand Valley State University
www.gvsu.edu

Software Engineering Process Models (1)

Erik Fredericks // frederer@gvsu.edu

Ethical discussion!

https://www.gamedeveloper.com/business/unity-to-start-charging-fee-pegged-to-game-installs

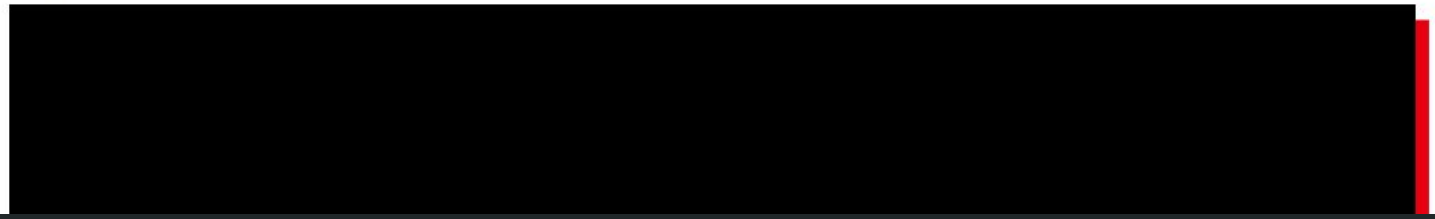
Business | 8 MIN READ | NEWS

Unity introducing new fee attached to game installs

Unity's new "Runtime Fee" will kick in after developers cross specific revenue and install thresholds.

Bryant Francis
Senior Editor

September 12, 2023



The Runtime Fee will kick in after developers cross specific revenue and install thresholds that scale with different subscription plans. For those on Unity Personal or Unity Plus licenses, the fee will kick in after a project crosses both $200,000 in revenue over 12 months and 200,000 total installs.

Developers operating on the Unity Pro or Unity Enterprise licenses will be granted a higher threshold before they begin owing fees. For those devs, the charge will kick in after a title earns $1 million over that same 12-month period and passes 1 million installs.

According to a breakdown sent to Game Developer by Unity, fees will be charged on a monthly basis. The amount charged per install will also vary depending on the license.

Unity Personal and Plus developers will pay a flat fee of 20 cents per install. Pro and Enterprise users will pay a smaller per-install fee that scales downward with the number of installations over the initial threshold. Pro user fees will begin at 15 cents per install and scale downward to as low as .02 cents per install, while Enterprise user fees will begin at 12.5 cents per install and scale down to as low as .01 cents per install.

Fees will also vary depending on what region a game is monetizing in. Unity has set an "emerging market monthly rate" for developers monetizing in countries like India. In those regions, Personal and Plus users would owe .02 cents per install, Pro users would owe .01 cents per install, and Enterprise users would owe .005 cents per install.

Developers of free-to-play games (which have a much higher install rate than premium titles) will have the option to offset this fee by adopting other Unity services, such as its LevelPlay advertising mediation service.

Team updates

Unaffiliated people from HW1 submissions have been assigned.

Friday I'll give your teams ~15 minutes to discuss directions for the semester

Start thinking about:

1) Your project proposal and what you'll be putting in it for HW2 that is due sooner than you think. ಠ_ಠ

Where Does Software Process Fit In?

We have gone over:

- Software Phases:
  - Definition
  - Development
  - Verification
  - Maintenance
  - Umbrella Activities
- Software Quality
- Software Stakeholders
- General Problem Solving:
  - Analysis
  - Synthesis

How do we put it all together? **Software Processes**.

Contents

2.1 The Meaning of Process

- A **process**: a **series of steps** involving activities, constraints, and resources that produce an intended output of some kind

What is your typical process for developing software?

2.1 The Meaning of Process
Process Characteristics

- Prescribes all major process activities
- Uses resources, subject to set of constraints (such as schedule)
- Produces intermediate and final products
- May be composed of subprocesses with hierarchy or links
- Each process activity has entry and exit criteria
- Activities are organized in sequence, so timing is clear
- Each process guiding principles, including goals of each activity
- Constraints may apply to an activity, resource or product

REMIND ME

...what is a **constraint** again?

And for funsies (req example)

Type	Example
Business	• reduce incorrectly processed orders by 50% by the end of next quarter • increase repeat orders from customer by 10% within six months after deployment
User/Stakeholder	• add new customer account • view order history • check order status • create new order
Functional/Solution	• display customer last name as a link to account history • allow sorting by account opening date
Non-Functional	• allow up to 200 concurrent users • require strong passwords of at least 8 characters in length containing a minimum of one non-alphabet character
Implementation/Transition	• must run on all Java platforms including 64-bit versions • users must pass an online certification before being allowed to use the system

2.1 The Meaning of Process
The Importance of Processes

- Impose **consistency** and **structure** on a set of activities
- Guide us to understand, control, examine, and improve the activities
- Enable us to capture our experiences and pass them along

2.2 Software Process Models
Reasons for Modeling a Process

- To form a common understanding
- To find inconsistencies, redundancies, omissions
- To find and evaluate appropriate activities for reaching process goals
- To tailor a general process for a particular situation in which it will be used

2.2 Software Process Models
Software Life Cycle

- When a process involves building a software, the process may be referred to as software life cycle
  - Requirements analysis and definition
  - System (architecture) design
  - Program (detailed/procedural) design
  - Writing programs (coding/implementation)
  - Testing: unit, integration, system
  - System delivery (deployment)

  - Maintenance

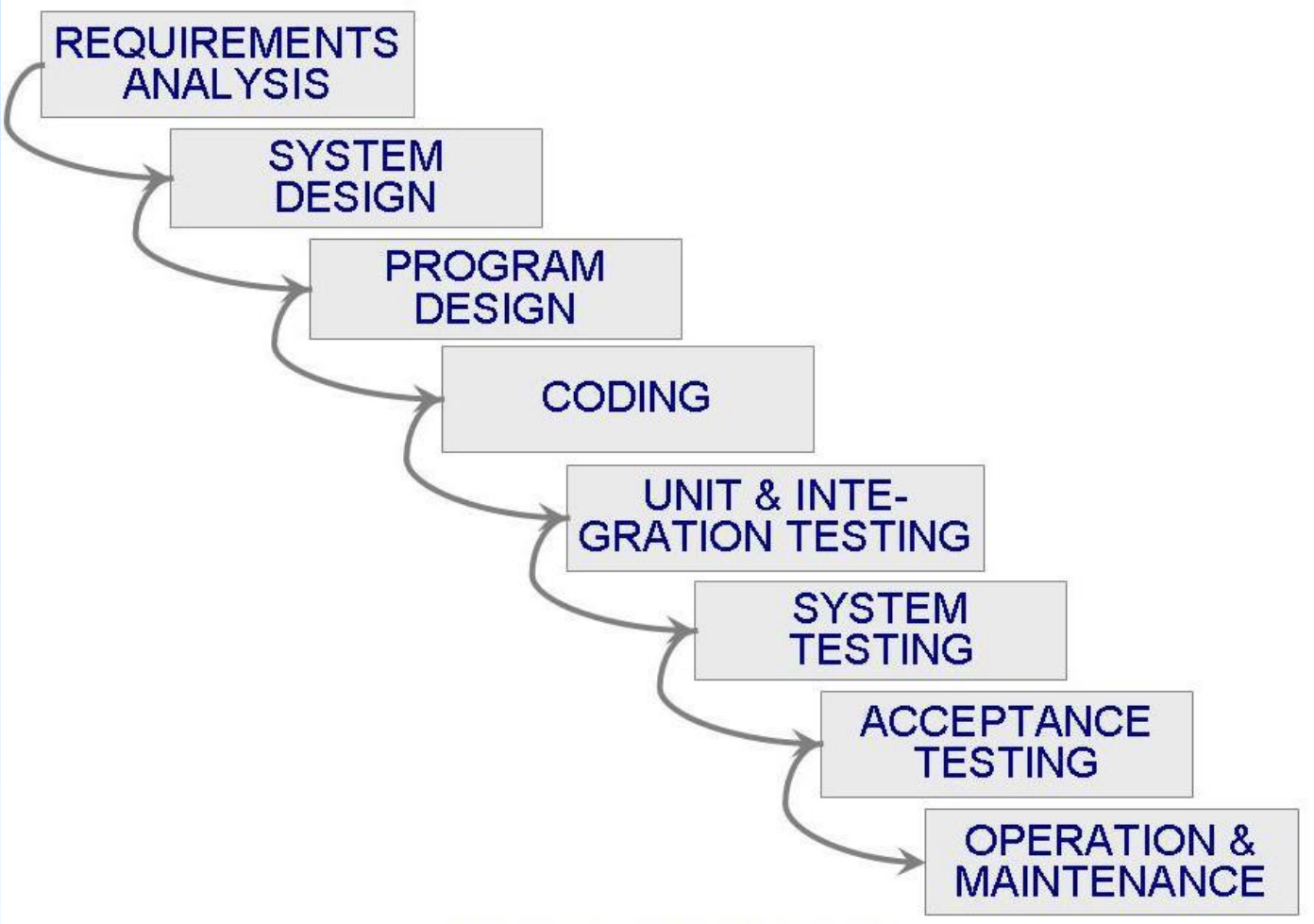
2.2 Software Process Models
Software Development Process Models

