CSE4006 Software Engineering

03. Perspective Process Models

prescriptive

Scott Uk-Jin Lee

Department of Computer Science and Engineering
Hanyang University ERICA Campus

1st Semester 2015

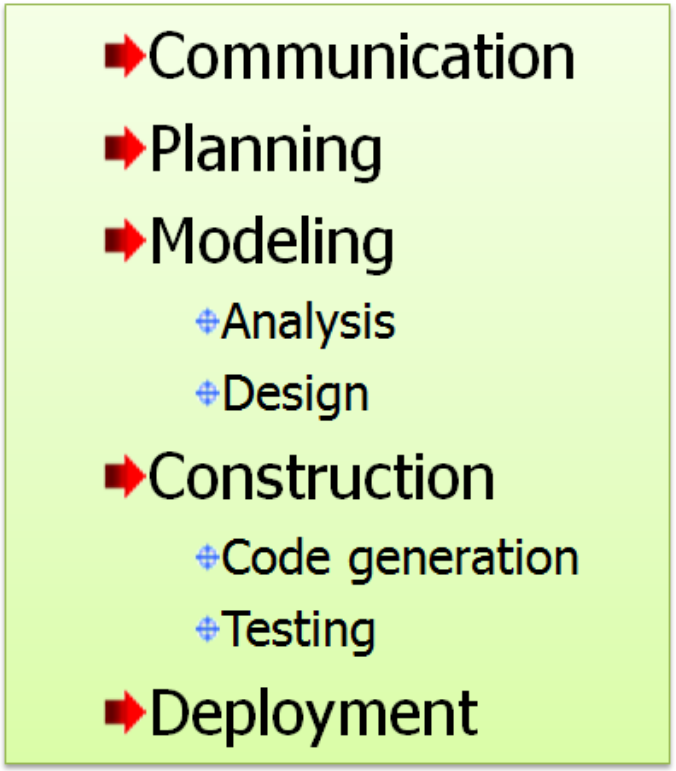
lab(se);

Prescriptive Models

- Prescriptive process model advocates an **orderly** approach to software engineering
- software life cycle
- prescriptive process model
  - development steps (operations), input data, results
  - Development teams need to establish a unique model that is suitable for the situation
  - Waterfall model, Incremental Model, RAD Model, Evolutionary Models, Unified Process

