CS3203: Software Engineering Project

Software Development Lifecycle

By: Dr. Bimlesh Wadhwa

Software Development LifeCycle (SDLC)

- A framework for software development
- Software development phases:
  - Analysis, Design, Implementation, Testing, Deployment, and Maintenance
- Also known as Software Development Process

SDLC Phases

- Requirements Analysis
  - a broad understanding of how the system will behave.
- Design
  - the technical architecture, technology exploration and prototype
- Implementation aka Building or Coding
  - code that fulfils the requirements.
- Testing aka quality assurance
  - test the system and ensure it does what you want it to do.
- Deployment
  - delivering the working system to the customer.
- Maintenance
  - fix post-deployment bugs and add new features

Artefacts:

Requirements document,
Architecture design,
Component design,
API design,
Source code,
Test Strategy,
Test Plans,
Test cases,
Installation manual,
Maintenance guide,
User guide,
Training manuals …..

Why should we use an SDLC

- A defined process to manage software complexities
  - Software development involves people, product, technology, …
    - » Requirements, design, specifications, code, test cases, …
    - » Phases, activities, milestones, …
    - » Organization of the team, communication channels, …
  - A common vocabulary for each step
  - Clearly-defined inputs and outputs from one step to the next

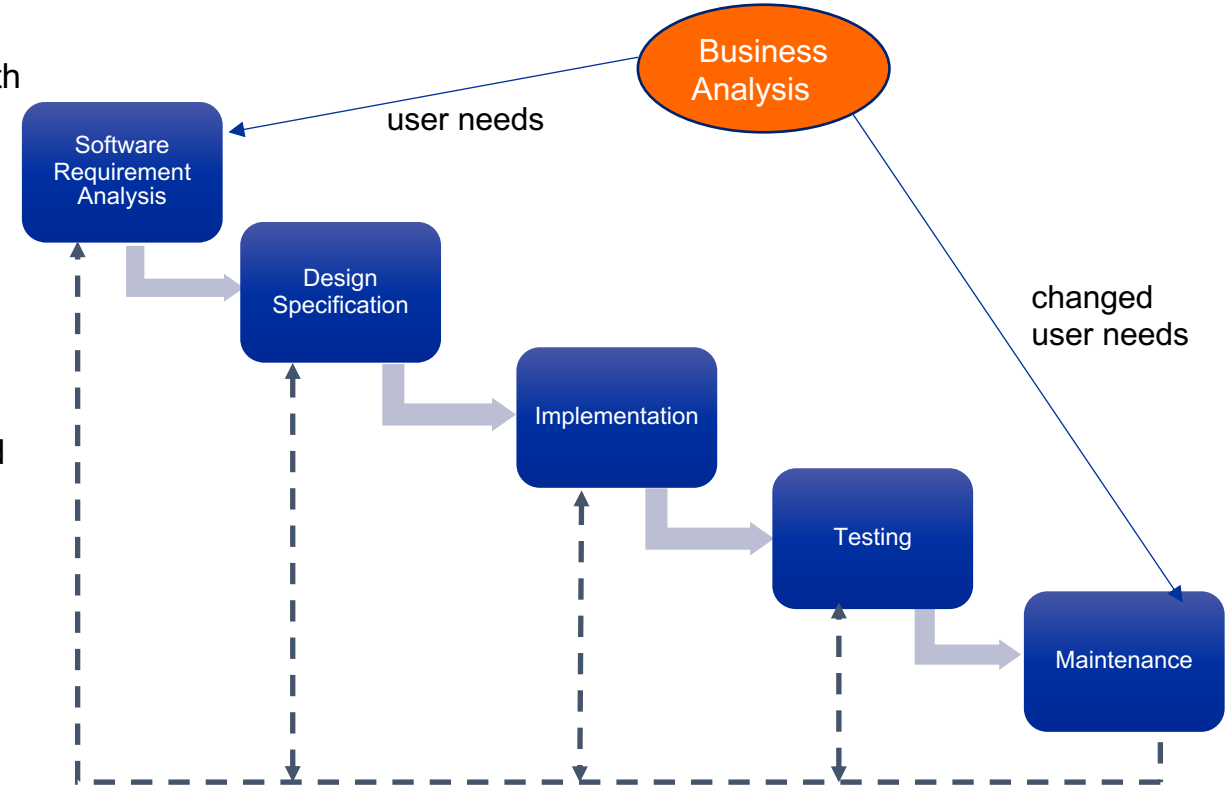
"If you fail to plan, you are planning to fail" - Benjamin Franklin