- Waterfall model
- V model
- Prototyping model
- Phased development: increments and iteration
- Spiral model
- Agile methods

2.2 Software Process Models
Waterfall Model

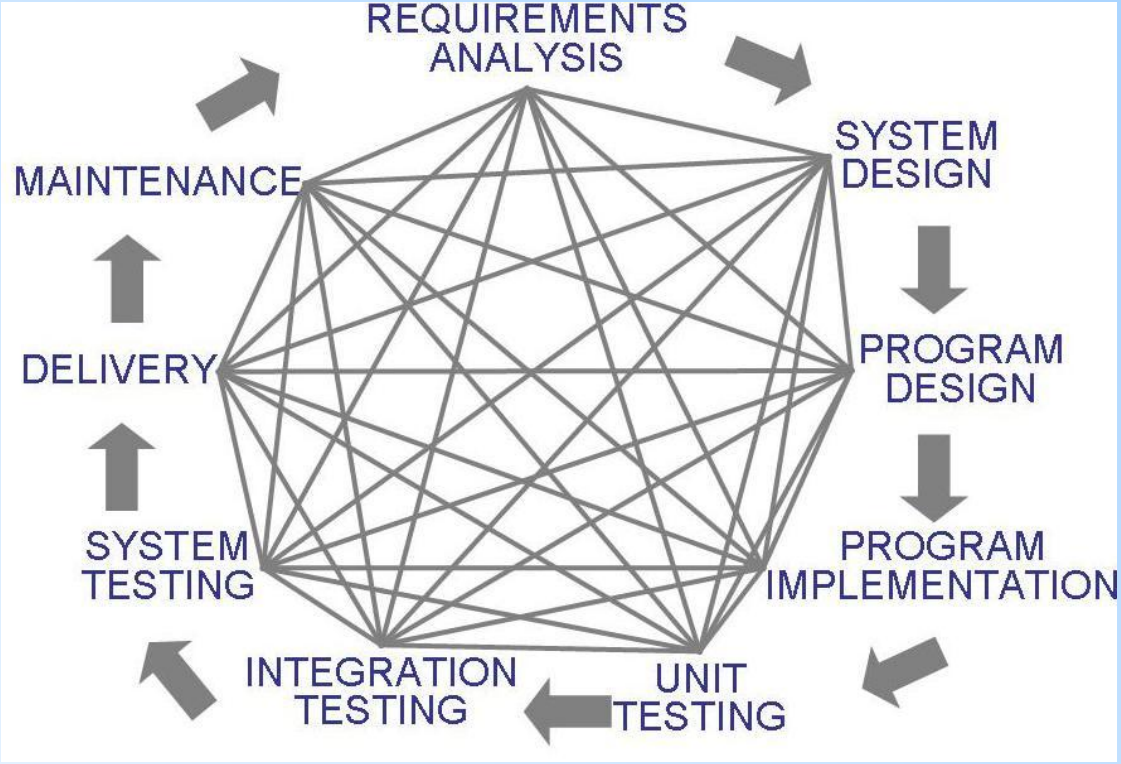
- One of the first process development models proposed
- Works for well understood problems with **minimal or no changes in the requirements**
- Simple and easy to explain to customers
- It presents
  - a very high-level view of the development process
  - sequence of process activities
- Each major phase is marked by milestones and deliverables (artifacts)

2.2 Software Process Models
Waterfall Model (continued)



2.2 Software Process Models
Waterfall Model (continued)

- There is no iteration in waterfall model
- Most software developments apply a great many iterations



2.2 Software Process Models
Sidebar 2.1 Drawbacks of The Waterfall Model

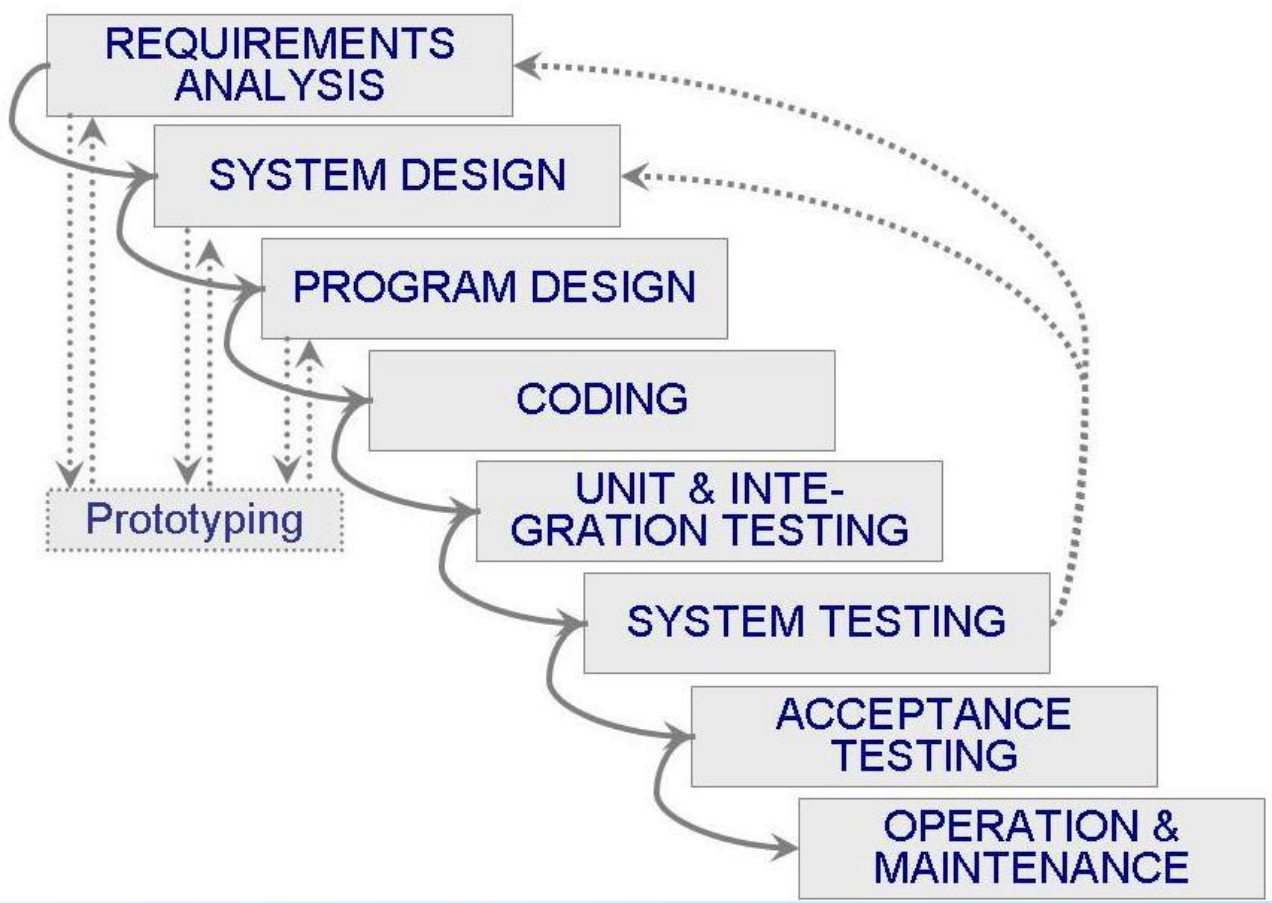
- Provides no guidance how to handle changes to products and activities during development (assumes requirements can be frozen)
- Views software development as manufacturing process rather than as creative process
- There is no iterative activities that lead to creating a final product
- Long wait before a final product

