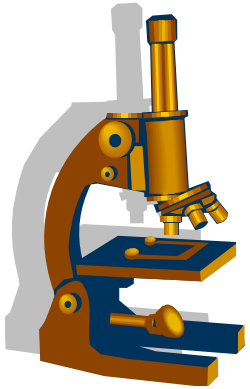
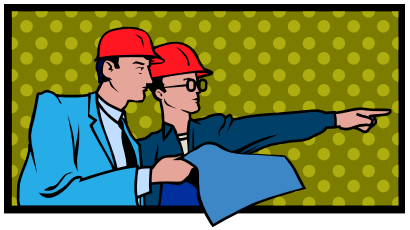
What we have done

Science/Engineering

- Seek new knowledge
- Search for basic laws and principles
- General approaches to a general problem

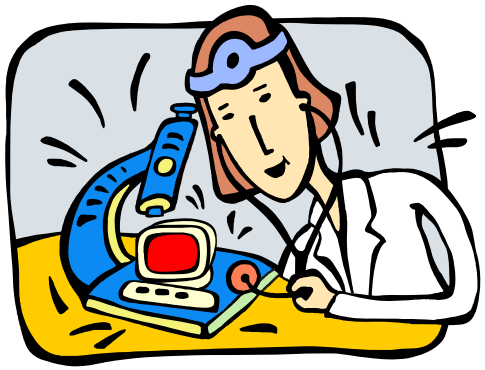


- Solve problems using knowledge
- Establish standard approaches
- Specific solution to a specific problem



Computer Science/Software Engineering

- Algorithms
- Theory of computation
- Finite state automata



- Software-intensive products
- Interaction with users, customers
- Design
- State machines

What we have done.

- We have explored how people build software.
- We have explored why people build software.
- We have explored where people are when they build software.

- Bottom line: people build software to achieve specific objectives

- A “glitch” in a state computer will not allow extended unemployment benefits until the program is modified. There is a hard limit programmed in the program on the number of weeks.

What makes software different?

- SE is a unique brand of engineering
  - Software is malleable
  - Software construction is human-intensive
  - Software is intangible
  - Software problems are unprecedentedly complex
  - Software directly depends upon the hardware
    - It is at the top of the system engineering “food chain”
  - Software solutions require unusual rigor
  - Software has discontinuous operational nature