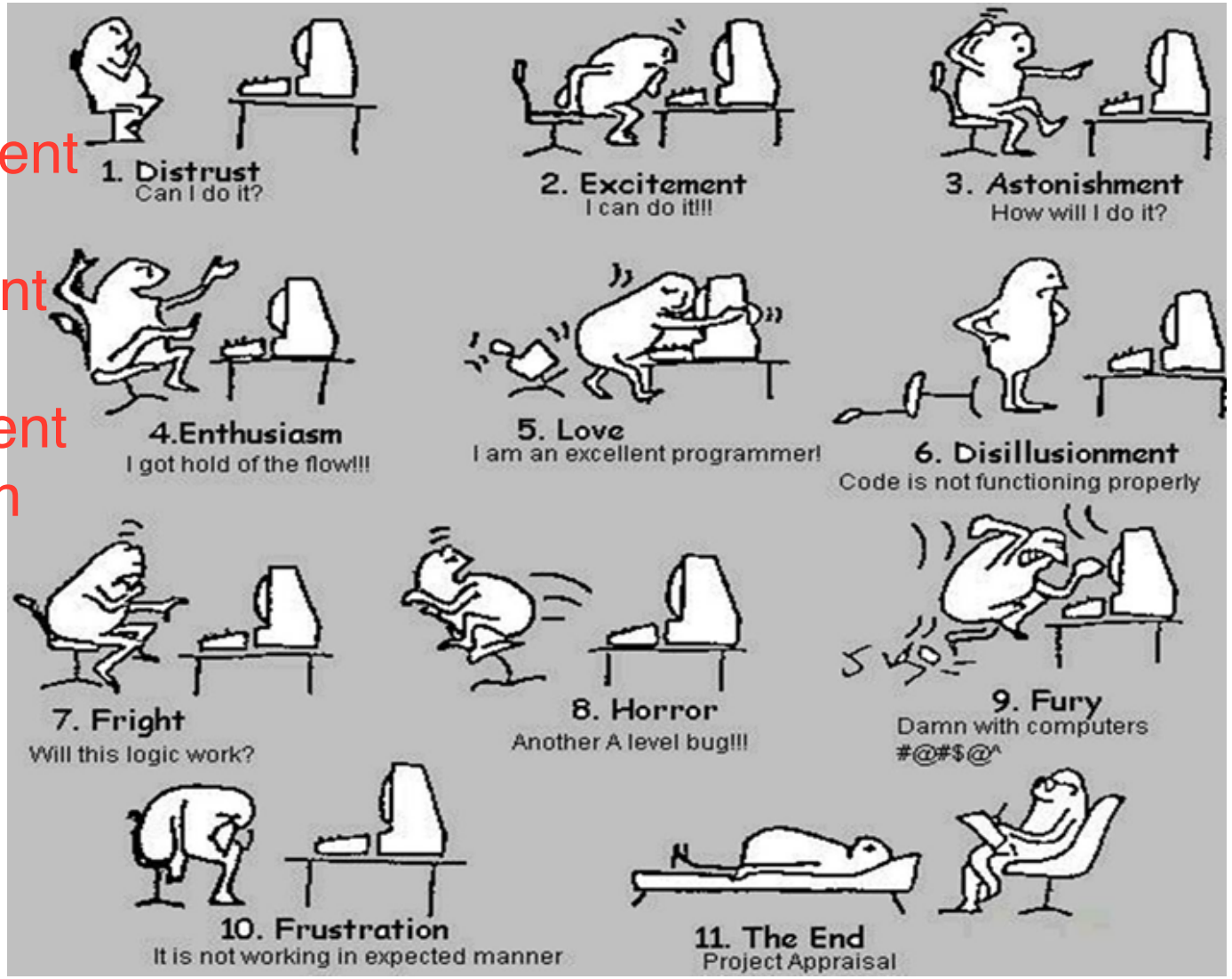
Prescriptive Models

- Prescriptive process model are based on framework activities



Software Development Life Cycle

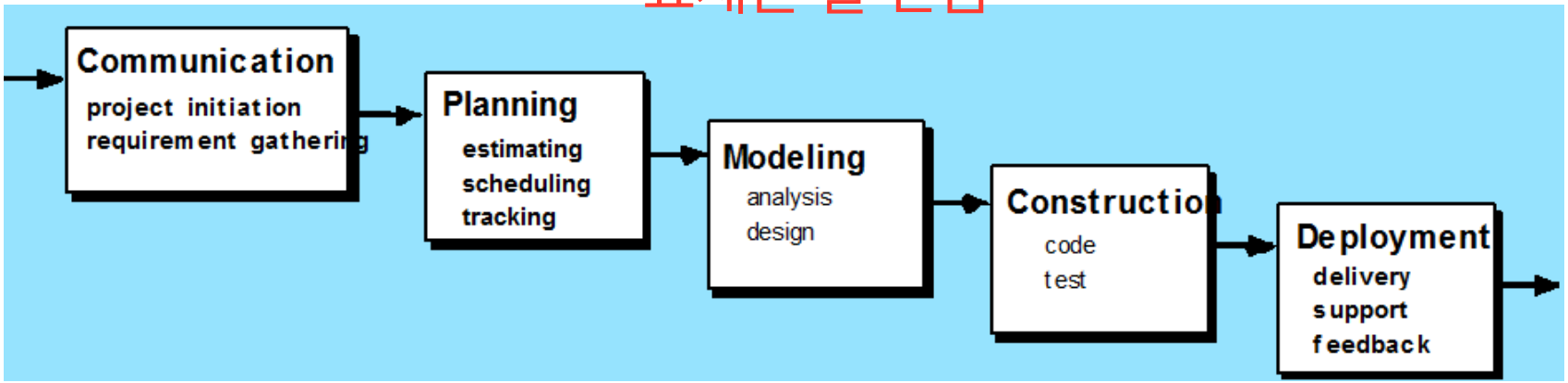
Requirement
Design
Implement
Test
Deployment
Maintain



lab(se);

Waterfall Model

옛날 방식
계속 requirement가 바뀌는 경우가 많기 때문에
요새는 잘 안씀



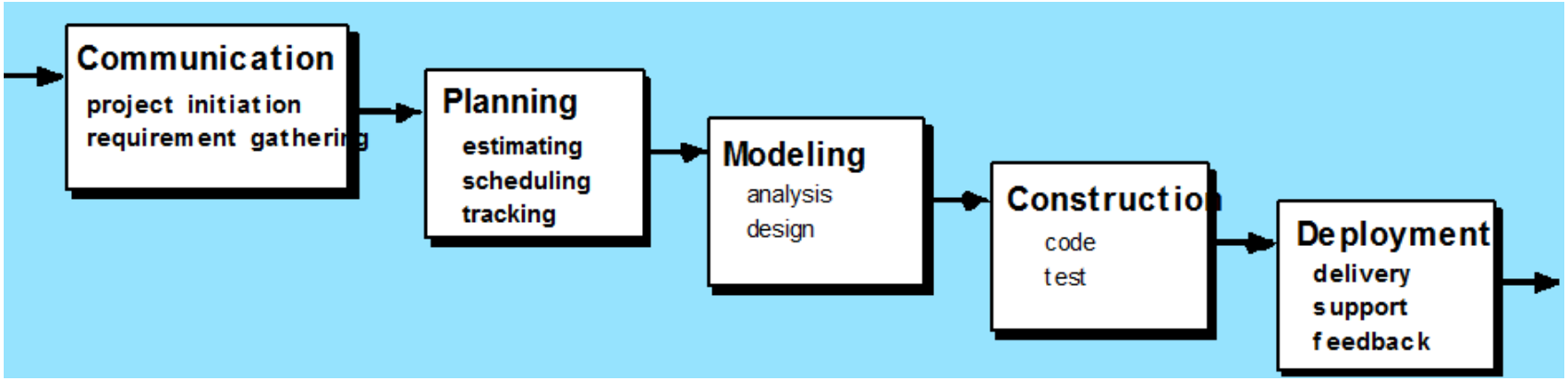
- What problems does the waterfall model have?
  1. Real projects rarely follow the sequential flow
  2. Difficult to accommodating the uncertainty in requirements
  3. A working version of software will not be available until late in the project

Waterfall Model

- Invented in the late 1950s for large air defense systems and popularized in the 1970s
- Each stage must be finished before the next stage begins
  - Ordinal: no interaction or overlap between each stage
  - Results of each stage must be checked before starting the next stage
  - Feedback to the immediate previous stage
- Suitable for developing simple system or if you are familiar with the application area
  - suitable for one-off process
  - suitable for developing a system to be utilized by a non-professional
- deliverable definition is important

lab(se);