2.2 Software Process Models
Waterfall Model with Prototype

- A prototype is a partially developed product
- Prototyping helps
  - developers assess alternative design strategies (design prototype)
  - users understand what the system will be like (user interface prototype)
- Prototyping is useful for verification and validation

2.2 Software Process Models
Waterfall Model with Prototype (continued)

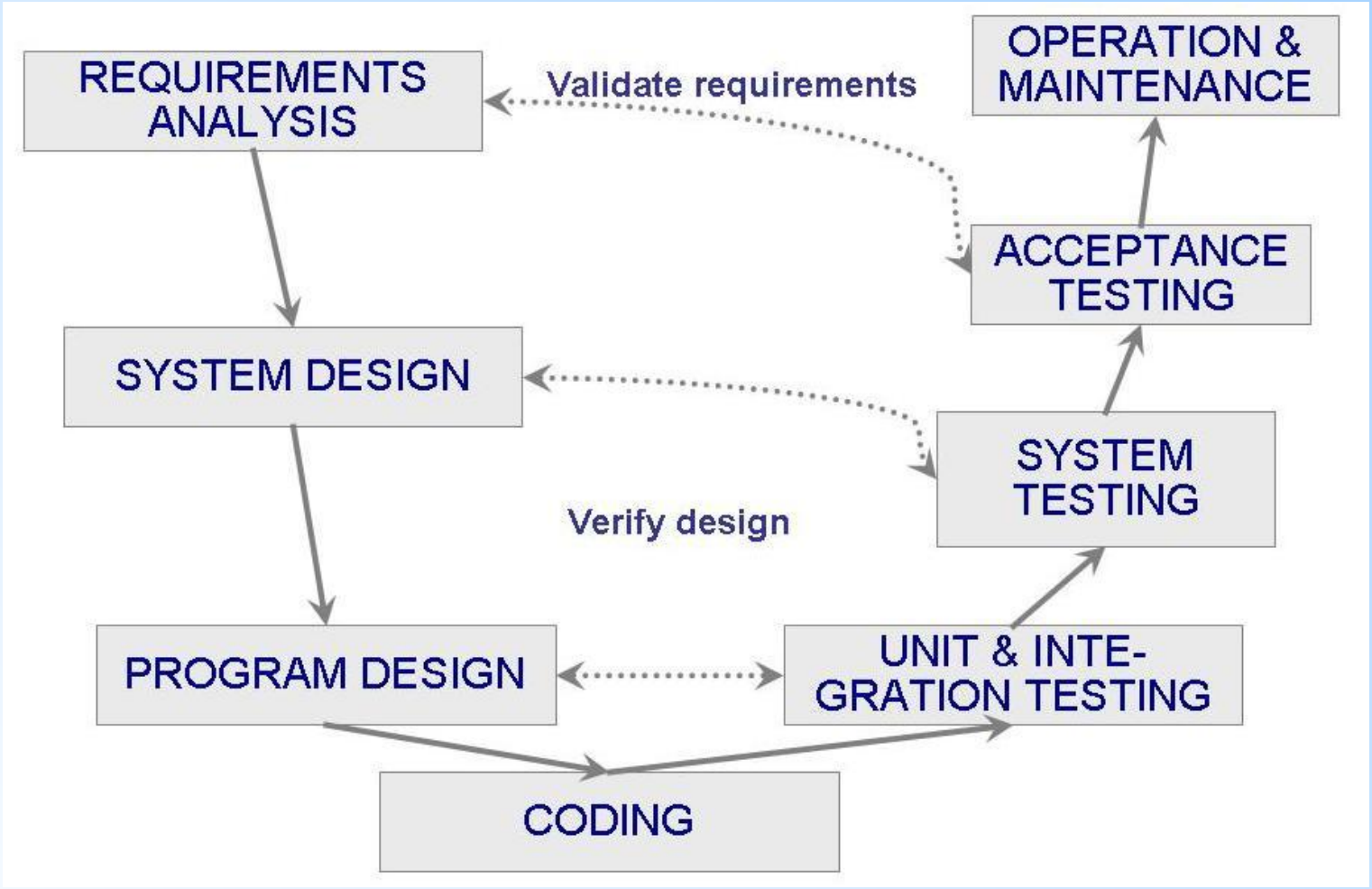
- Waterfall model with prototyping



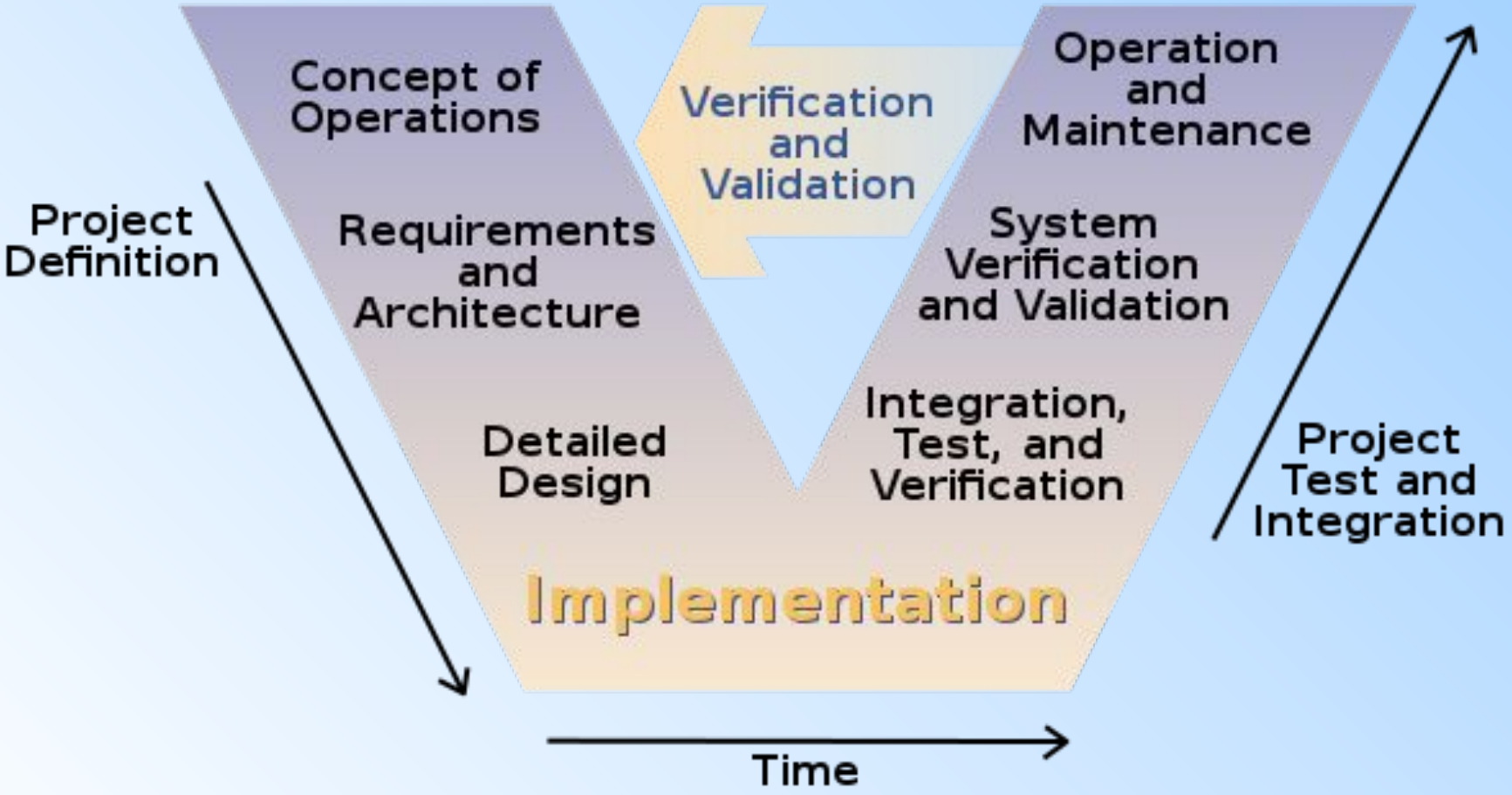
2.2 Software Process Models
V Model

- A variation of the waterfall model
- Uses unit testing to verify procedural design
- Uses integration testing to verify architectural (system) design
- Uses acceptance testing to validate the requirements
- If problems are found during verification and validation, the left side of the V can be re-executed before testing on the right side is re-enacted

