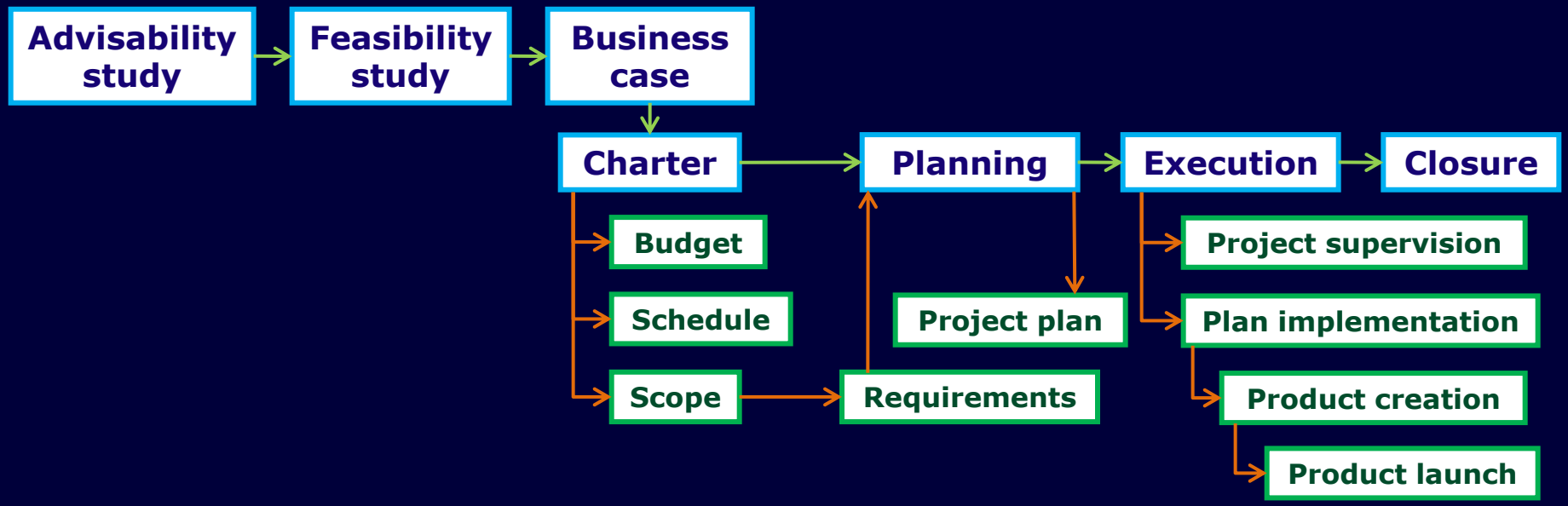
Project Management

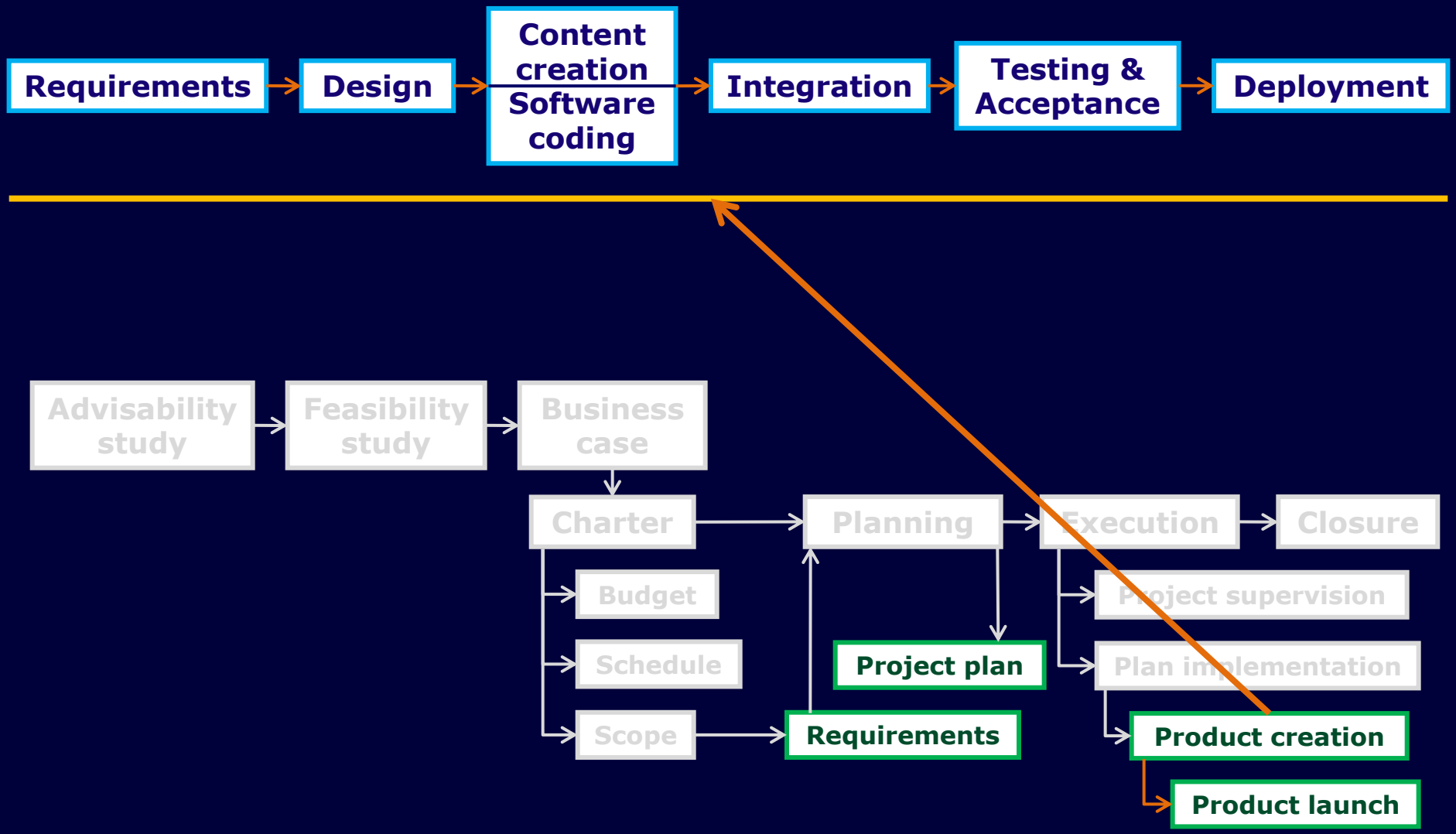
Software development models & methodologies