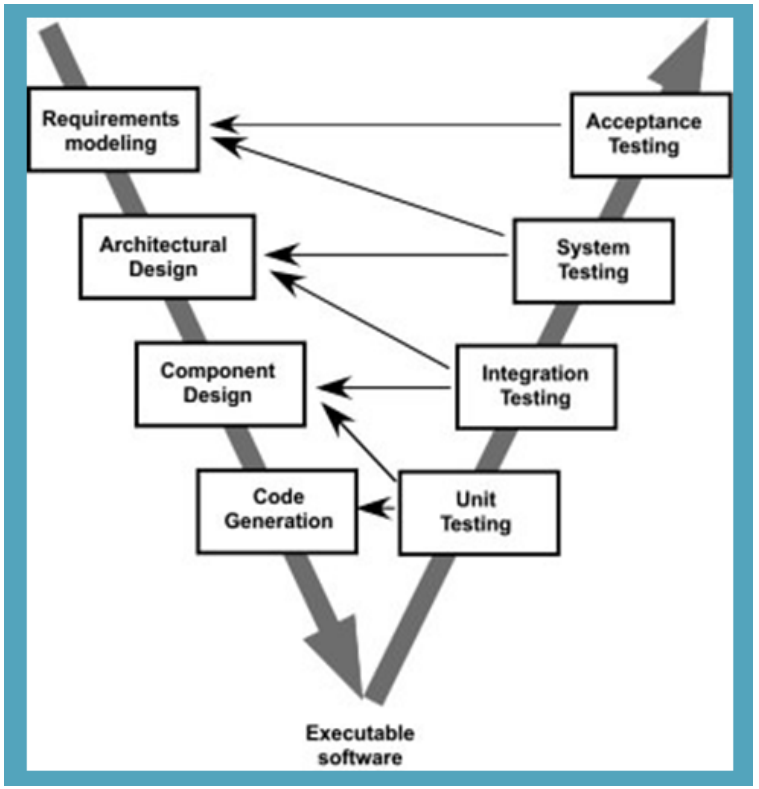
Waterfall Model



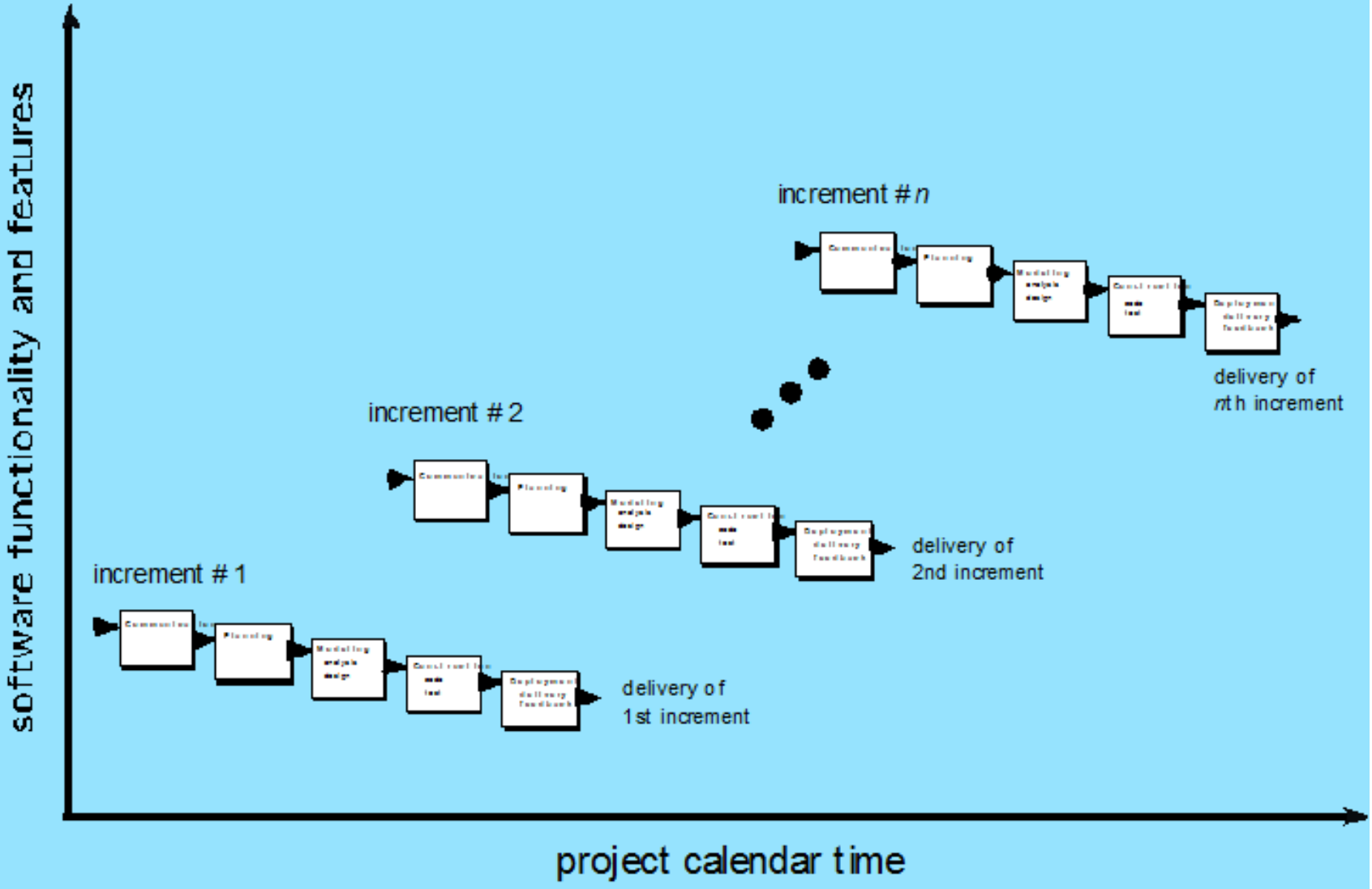
- Bradac found that the linear nature of the waterfall model leads to **blocking states**
  - where some members must wait for other members of the team to complete dependent tasks
  - especially, at the beginning of the project
- Still, however, the waterfall model serve as a useful process model where requirements are fixed and work is to proceed to completion in a linear manner