2.2 Software Process Models
V Model (continued)

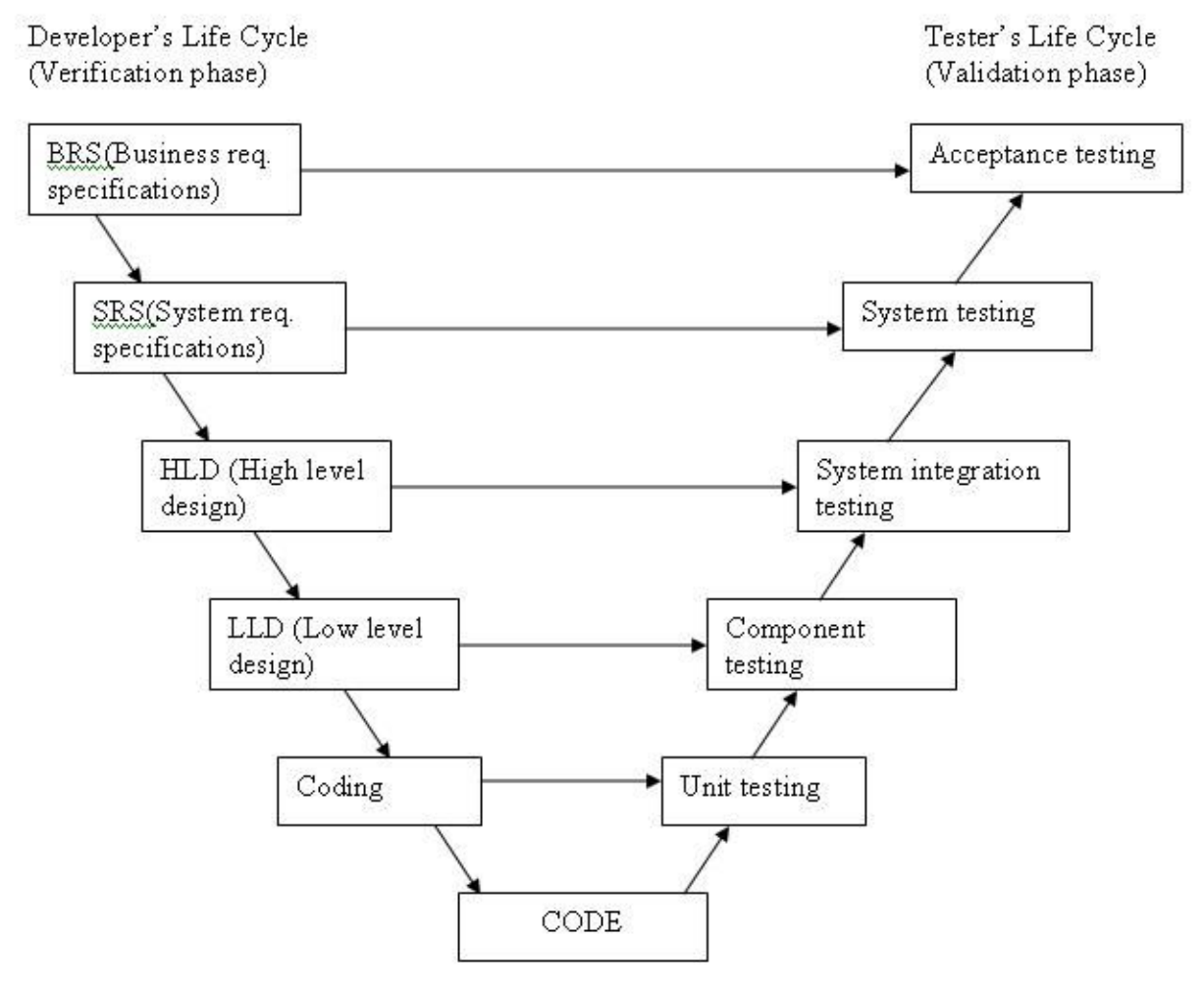


2.2 Software Process Models
V Model* (continued)



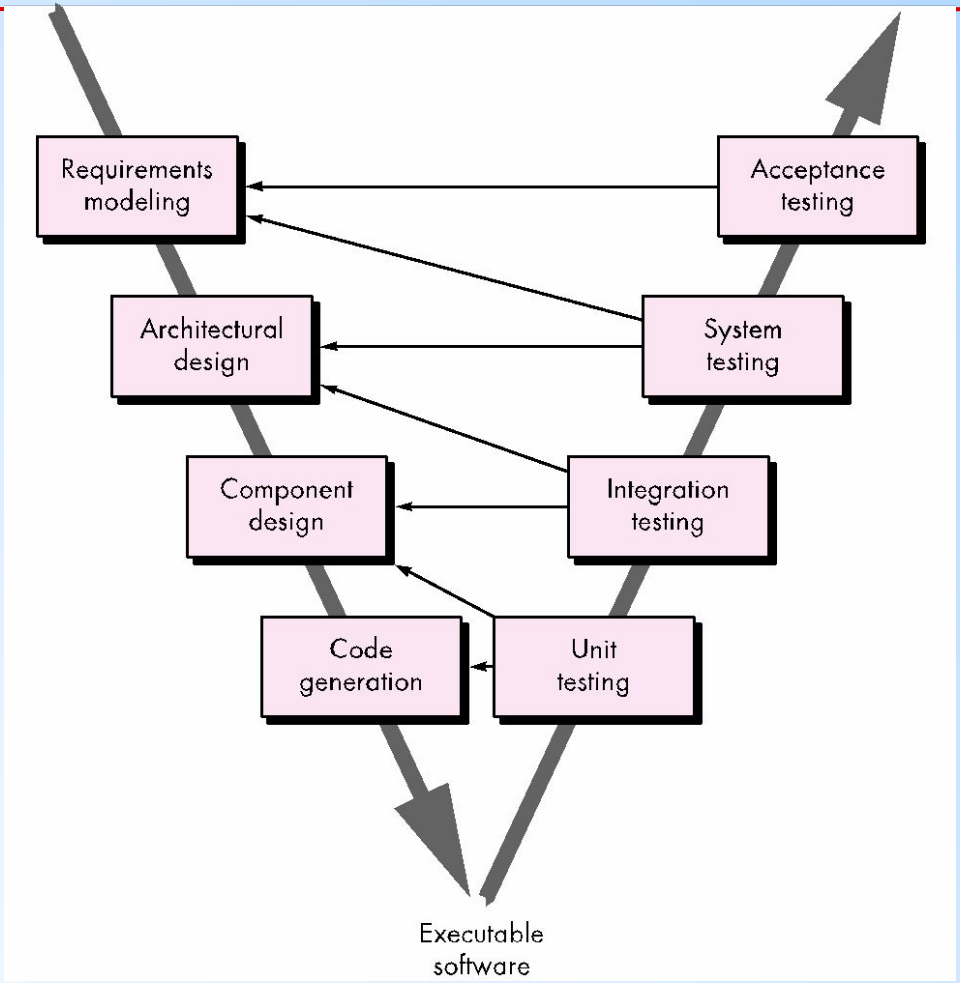
*Clarus Concept of Operations. Publication No. FHWA-JPO-05-072, Federal Highway Administration (FHWA), 2005