neil@minkley.fr

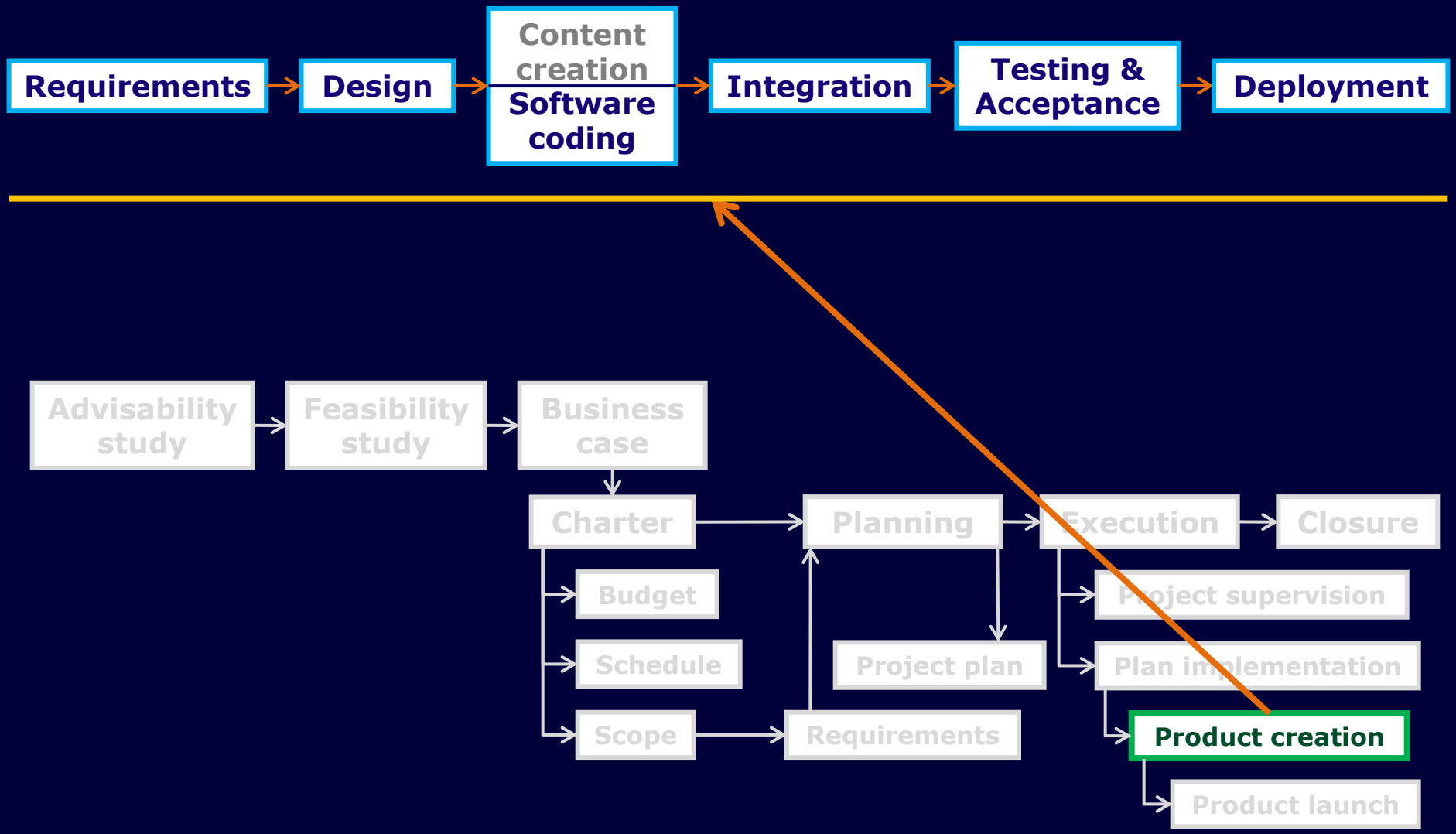
Project life cycle



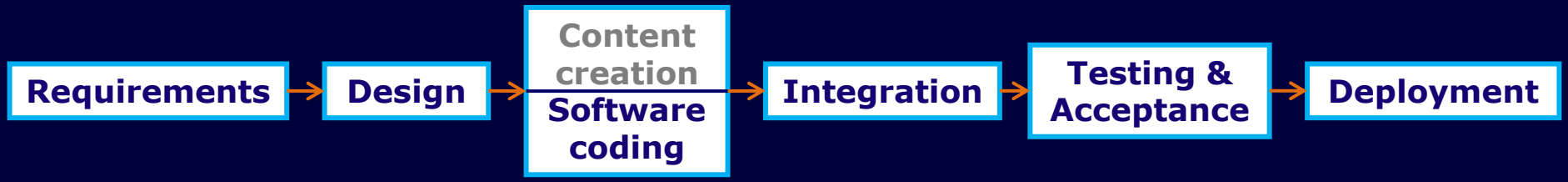
Product creation phases