V Model

- Variation of the Waterfall model
  - focuses on tasks and verification of result
- Pros : can reduce errors
- Cons : not easy to deal with changes as there is no repetition
- Suitable for areas where high reliability is required

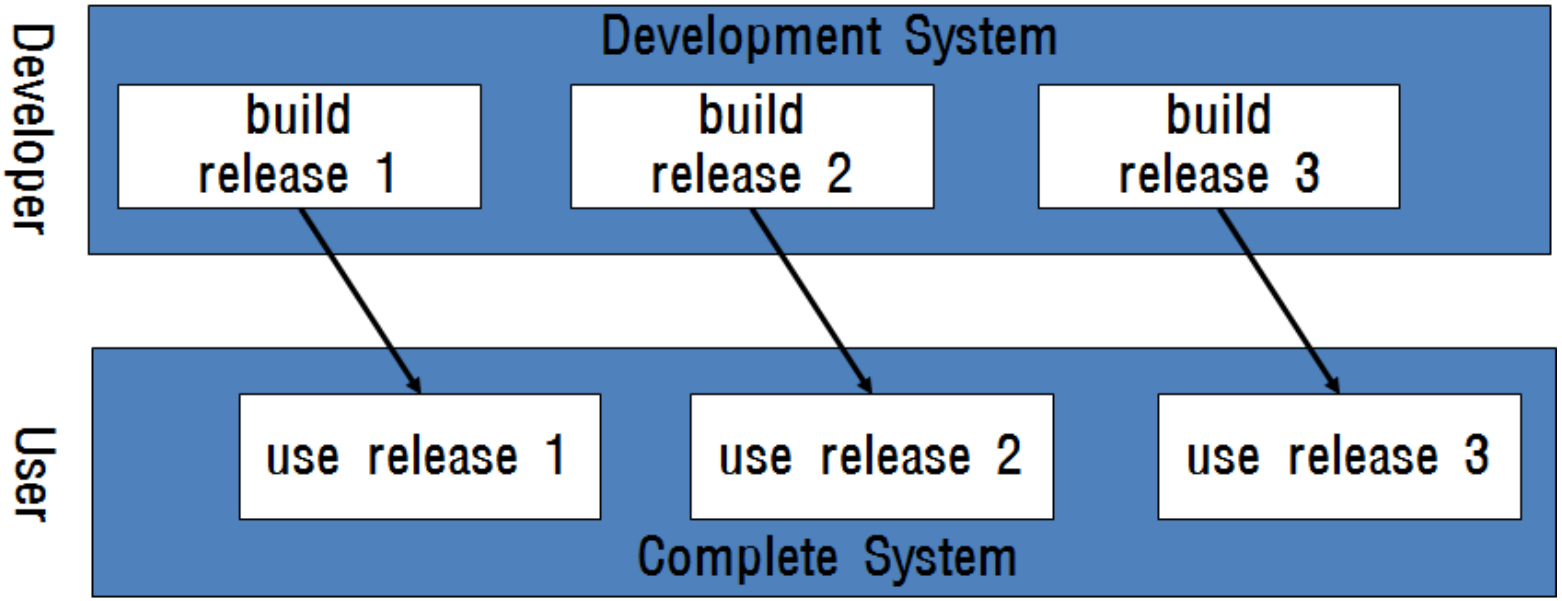


Incremental Model



Incremental Model

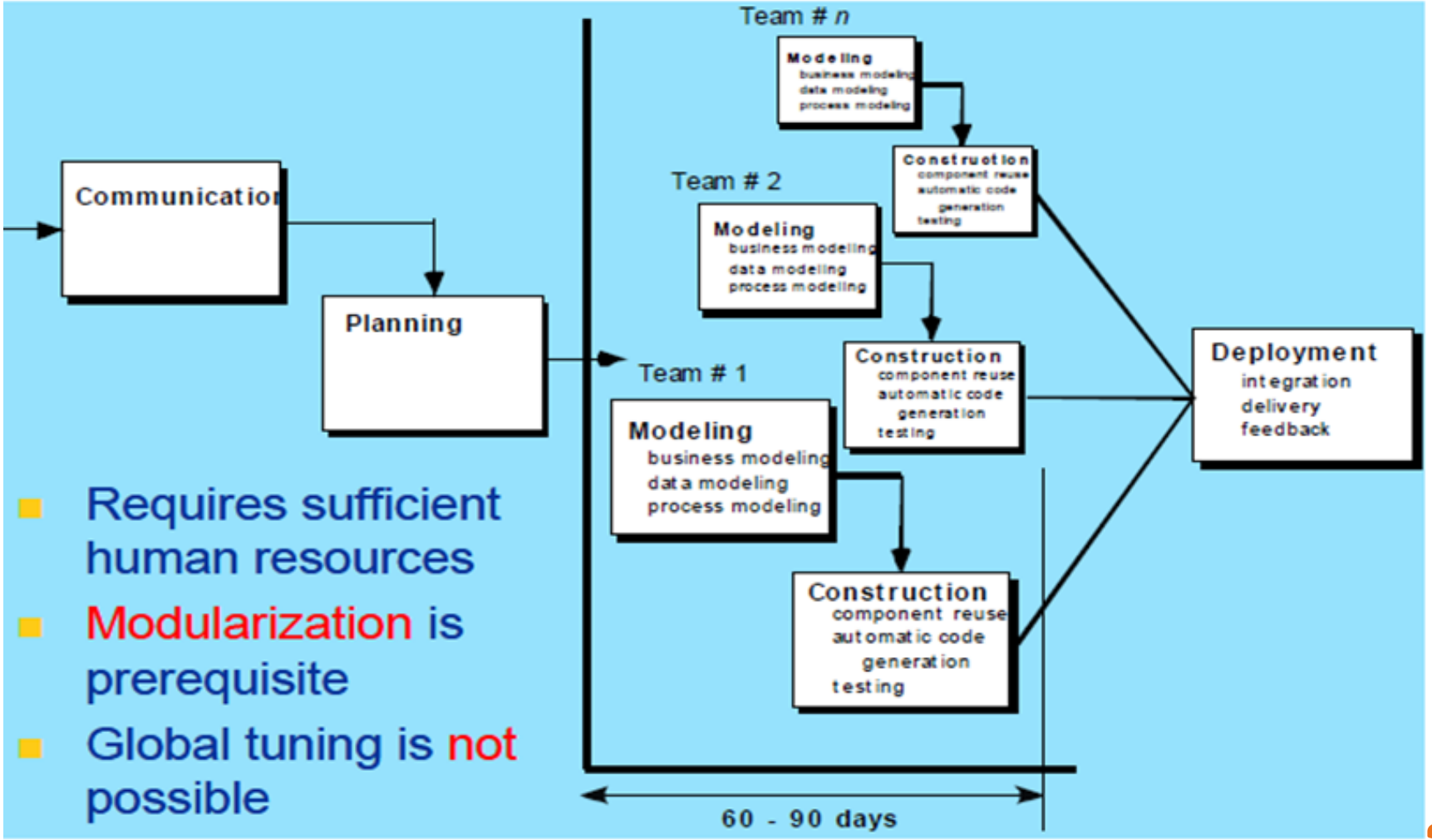
- Environment where development cycles are short
  - time-to-market is directly related to profits
  - way to reduce development time : divide the system and deploy