2.2 Software Process Models
V Model* (continued)



* http://istqbexamcertification.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/

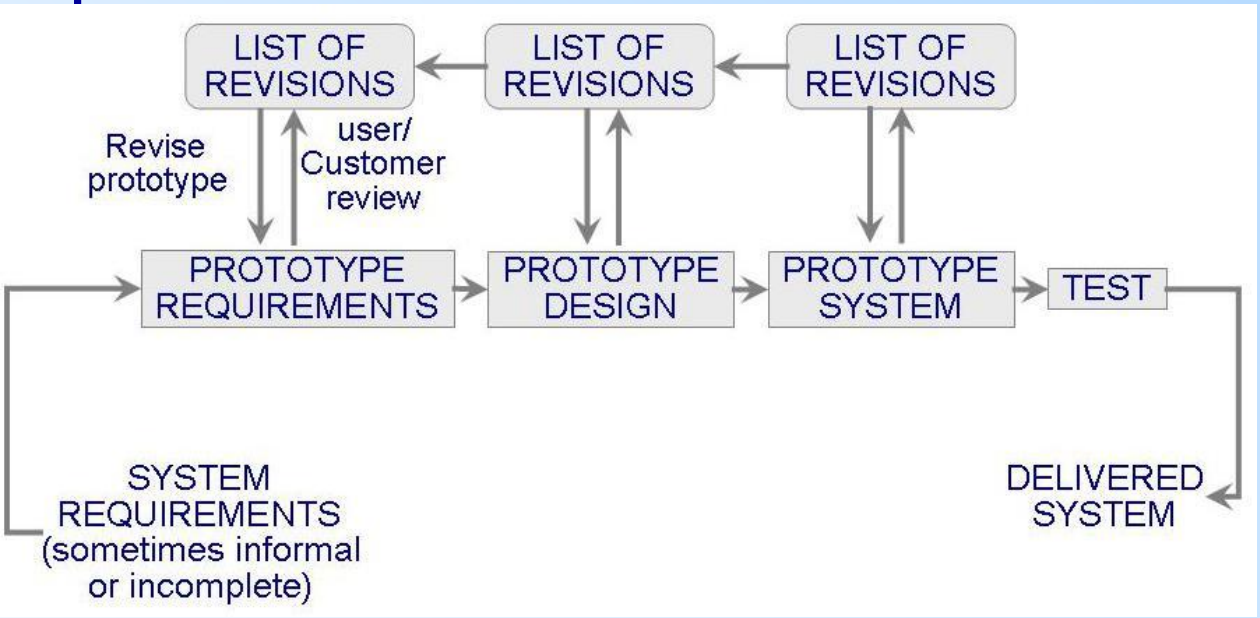
2.2 Software Process Models
V Model* (continued)



* Pressman, Roger S. *Software engineering: a practitioner's approach*

2.2 Software Process Models
Prototyping Model

- Allows repeated investigation of the requirements or design
- Reduces risk and uncertainty in the development



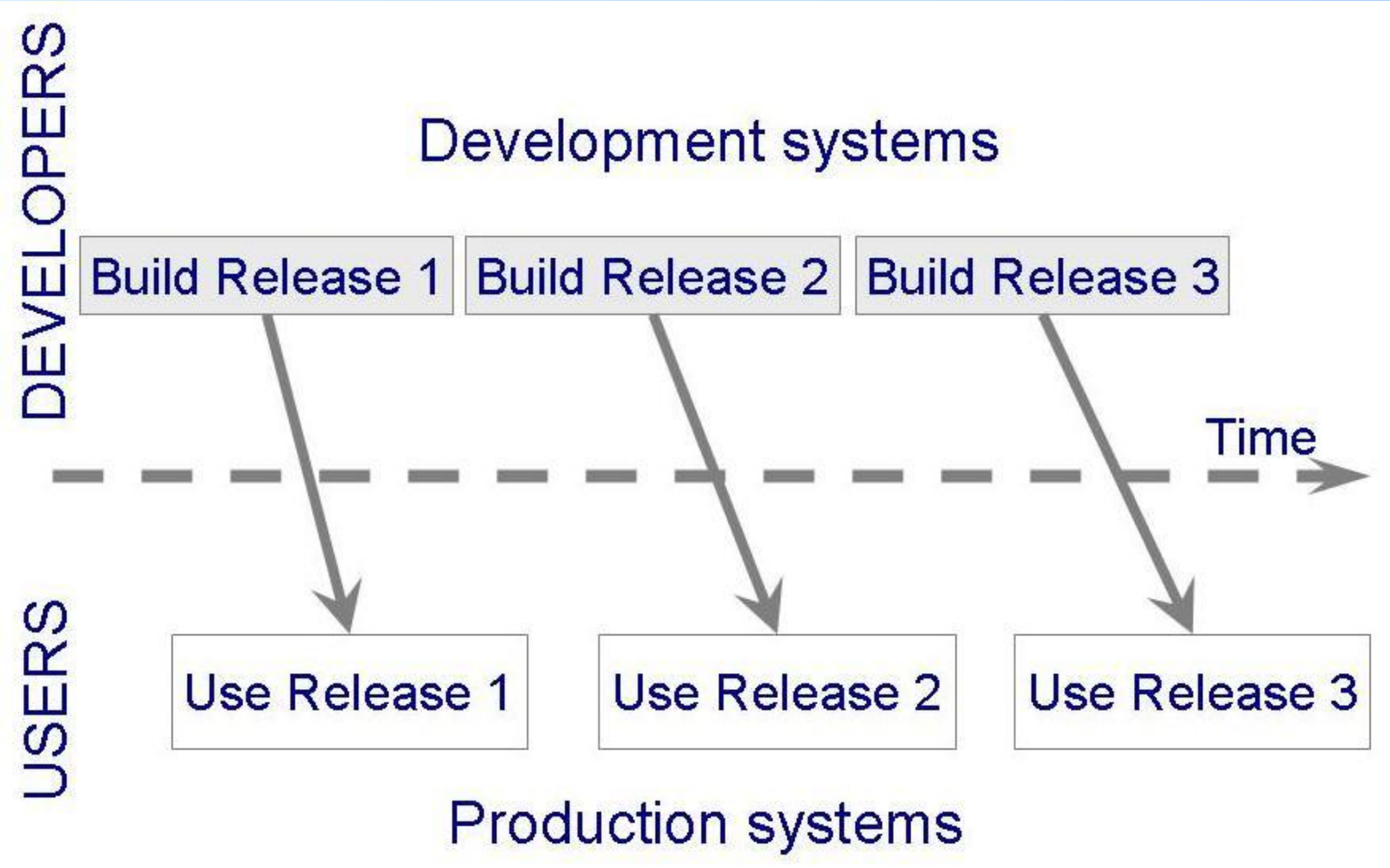
2.2 Software Process Models

Phased Development: Increments and Iterations

- Shorter cycle time
- System delivered in pieces
  - enables customers to have some functionality while the rest is being developed
- Allows two systems functioning in parallel
  - the production system (release *n*): currently being used
  - the development system (release *n+1*): the next version