Software development phases



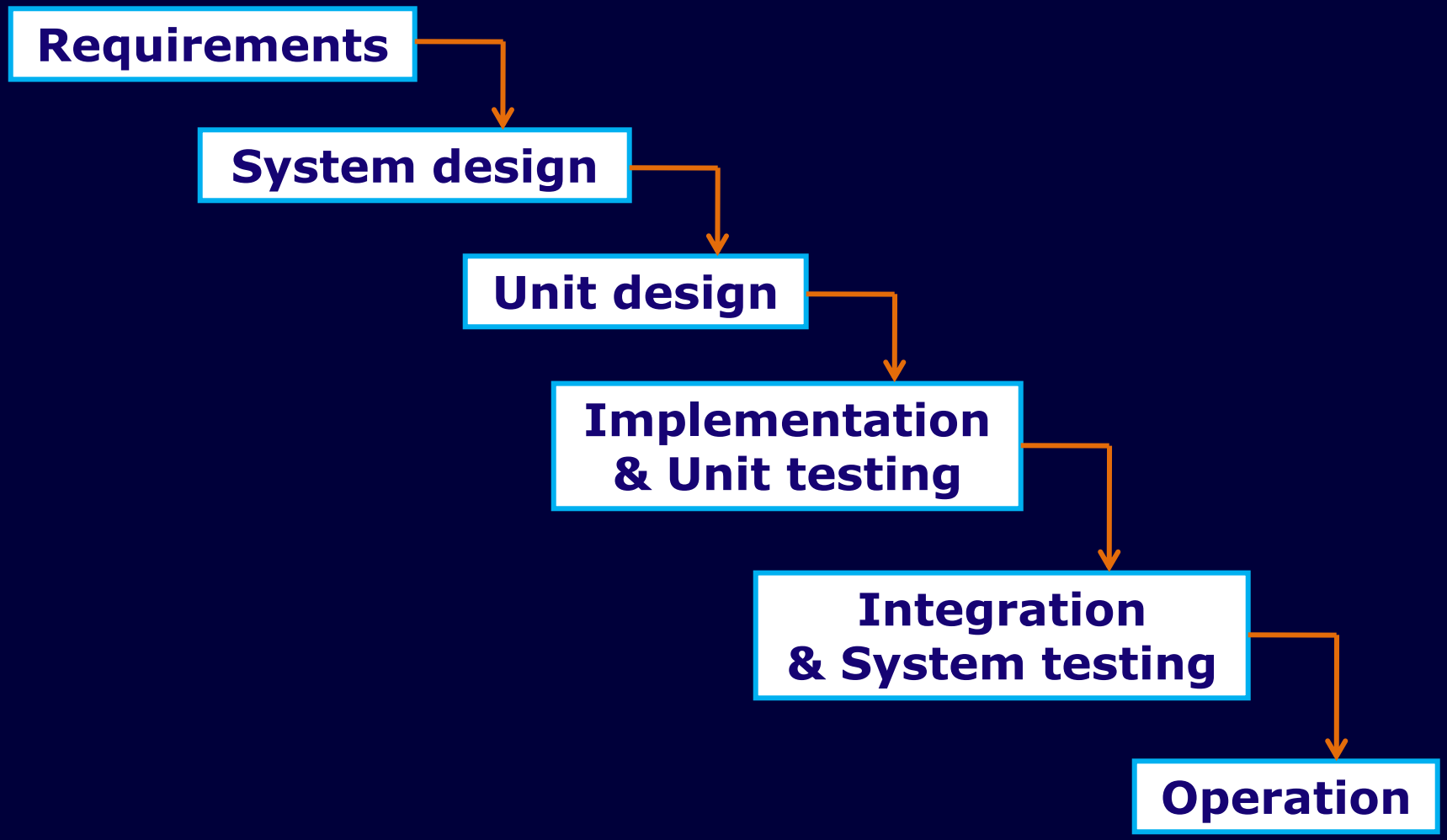
Software development life cycle models



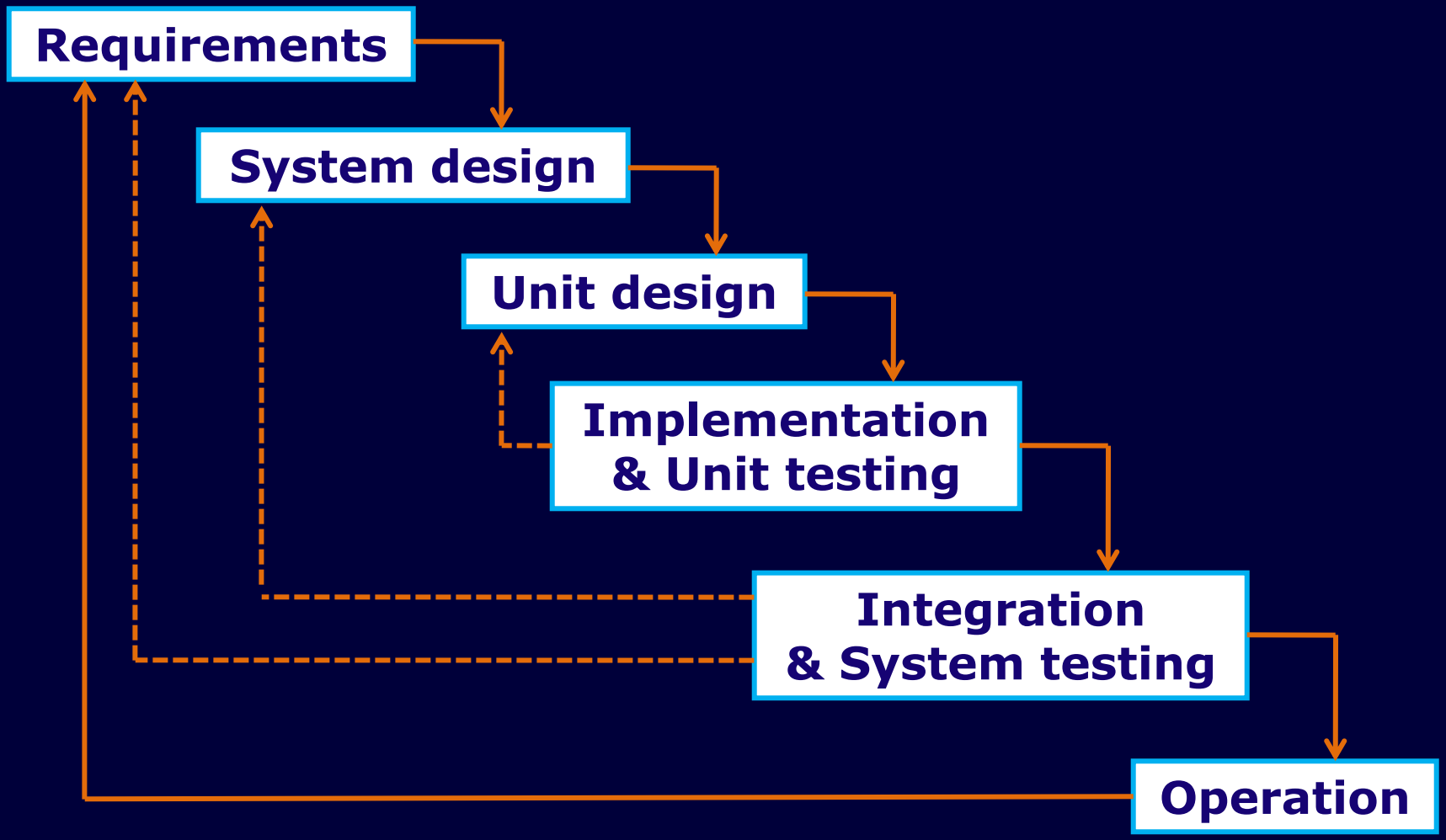
- Waterfall model
- Incremental model
- V model
- Spiral model

- Agile software development
- XP (eXtreme Programming)