Incremental Model

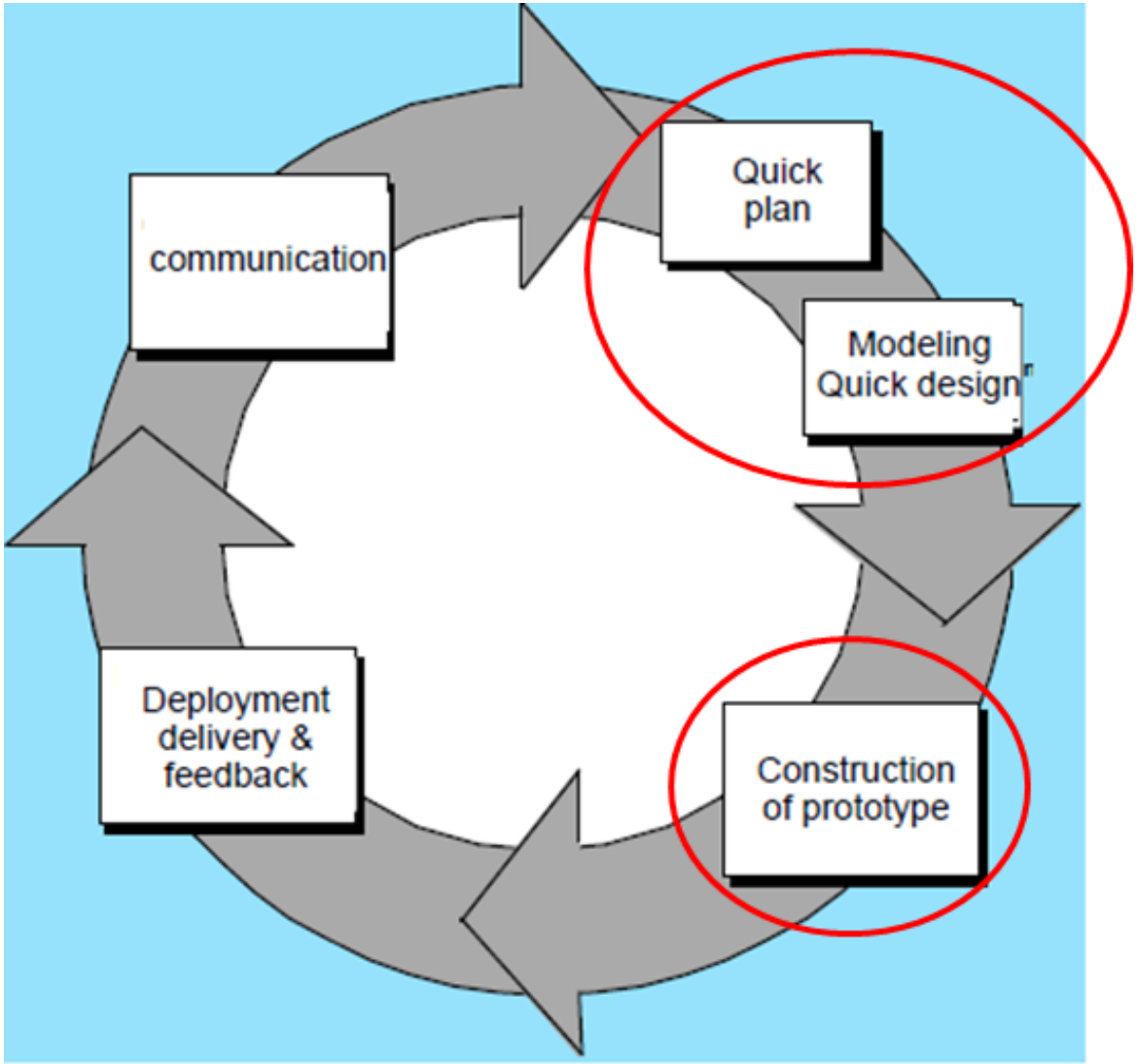
- Release configuration method
  - incremental method: deployment based on functionality
- Incremental development
  - although the lack of functionality, early usage education is possible
  - the first to market software leads to rapid market emergence
  - frequent deployment enables the quick and continual correction of unexpected problems occurring in the running system
  - development team can focus on different area of specialization for each release