Top Level

- Roles
- Process models/ processes
- Tools
- Models
- Practices
- Design

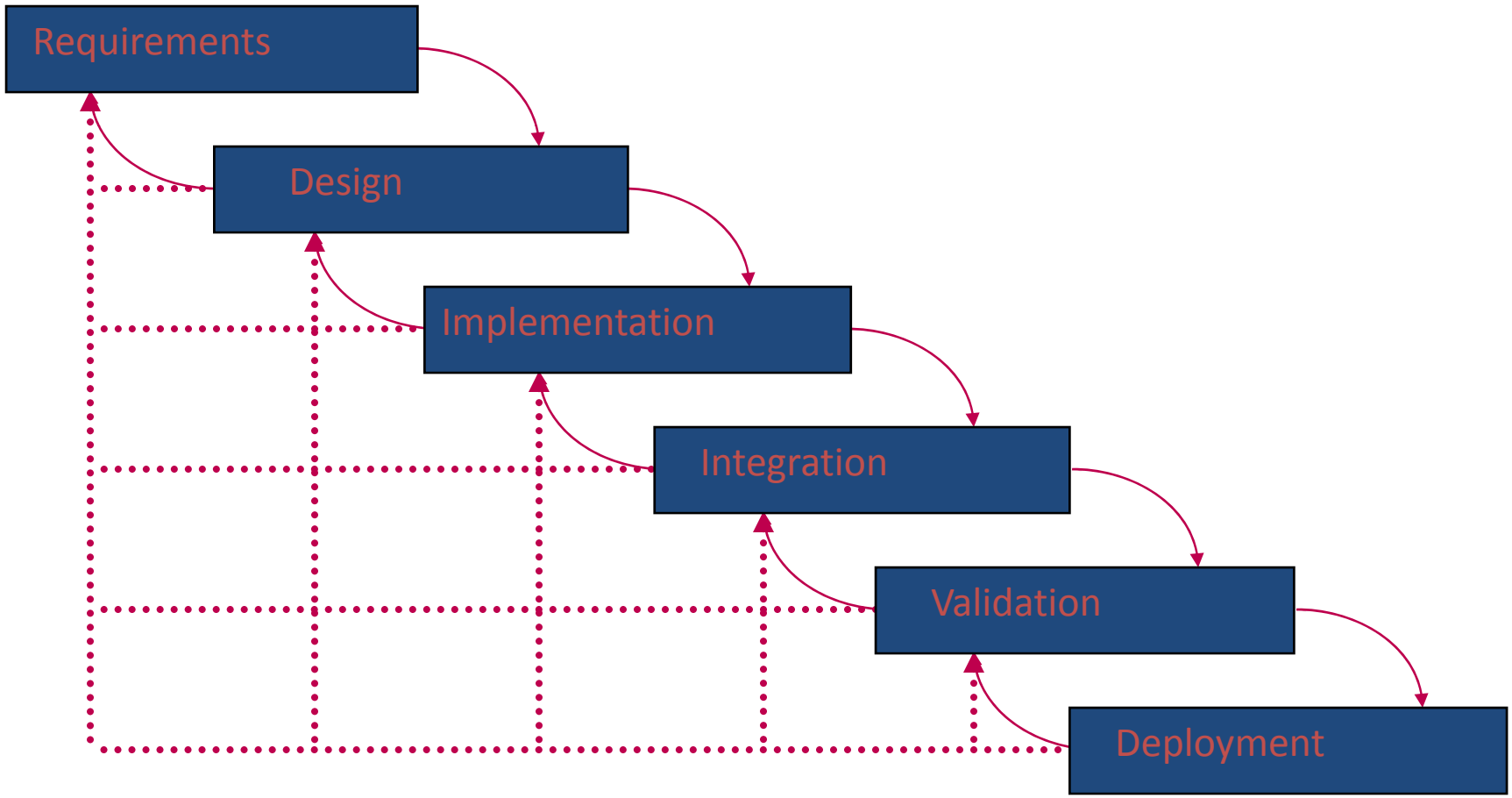
Roles

- Project manager
- Architect
- Designer
- Programmer
- Tester
- User
- Customer
- Product manager