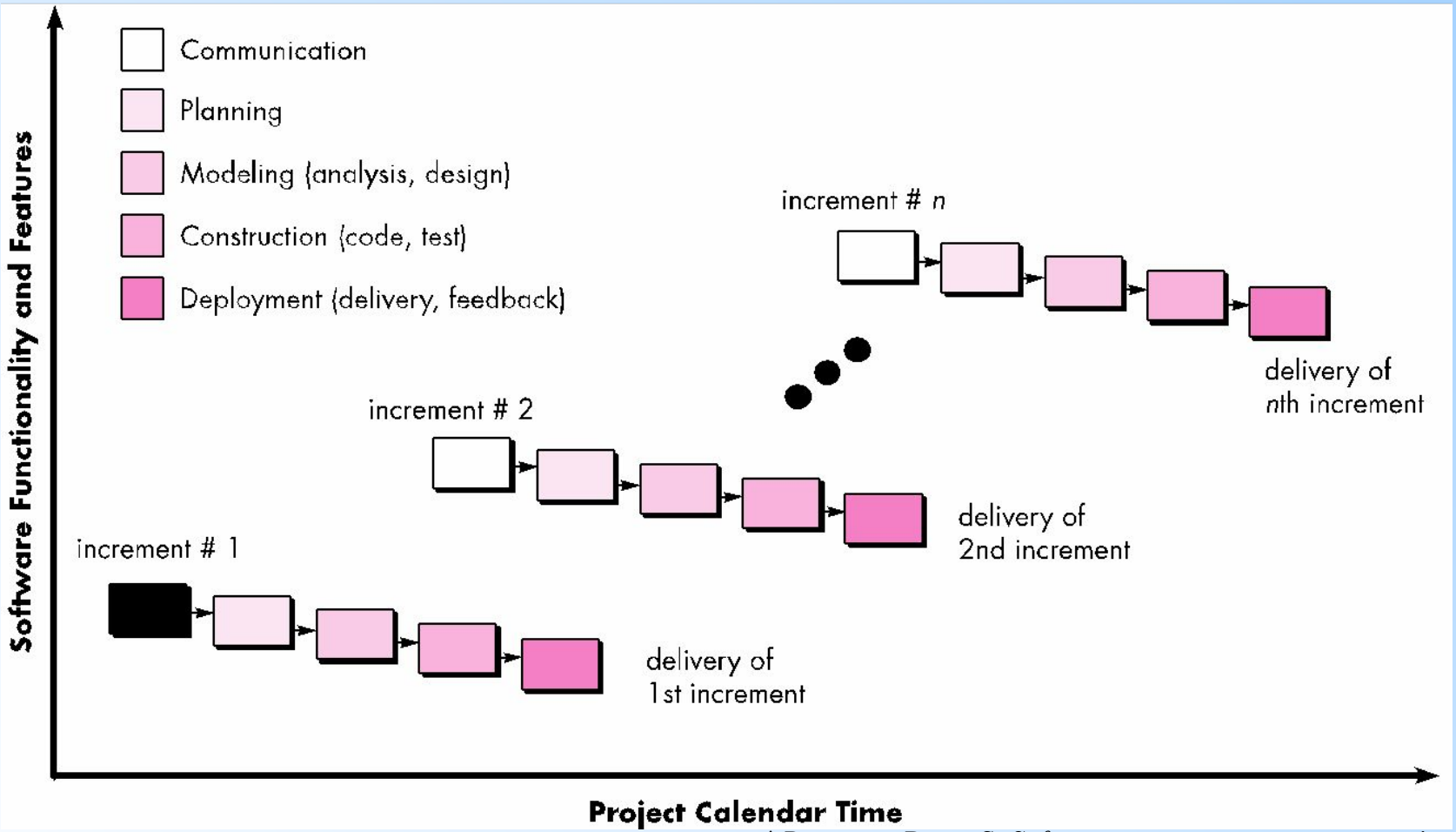
2.2 Software Process Models

Phased Development: Increments and Iterations (continued)



2.2 Software Process Models

Phased Development: Increments and Iterations* (continued)

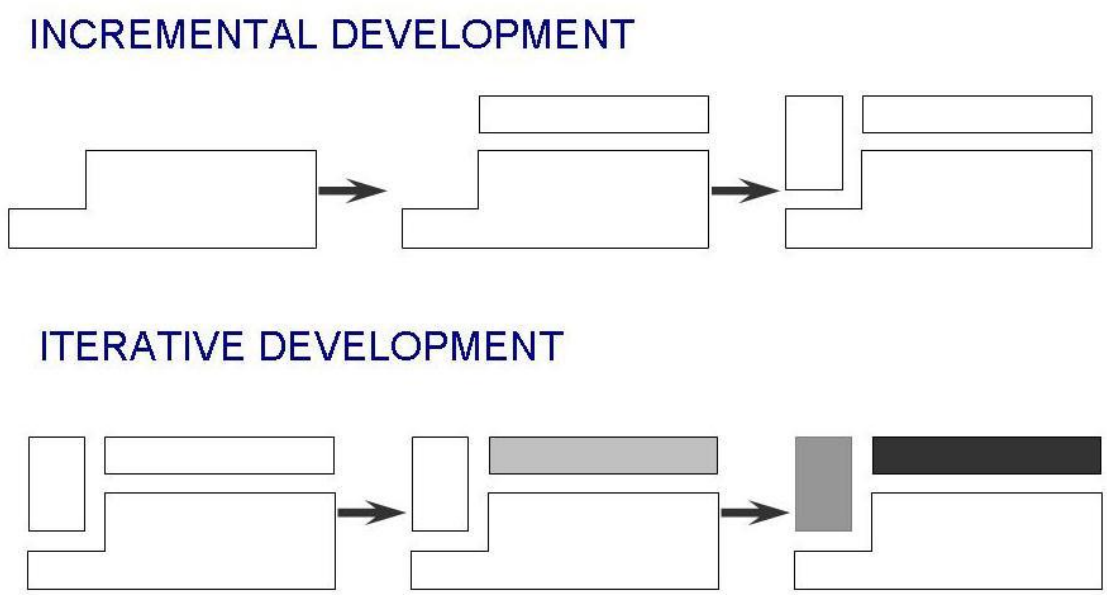


* Pressman, Roger S. *Software engineering: a practitioner's approach*

2.2 Software Process Models

Phased Development: Increments and Iterations (continued)

- **Incremental development:** starts with small functional subsystem and adds functionality with each new release
- **Iterative development**: starts with full system, then changes functionality of each subsystem with each new release



2.2 Software Process Models
Phased Development: Increments and Iterations (continued)

- Phased development is desirable for several reasons
  - Training can begin early, even though some functions are missing
  - Markets can be created early for functionality that has never before been offered
  - Frequent releases allow developers to fix unanticipated problems globally and quickly
  - The development team can focus on different areas of expertise with different releases

2.2 Software Process Models
Spiral Model

- Suggested by Boehm (1988)
- Combines development activities with risk management to minimize and control risks
- The model is presented as a spiral in which each iteration is represented by a circuit around four major activities
  - Plan
  - Determine goals, alternatives and constraints
  - Evaluate alternatives and risks
  - Develop and test