Rapid Application Development (RAD) Model



- Requires sufficient human resources
- Modularization is prerequisite
- Global tuning is not possible

Evolutionary Model: Prototyping



Evolutionary Model: Prototyping

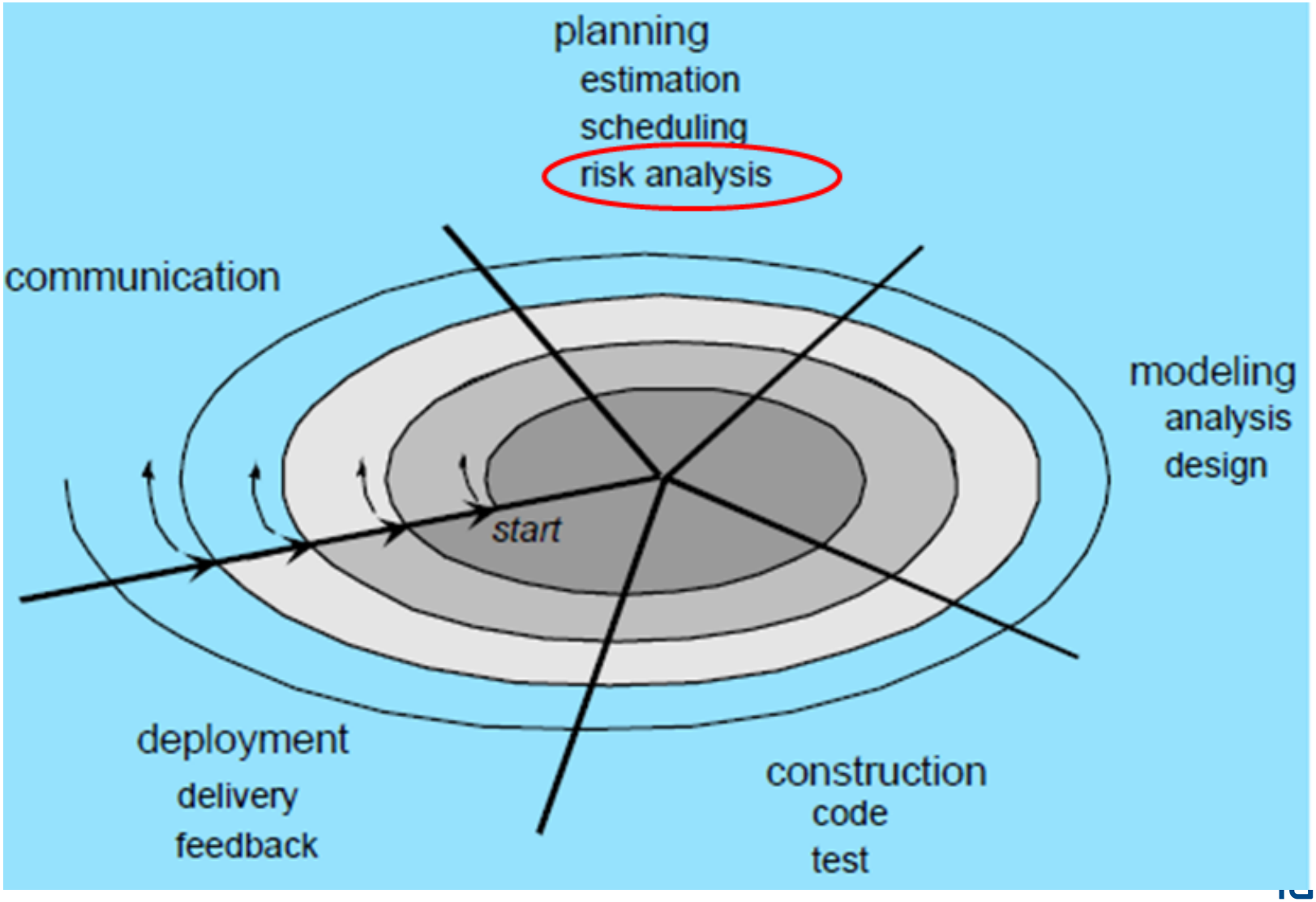
- Prototyping application
  - extract customer's requirements more accurately
  - test feasibility of algorithm, test compatibility with operating system, trial of interface creation
- Prototyping tool
  - screen generator, visual programming, 4th generation programming language
- Prototyping provides reference model
  - provides mechanism for users and developers to communicate
- Prototyping usage
  - simply to extract requirements: one-off usage
  - to predict possibility of implementation: incorporates maintenance during the development stage

Evolutionary Model: Prototyping

- Pros
  - users opinions are well reflected
  - user can participate with an interest and developers can extract requirements accurately
- Cons
  - causes misunderstanding and expectations (on quick delivery of the product)
  - difficulties in management (definition of intermediate deliverables are difficult)
- Applications
  - when the detailed requirements for software is not clearly identified
  - when software engineers are not sure of the efficiency of algorithm, usability of software, etc
  - when you want to try a new and innovative technology

lab(se);

Evolutionary Model: Spiral



Evolutionary Model: Spiral

- Divide software functionalities and develop incrementally
  - reduces the possibility of failure, ease of testing, feedback
- Incremental releases
- Evolution stage
  - Planning: select goals and feature, decide on constraints
  - Estimation: evaluate the previous stage and estimate the cost
  - Scheduling
  - **Risk analysis**: analyze priority of feature selection and risk factors