Popular SDLC models

- Waterfall
- Iterative
- Agile e.g. Scrum

Waterfall Model

- Origin - Royce1
- A highly structured, predetermined path through a set of phases
- Good for
  - stable requirements
  - familiar domain and solution
- Many variants in industry
- Rigid to adapt to changing requirements
- Restrictions considered draconian and counterproductive.
- Limited ability to deliver smaller and independently testable software modules.
- Waterfall approach makes evaluating and integrating new technology while developing software even more difficult.



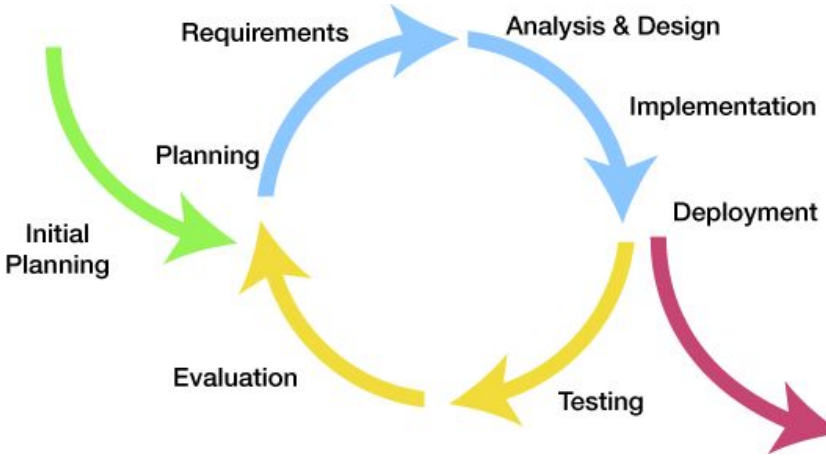
1. Managing Development of Large Software Systems, Royce W W, IEEE WESCON 1970, pp1-9

Iterative approach

- Development of a software through repeated cycles (iterations), handling smaller chunks of complexity at a time (increments).
- Developers learn from previous iterations and make changes to future iterations.

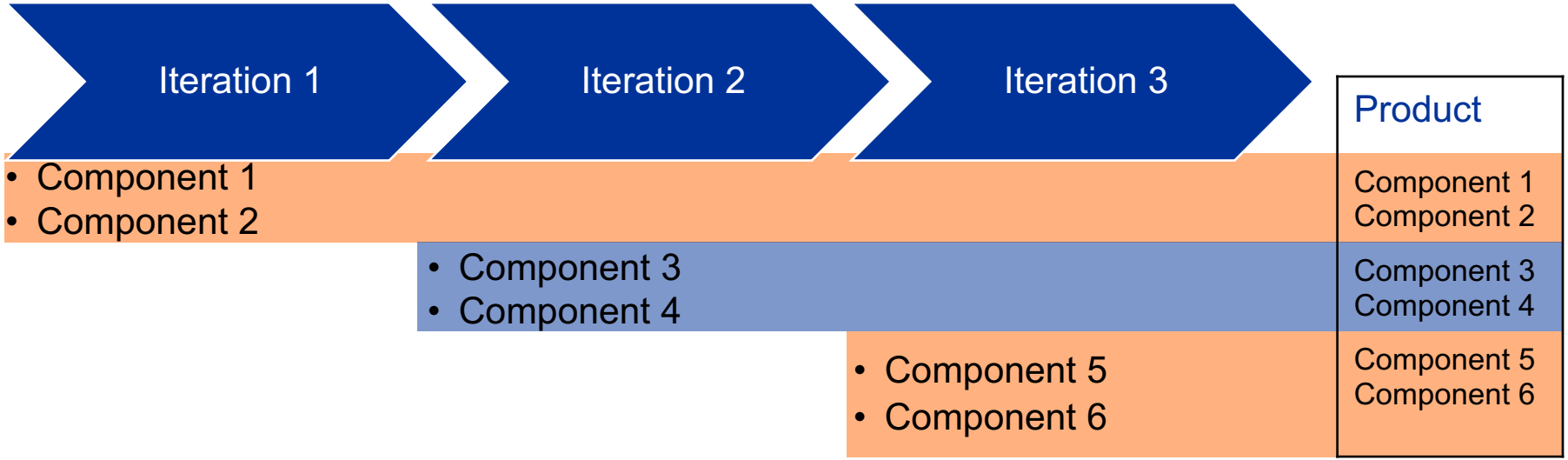
3 key concepts :