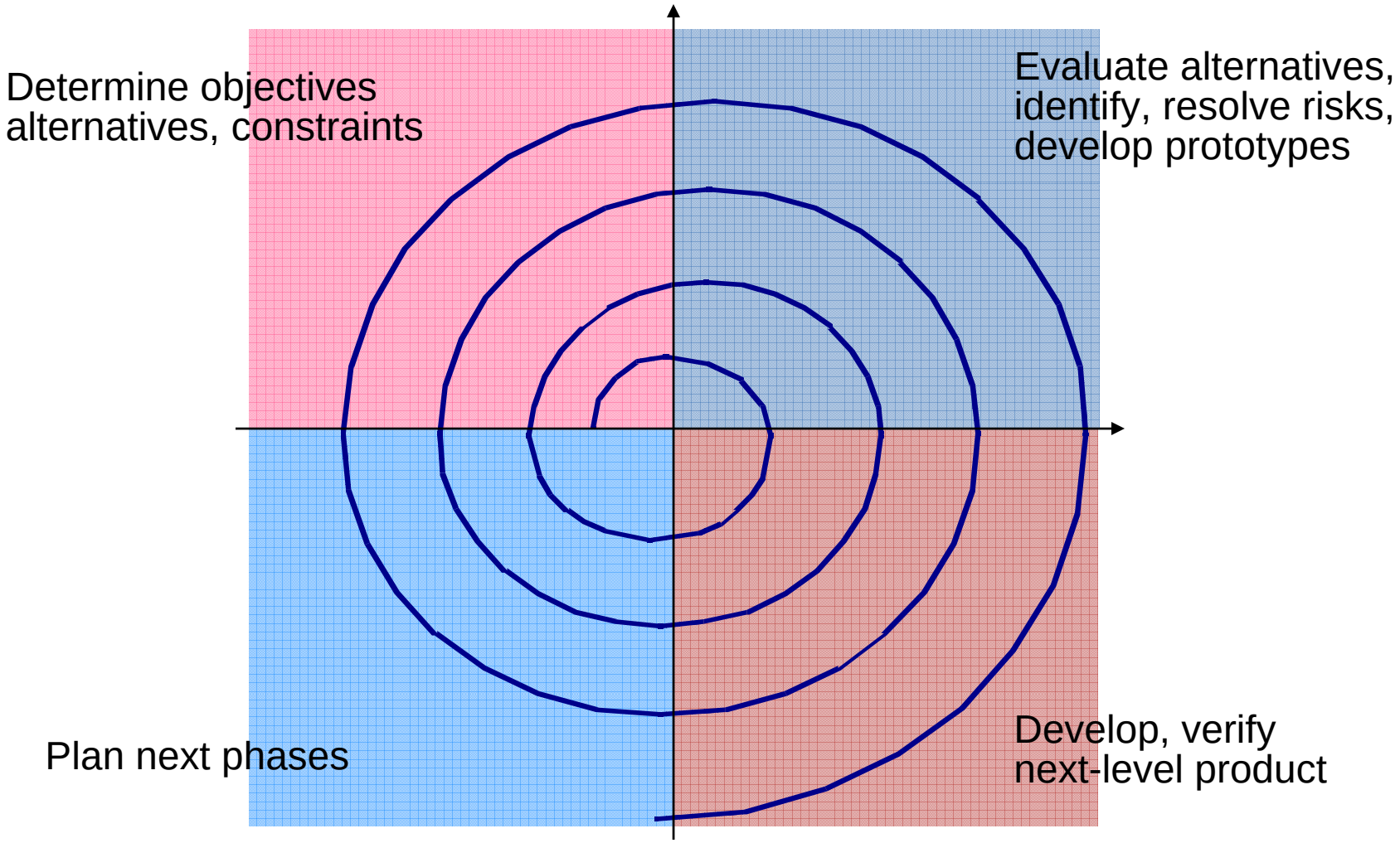
Development Processes

- Models
  - Iterative, incremental
  - Agile
- Organizational/business strategy
  - Software Product Line
  - Open source