2.2 Software Process Models
Spiral Model (continued)

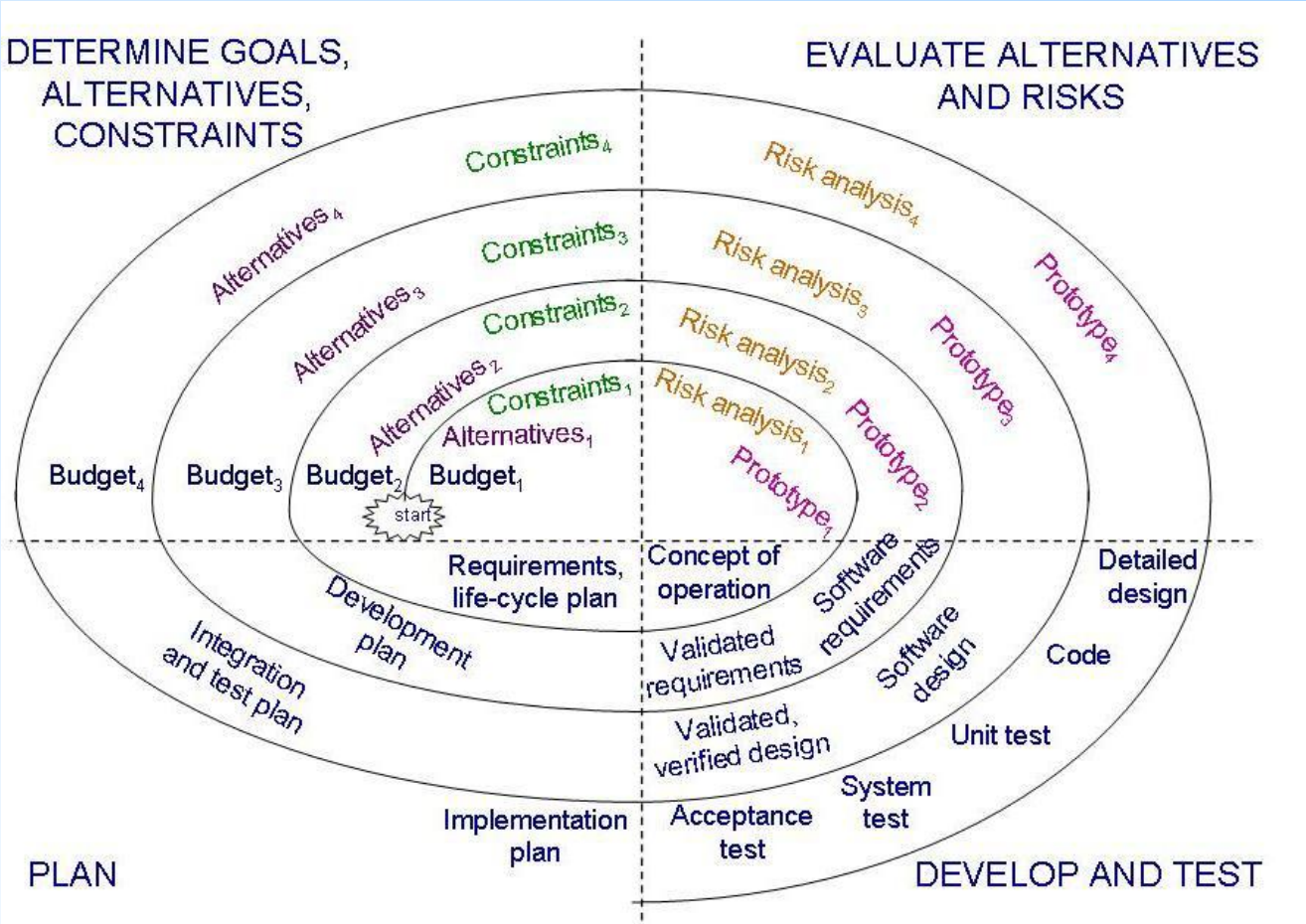
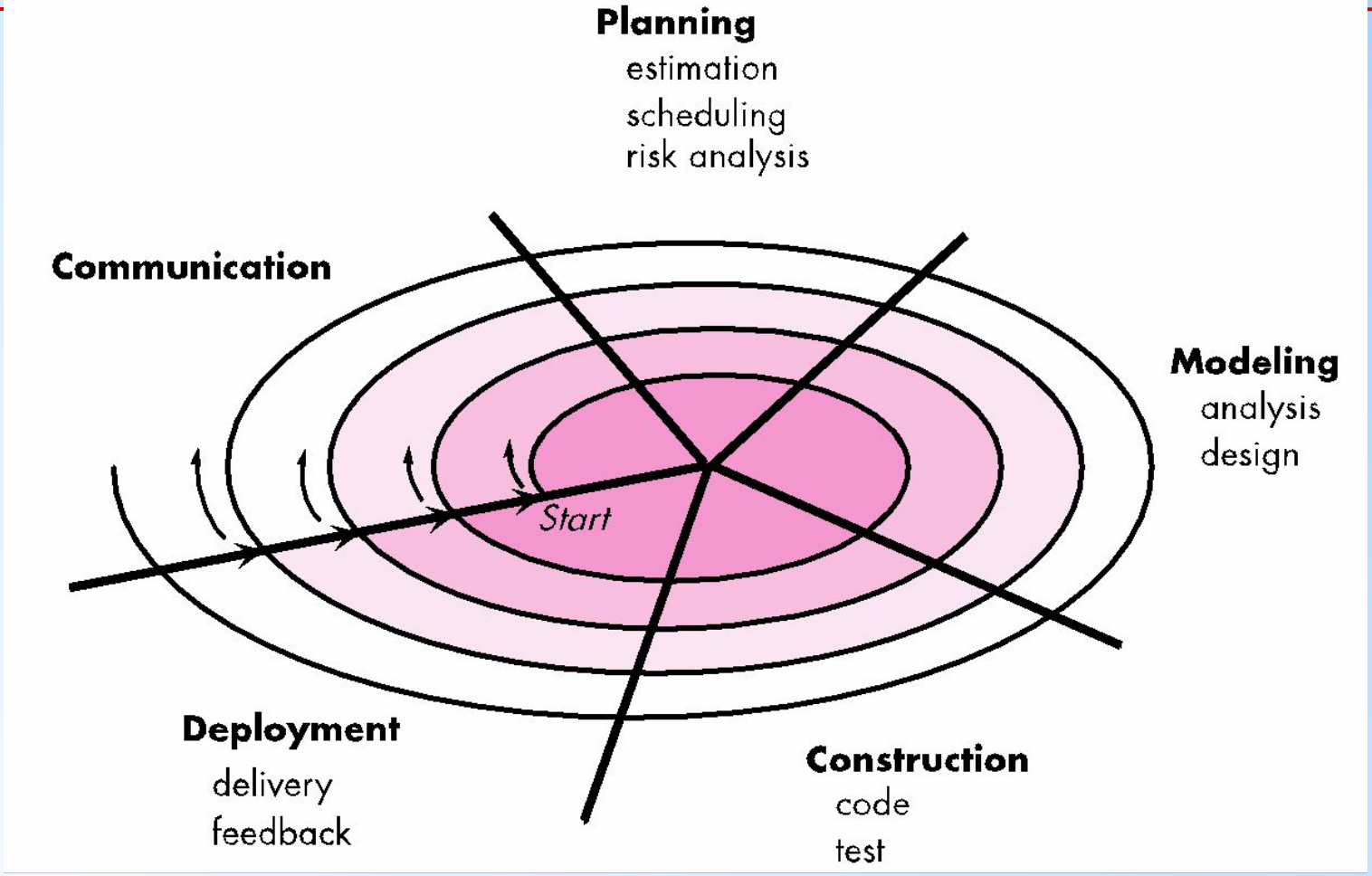


Figure 2.10 the spiral model.

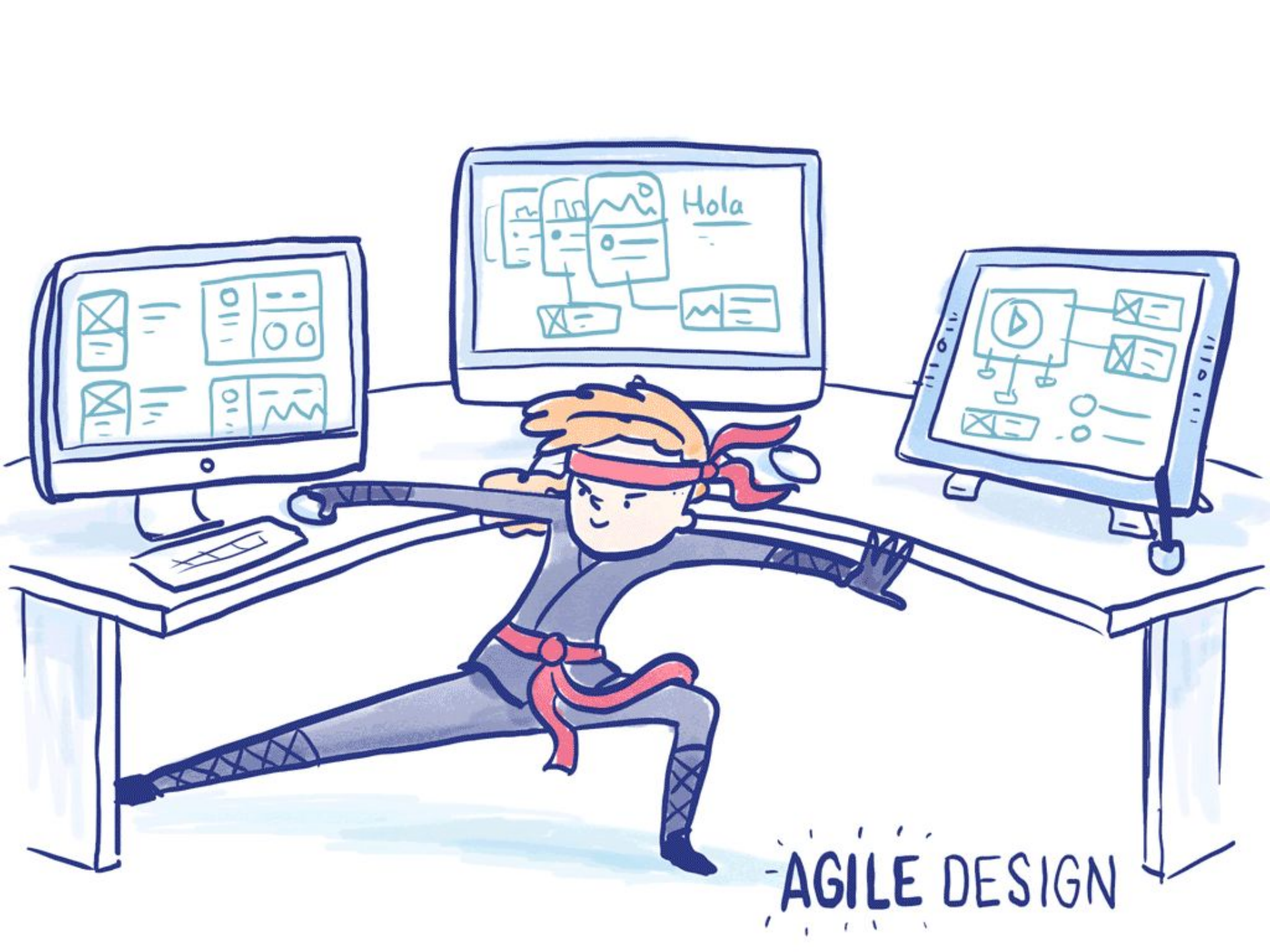
2.2 Software Process Models
Spiral Model* (continued)