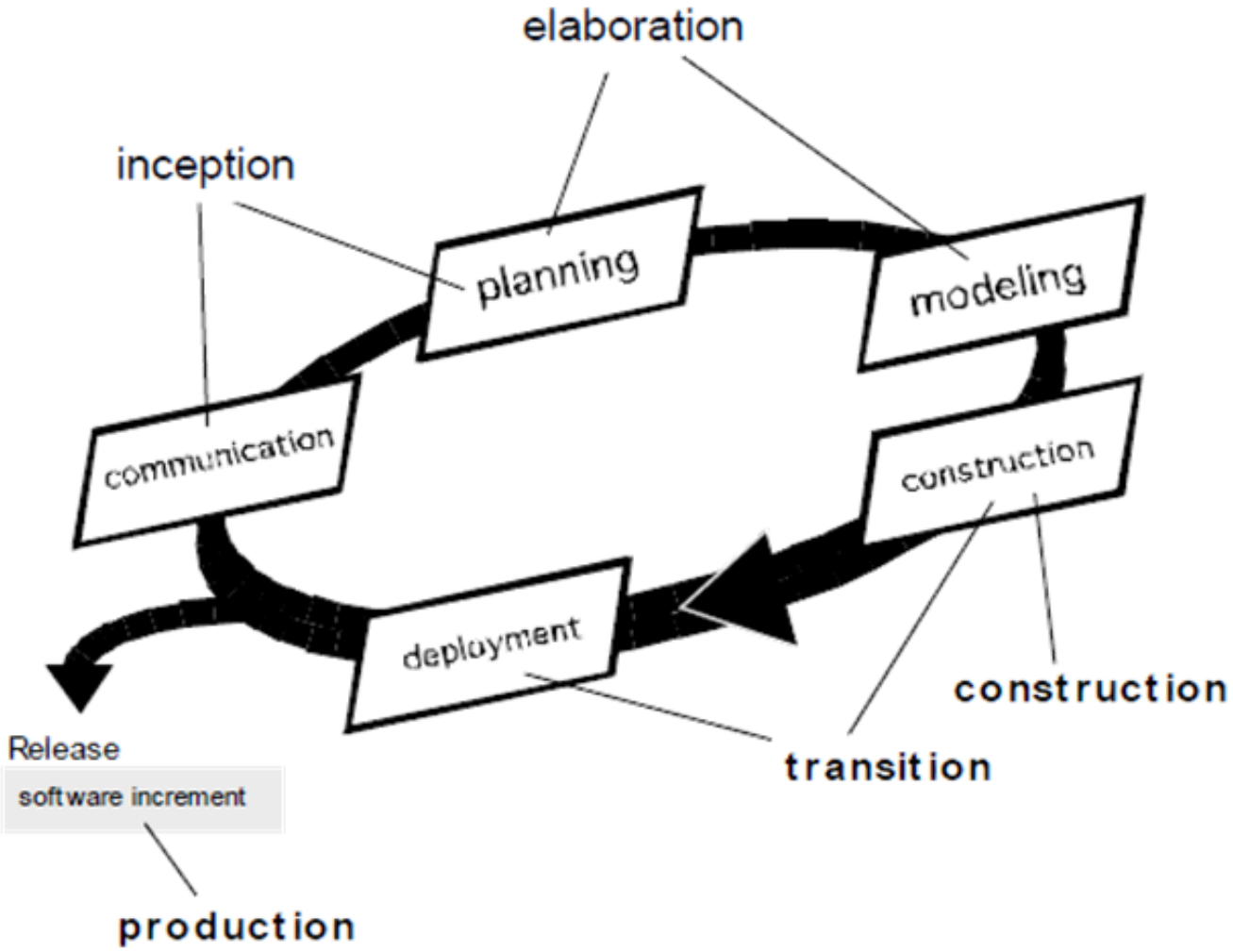
Evolutionary Model: Spiral

- Pros
  - suitable for large-scale system development: risk reduction mechanism
  - iterative development and testing: toughness improvement
  - functionality that is not included in a cycle can be added in next cycle
- Cons
  - management and risk analysis is important
- Applications
  - when financial or technical risk is large
  - when it is difficult to understand the requirements or architecture