- **smaller chunks** (limited scope).
  - contain complexity and reduce risk.
  - User involvement in testing and acceptance during each increment and thus early feedback.
- **early feedback.**
  - Developers learn from previous iterations and apply changes to next iterations.
  - helps correcting costly decisions and implementations early on in the process.
- **repeated development cycles**.
  - a set of smaller waterfalls. Each part is shorter and less complex.



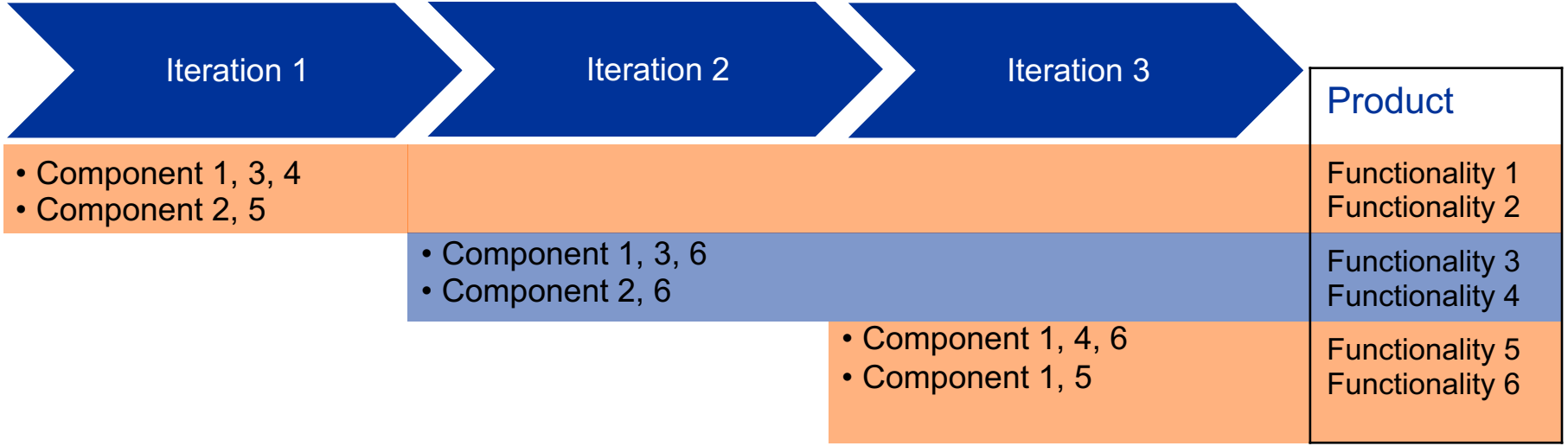
Depth-first Iterations

- An iteration focuses on developing some components



Breadth-first Iterations

- An iteration evolves all components in parallel; focus on completing some functionality.



Agile

- Early 2000s : pressure for faster delivery cycles, changing requirements placed emphasis on delivering testable software ‘quickly’ through collaborative efforts of self-organizing cross-functional teams and their customers

http://agilemanifesto.org/

Scrum

- An Agile Process designed for small teams
  - Iterative and incremental
- Requirements are converted to tasks that can be completed within time-boxed iterations, called *sprints*
  - Each Sprint (*iteration)*- commonly 2 weeks to 4 weeks
    - Fully integrated, tested, documented deliverable(software piece)
  - Sprint planning
    - Prepare sprint backlog, sprint goal,
    - 4 hrs for a 2-week sprint
  - Progress tracking in 15-minute time-boxed daily stand-up
    - Same time, same place
    - What was completed? What is planned for the day? What could be the impediments?
  - Sprint review to demonstrate the work completed and to review
    - 2 hr for a 2-week sprint
  - Sprint retrospective to review the past sprint for improvements
    - What went well? What did not go well? What could be improved for better productivity?
    - 1.5 hr for a 2-week sprint

https://scrumguides.org/scrum-guide.html

Other SDLC models

- Evolutionary Prototyping
- Rapid Application Development(RAD)
- Spiral
- Agile – Lean / Kanban / Scrumban / Extreme Programming/….
- Feature Driven Development
- Test Driven Development
- Domain Driven Design and Development
- Rational unified Process (RUP)
- DevOps