* Pressman, Roger S. *Software engineering: a practitioner's approach*

2.2 Software Process Models
Agile Methods

- Emphasis on flexibility in producing software quickly and capably
- Agile manifesto
  - Value individuals and interactions over process and tools
  - Prefer to invest time in producing working software rather than in producing comprehensive documentation
  - Focus on customer collaboration rather than contract negotiation
  - Concentrate on responding to change rather than on creating a plan and then following it


Hola
AGILE DESIGN

2.2 Software Process Models
Agile Methods: Examples of Agile Process

- Extreme programming (XP)
- Crystal: a collection of approaches based on the notion that every project needs a unique set of policies and conventions
- Scrum: 30-day iterations; multiple self-organizing teams; daily “scrum” coordination
- Adaptive software development (ASD)

2.2 Software Process Models
Agile Methods: Extreme Programming

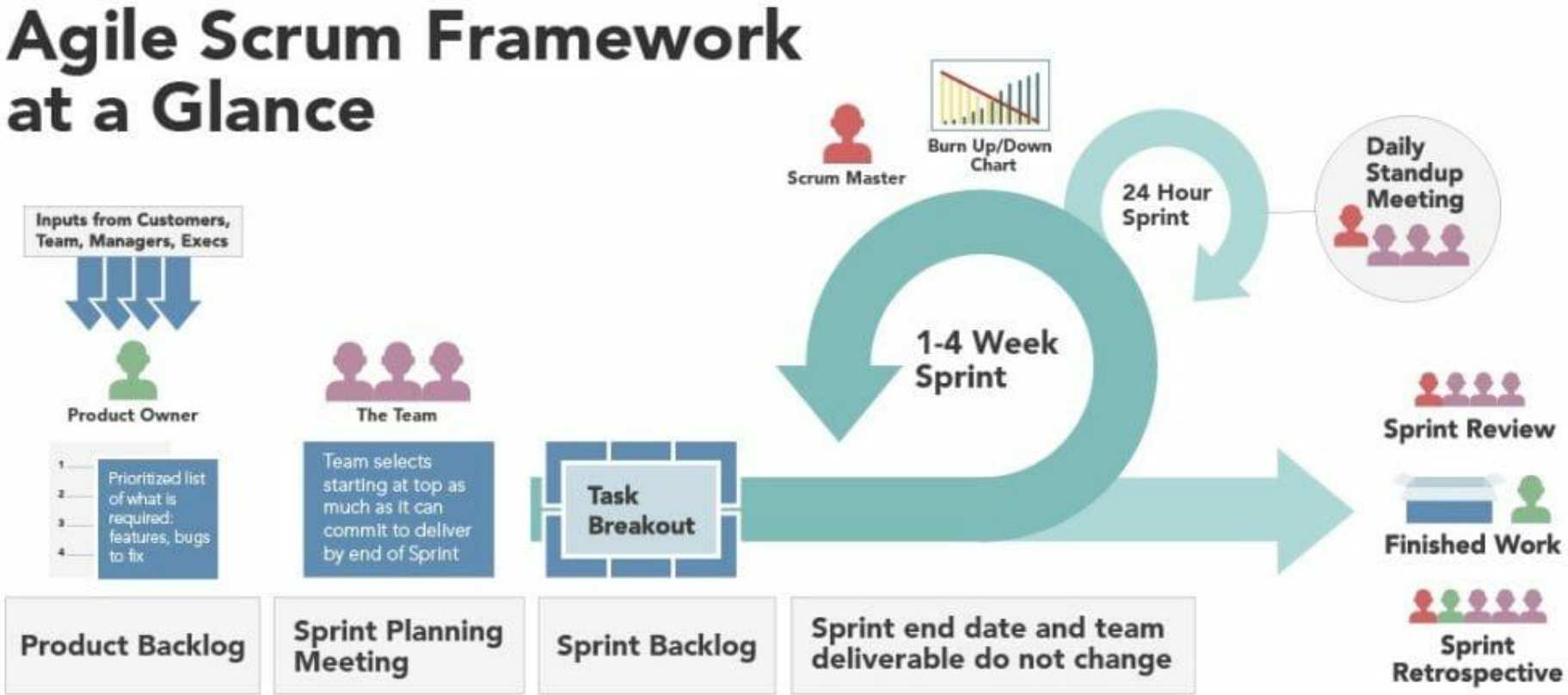
- Emphasis on four characteristics of agility
  - *Communication*: continual interchange between customers and developers
  - *Simplicity*: select the simplest design or implementation
  - *Courage*: commitment to delivering functionality early and often
  - *Feedback*: loops built into the various activities during the development process

2.2 Software Process Models
Agile Methods: Twelve Facets of XP

- The planning game *(customer defines value)*
- Small release
- Metaphor *(common vision, common names)*
- Simple design
- Writing tests first
- Refactoring
- Pair programming
- Collective ownership
- Continuous integration *(small increments)*
- Sustainable pace *(40 hours/week)*
- On-site customer
- Coding standard

2.2 Software Process Models
Sidebar 2.2 When is Extreme Too Extreme?

- Extreme programming's practices are interdependent
  - A vulnerability if one of them is modified
- Requirements expressed as a set of test cases must be passed by the software
  - System passes the tests but is not what the customer is paying for
- Refactoring issue
  - Difficult to rework a system without degrading its architecture


Agile Scrum Framework at a Glance
Scrum Master
Burn Up/Down Chart
Daily Standup Meeting
24 Hour Sprint
1-4 Week Sprint
Inputs from Customers, Team, Managers, Execs
Product Owner
The Team
Prioritized list of what is required: features, bugs to fix
Team selects starting at top as much as it can commit to deliver by end of Sprint
Task Breakout
Sprint Review
Finished Work
Sprint Retrospective
Product Backlog
Sprint Planning Meeting
Sprint Backlog
Sprint end date and team deliverable do not change

2.3 What this means for You

- Process development involves activities, resources, and product
- Process model includes organizational, functional, behavioral and other perspectives
- A process model is useful for guiding team behavior, coordination and collaboration

IF we're close to time…

(or it is 1:30)

Meet up with your team and come up with:

1) A team name
2) A tentative team project **and** a language you'll be using!