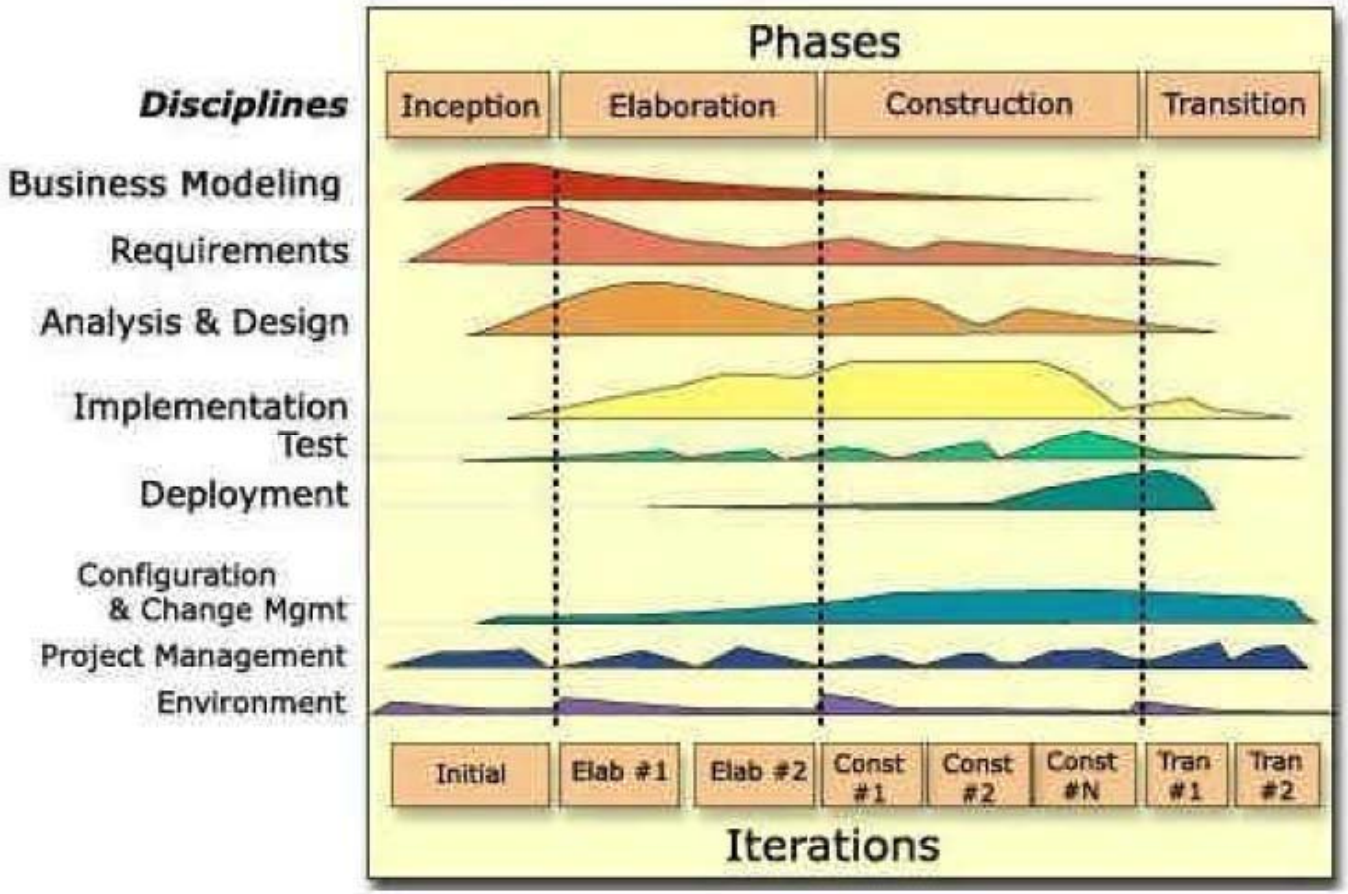
Software Development Lifecycle
Waterfall Model



Software Development Lifecycle
Spiral Model



Rational Unified Process



Tools

- Eclipse
- Junit
- PSK
- EPF

Models

- Requirements/use cases/features
- Non-functional requirements
- Architecture

Non-functional requirements

- Qualities must be measurable
- Measurement requires that qualities be precisely defined
- Improvement requires accurate measurement
- Currently most qualities are informally defined and are difficult to assess

Practices

- 29 practices in software product line strategy
- Practice encompasses multiple processes
  - The testing practice includes unit testing, integration testing, etc.

Design

- Patterns
- Separation of Concerns
- MVC

Components

- Third-party software “pieces”
- Plug-ins / add-ins
- Applets
- Frameworks
- Open Systems
- Distributed object infrastructures
- Compound documents
- Legacy systems

Design of concurrent/distributed systems

- Process/thread
- Non-determinism
- Race conditions/deadlock

Change Management

- CM is a discipline whose goal is to control changes to large software through the functions of
  - Component identification
  - Change tracking
  - Version selection and baselining
  - Software manufacture
  - Managing simultaneous updates (team work)

Maintenance

- Maintenance costs > 60% of all development costs
  - 20% corrective
  - 30% adaptive
  - 50% perfective

Software Engineering Axioms

- Adding developers to a project will likely result in further delays and accumulated costs
- Basic tension of software engineering
  - better, cheaper, faster — pick any two!
  - functionality, scalability, performance — pick any two!
- The longer a fault exists in software
  - the more costly it is to detect and correct
  - the less likely it is to be properly corrected
- > 70% of all faults detected in large-scale software projects are introduced in requirements and design
  - detecting the causes of those faults early may reduce their resulting costs by a factor of 100 or more