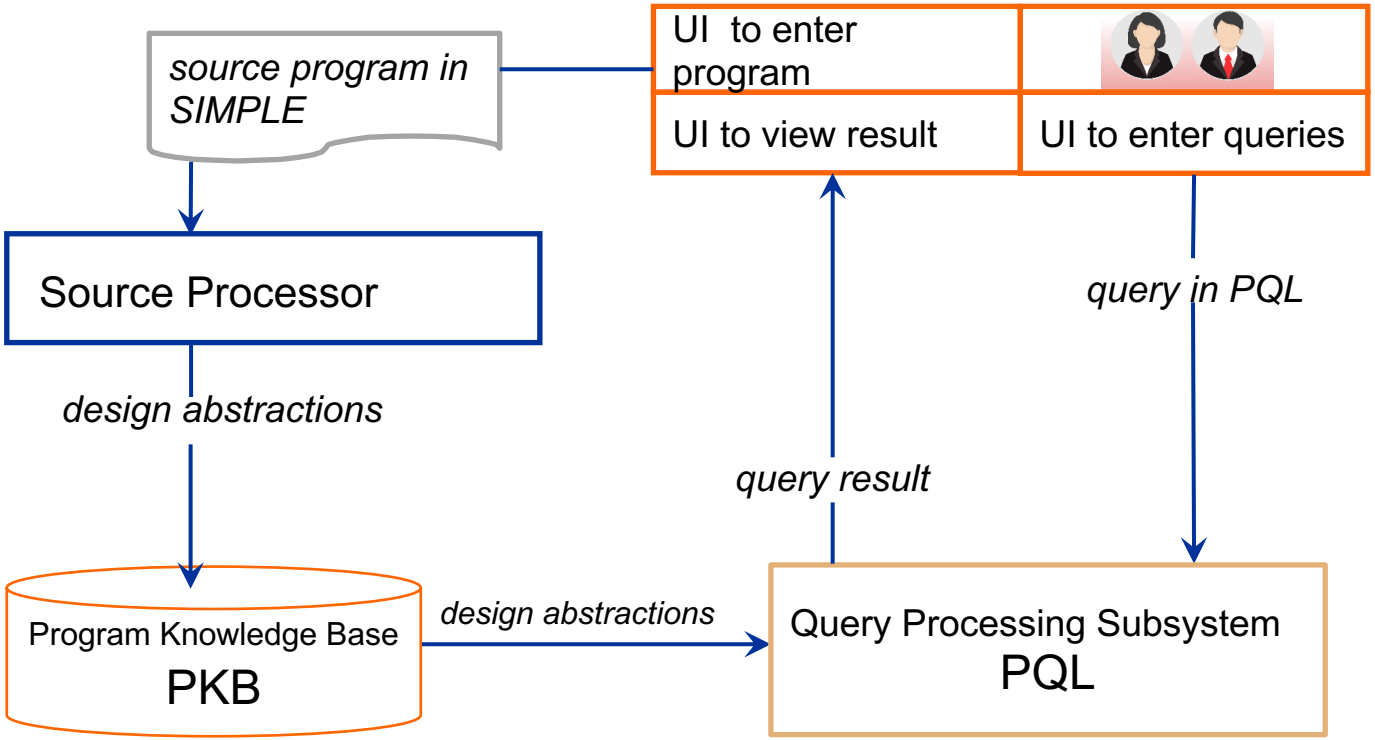
and many more

SDLC Common Practices

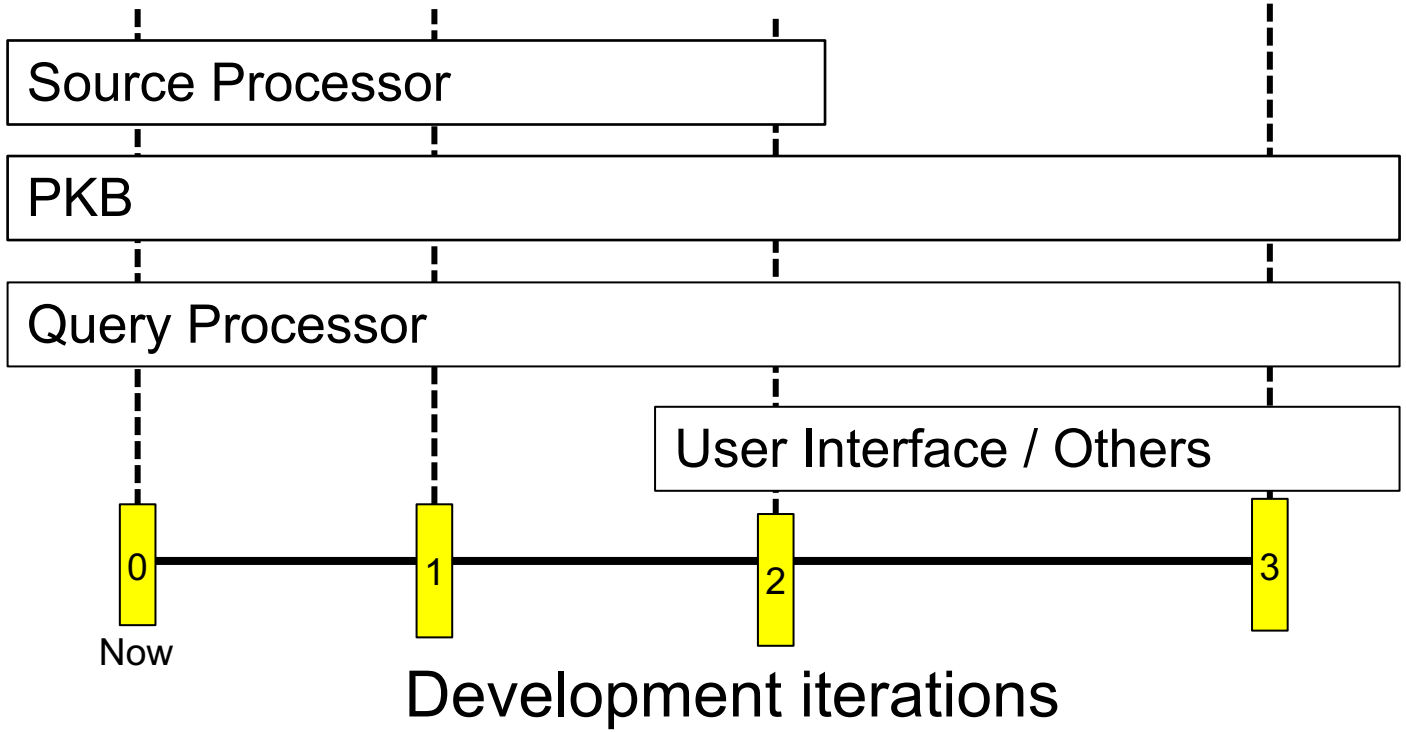
Practices that reduce risk and increase the chances of success.

- Source control
- Continuous integration
- Application management (governance, development and operations)
  - project management, bug tracking, and analytics to assist in decision making.

SPA Development



SPA Development Process



Documenting SPA Development Process

- Artefacts of development Process (see list on slide 3)
  - Balance between no documentation and excessive documentation
    - Most necessary and relevant
    - Simple and audience friendly
    - Use cross-links
    - Clear language
    - Up-to-date
    - Use visuals

Also, record how the development process is managed (Project management)

- *Plans, Estimates, Schedules, Roles, Standards, meeting notes, working papers, Deliverables, Process flow diagrams*
- *Reflective of how time and human resource used during development*

Summary

- An SDLC is like a set of ‘good habits’ that define the sequence of steps performed for developing a software.
- There are many different SDLC models practiced in the industry.
- Having well defined and documented SDLC is an essential element of a quality system.