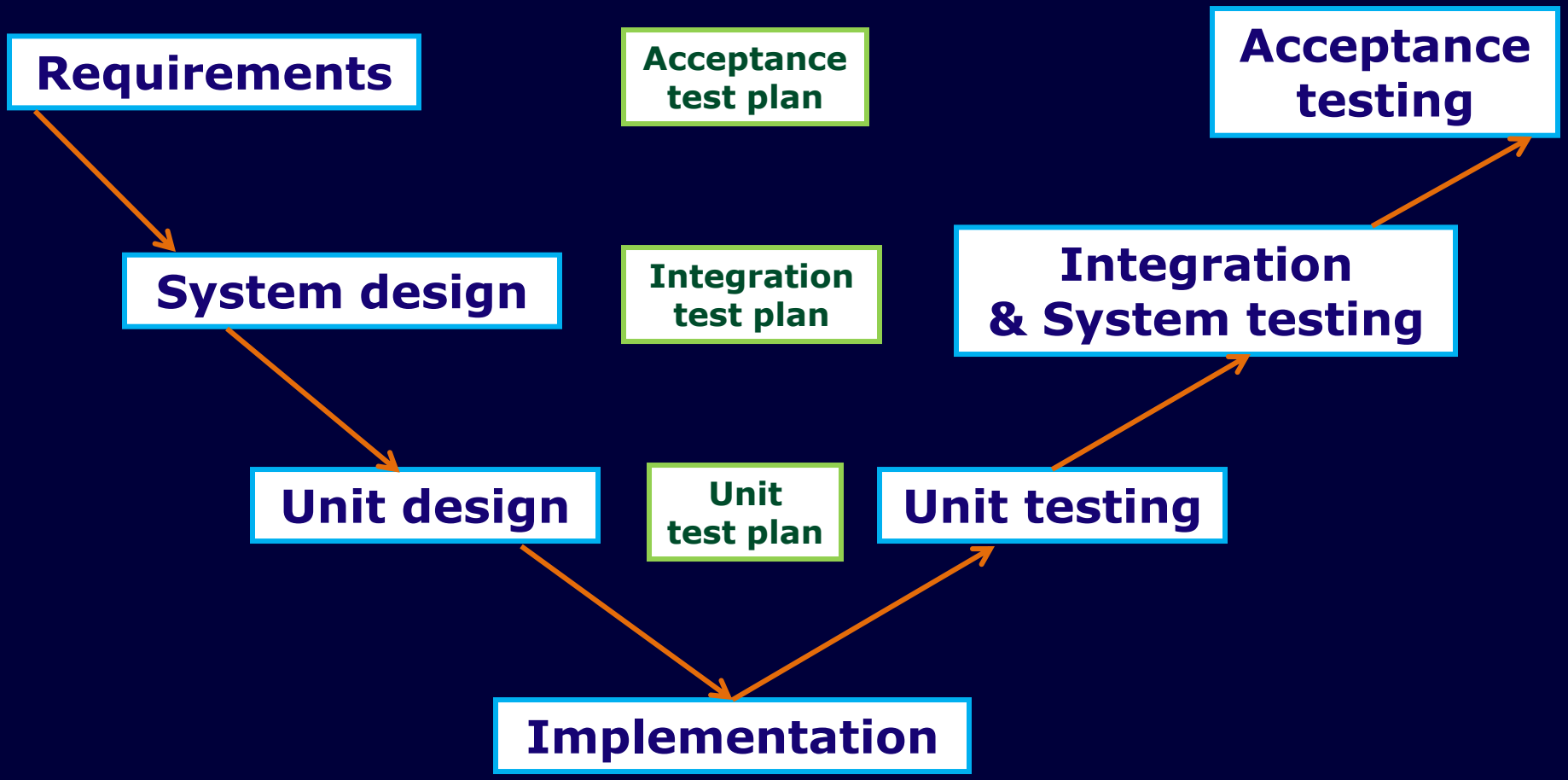
Waterfall model



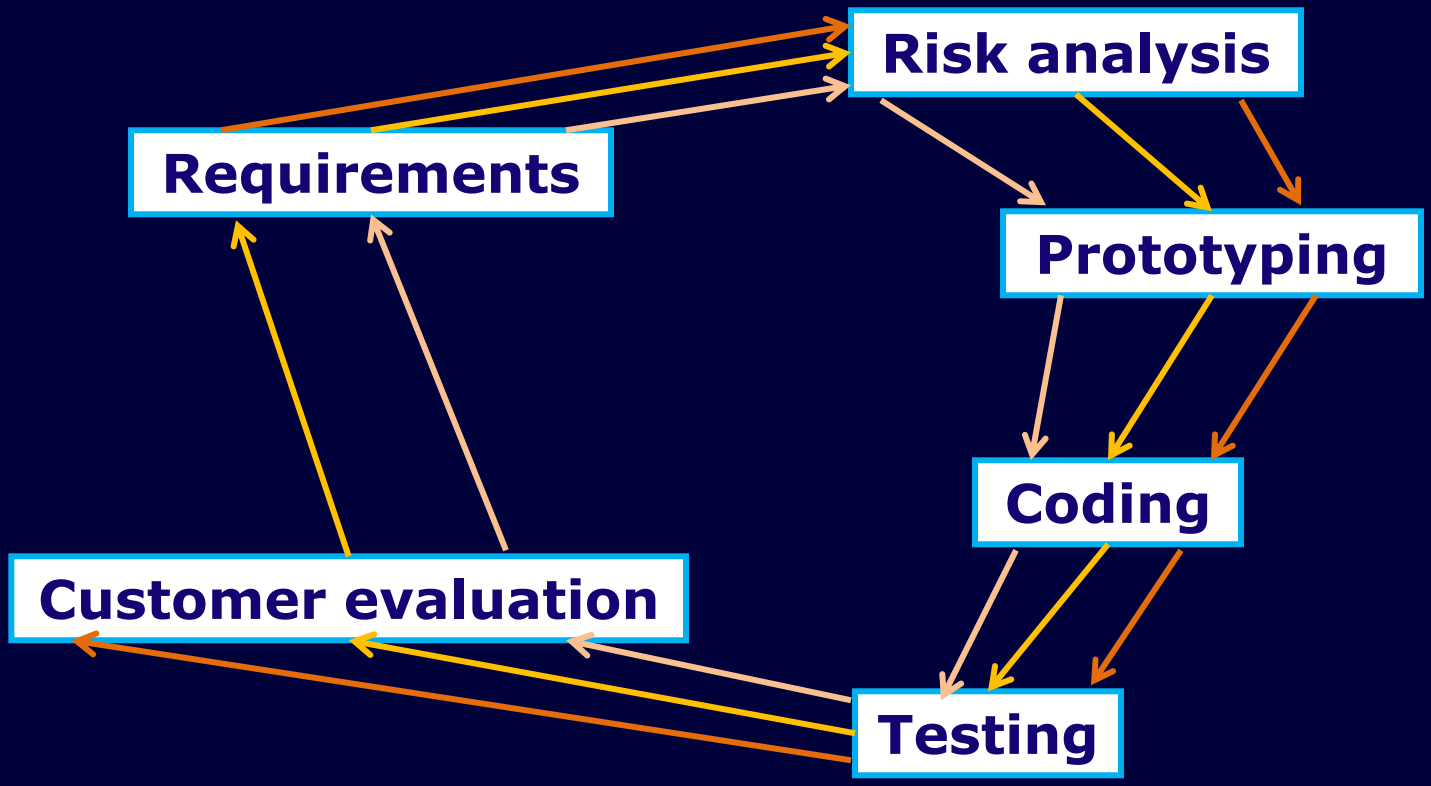
Incremental model



V model

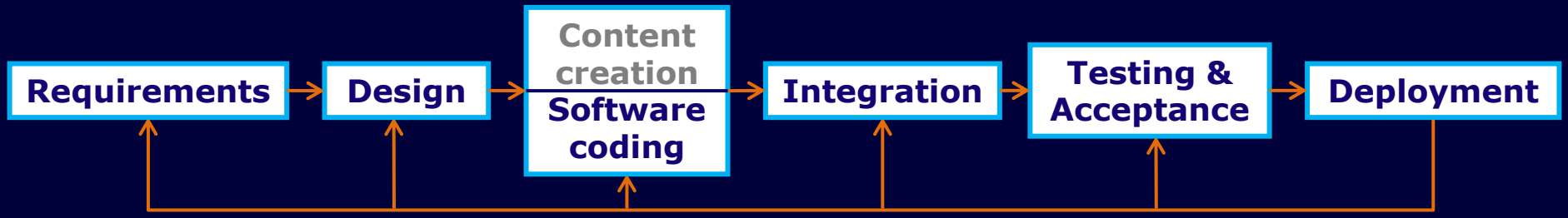

Requirements
Acceptance test plan
Acceptance testing
System design
Integration test plan
Integration & System testing
Unit design
Unit test plan
Unit testing
Implementation

Spiral model



Questions?

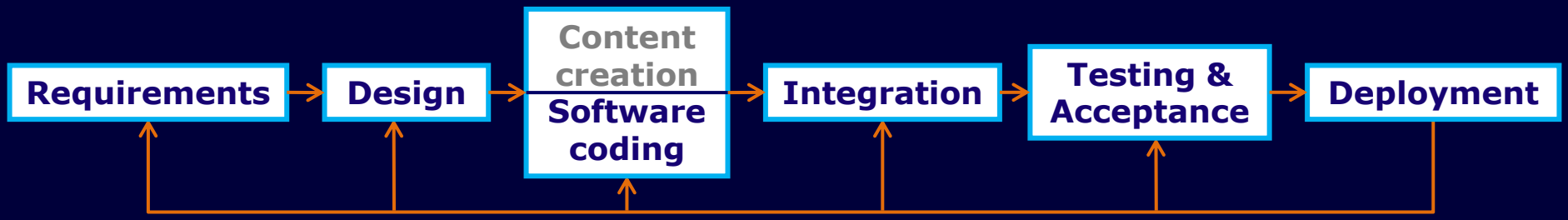