Other Process Models

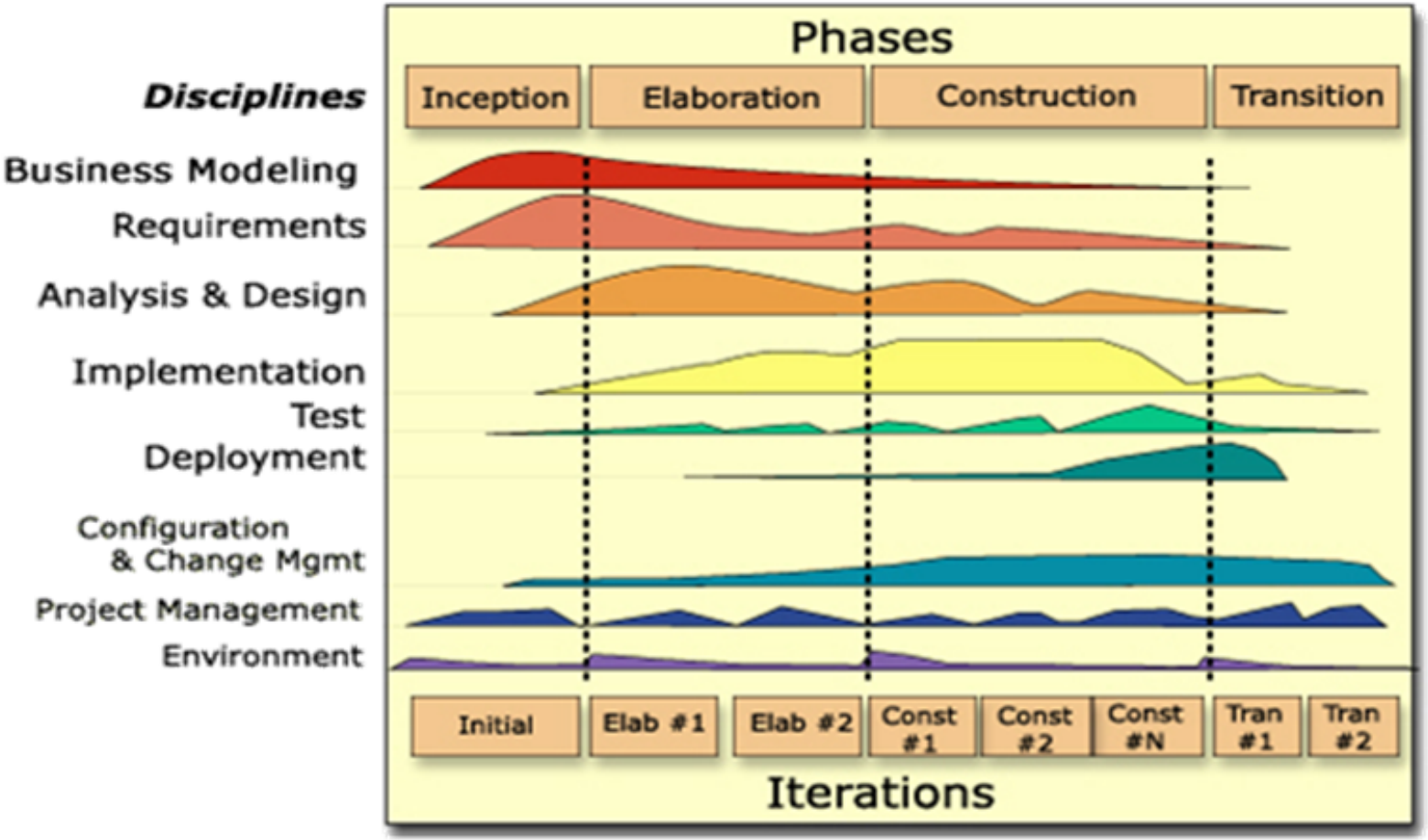
- Component based development
  - the process to apply when reuse is a development objective
- Formal methods
  - emphasizes the mathematical specification of requirements
- AOSD
  - provides a process and methodological approach for defining, specifying, designing, and constructing aspects
- **Unified Process**
  - a "use-case driven, architecture-centric, iterative and incremental" software process closely aligned with the **Unified Modeling Language (UML)**

Unified Process (UP)



UP Phases

RUP: Rational UP by IBM



UP Work Products

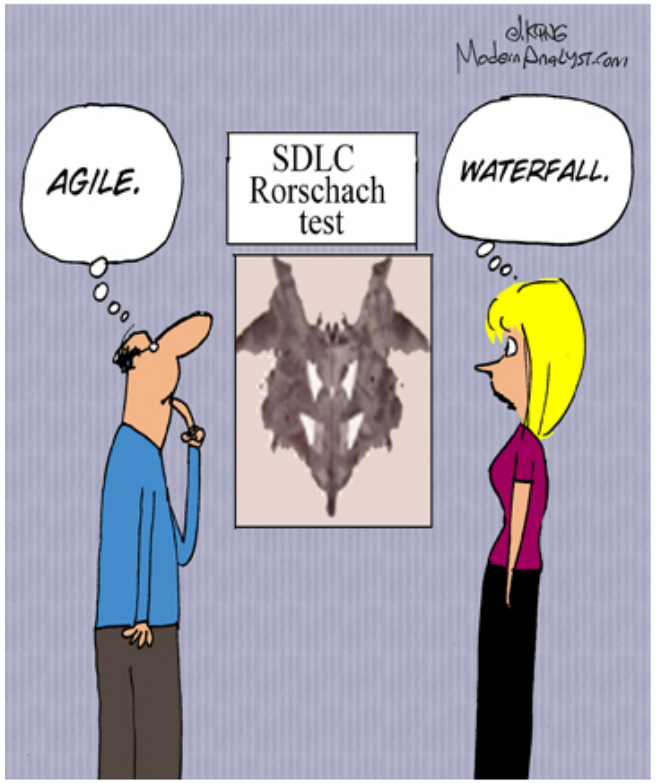
Inception phase
- Vision document
- Initial Use-Case model
- Initial project glossary
- Initial business case
- Project plan – phases & iterations
- Business model if necessary
- One or more prototypes

Elaboration phase
- Use-Case model
- Supplementary req. including non-functional
- Analysis model
- Software architecture description
- Executable architectural prototype
- Preliminary design model
- Revised risk list
- Project plan including
  - iteration plan
  - adapted workflows
  - milestones,
  - technical work products
- Preliminary user manual

Construction phase
- Design model
- Software components
- Integrated software increment
- Test plan and procedure
- Test cases
- Support documentation
  - User manuals
  - Installation manual
  - Description of current increment

Transition phase
- Delivered software increment
- Beta test reports
- General user feedback

Selecting a Process Model



Process Model Restaurant Main Menu

Spiral model Risk reduction, build very stable system	₩12,000
Prototyping model Phased release, continuous improvement	₩10,000
Incremental model Divide system by functionality & build	₩11,000
Waterfall model Traditional, unique model	₩9,500
RAD model Appropriate for short schedule	₩10,000

Selecting a Process Model