Agile software development (1)



- Flexibility
- Interactivity
- 10 key principles

Agile Alliance

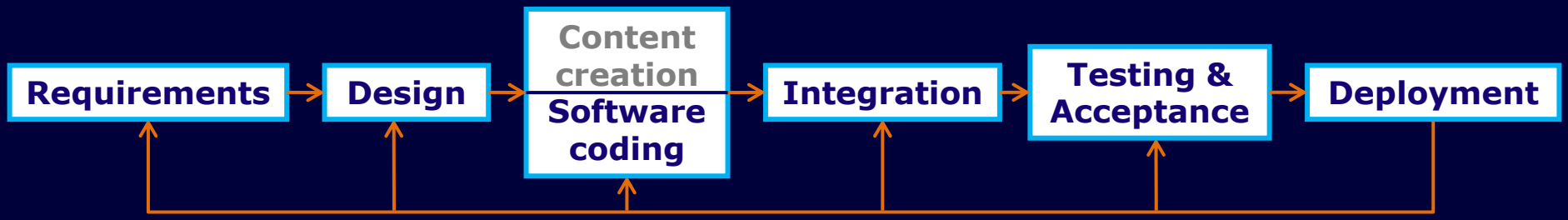
Agile software development (2)



- Actively involve users
- Empower development team to make decisions

Agile Alliance

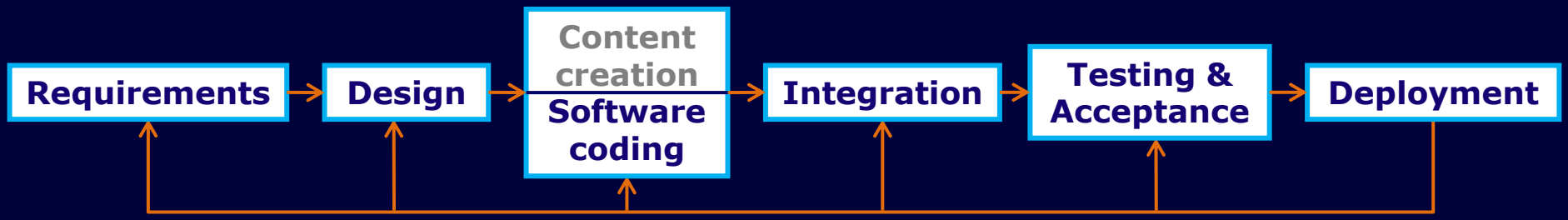
Agile software development (3)



- Allow requirements to evolve but keep timescale fixed
- Capture requirements at highest level of description

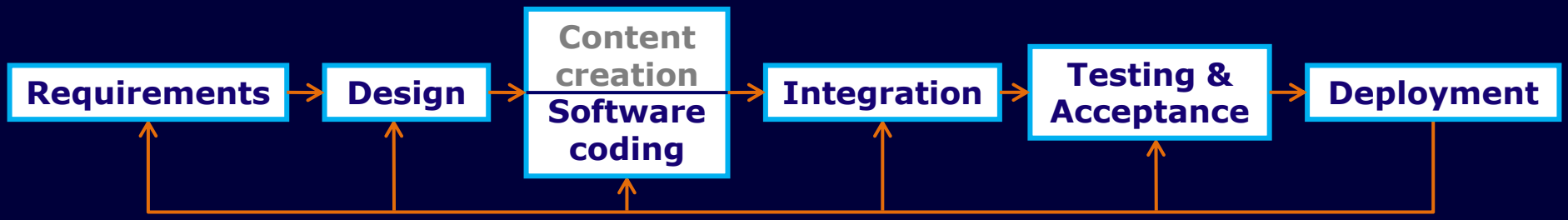
Agile Alliance

Agile software development (4)



- Develop small incremental releases, and iterate
- Make frequent delivery of product (to test…)
- Complete a feature before moving on to the next

Agile software development (5)

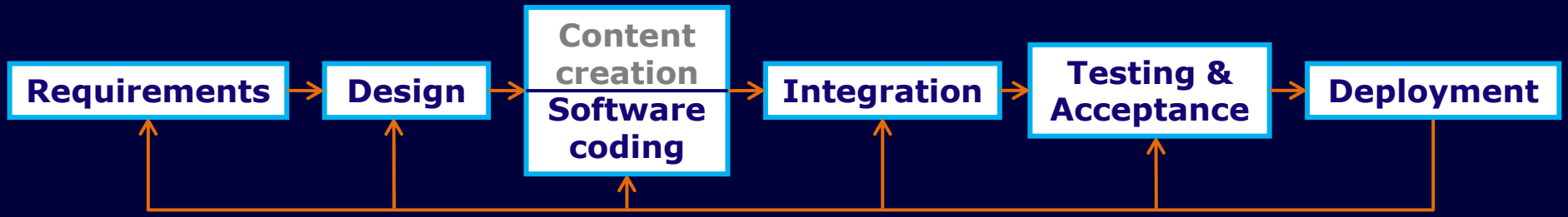


- Apply the “80/20 rule” (Pareto’s principle)
- Integrate testing throughout the project life cycle
- Rely on collaborative approach between stakeholders

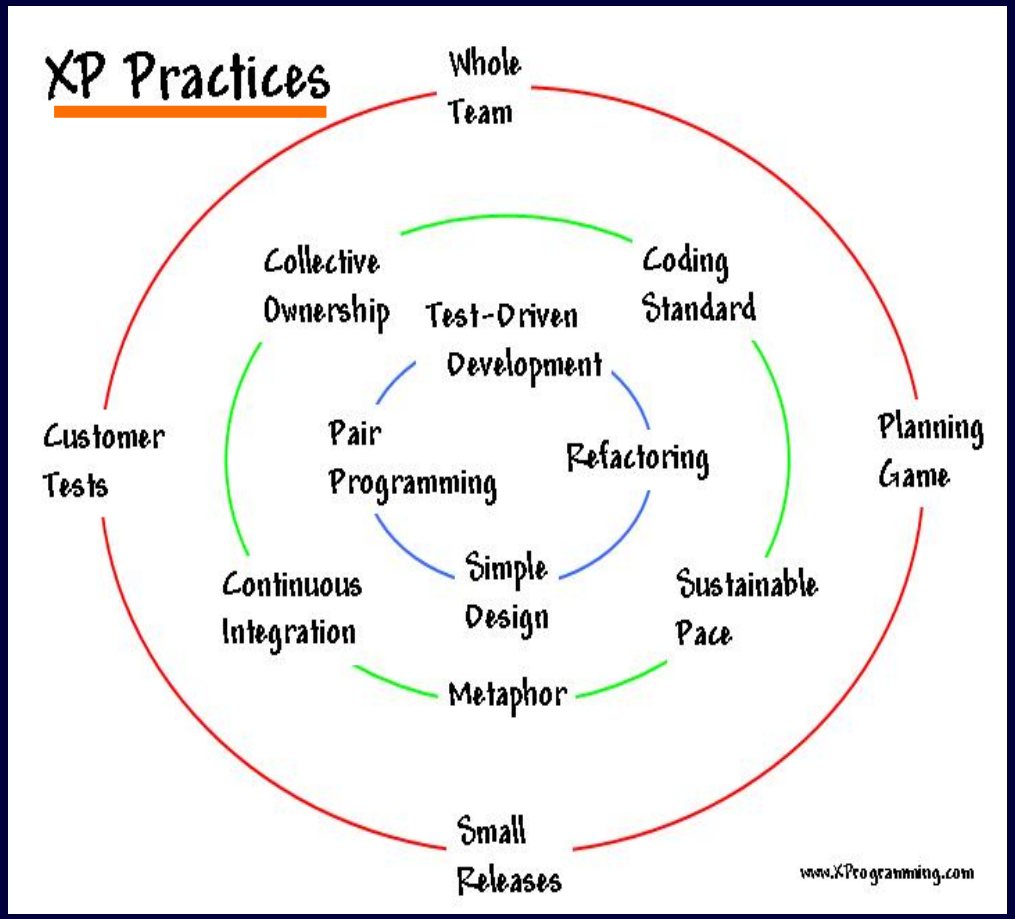
Agile Alliance

Questions?

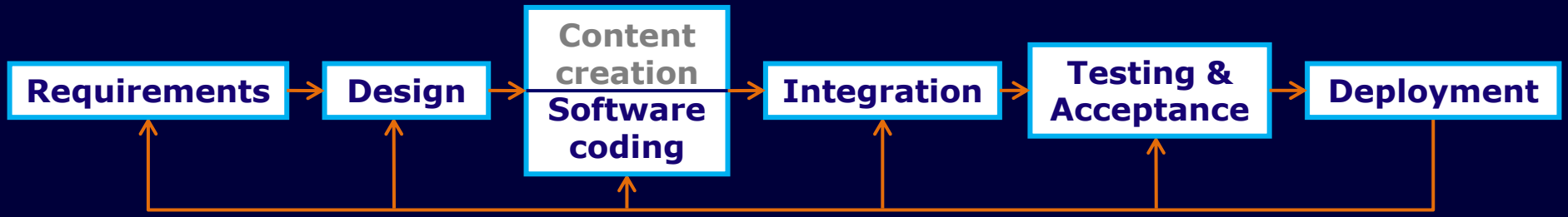
XP: eXtreme Programming (1)



- Communication
- Simplicity
- Feedback
- Courage

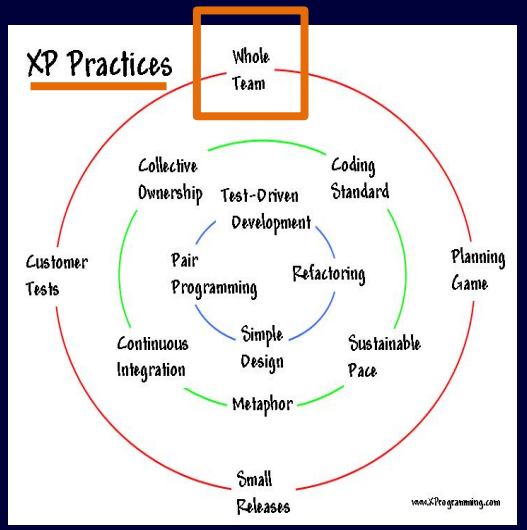


XP: eXtreme Programming (2)

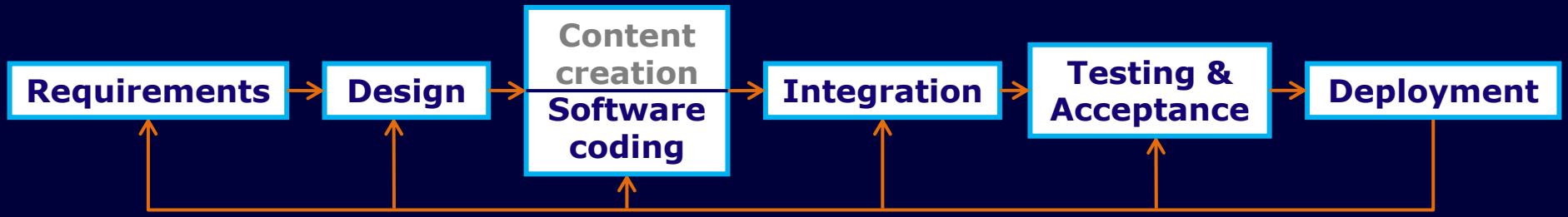


Whole team:

- all contributors to the project form a single team, including at least one business/user representative

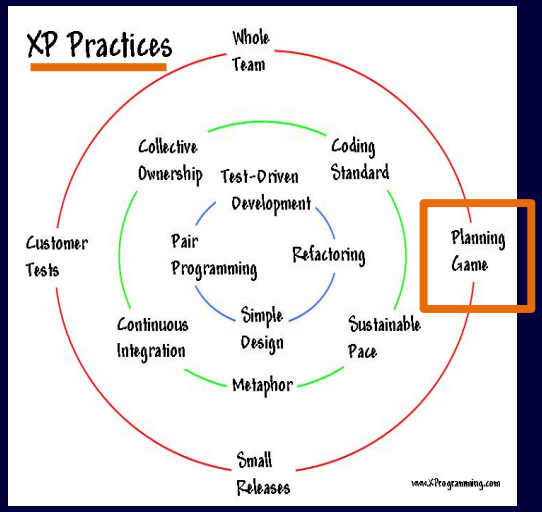


XP: eXtreme Programming (3)

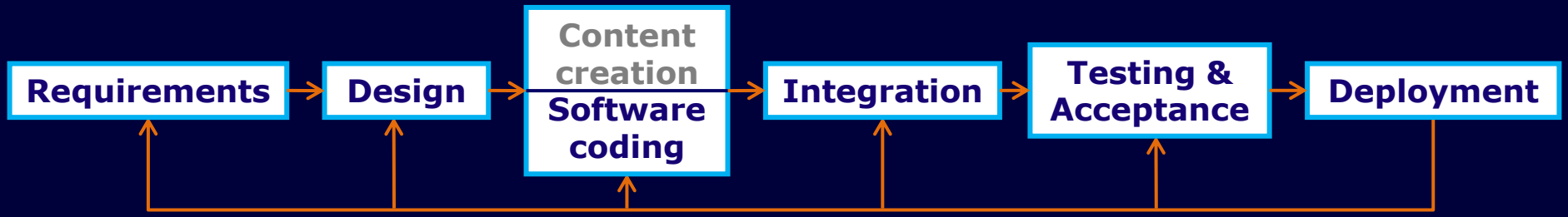


Planning game:

- steer the project rather than exactly predict what needs to be done and how long it will take:
  - ✓ **Release planning:** due dates for deliverables
  - ✓ **Iteration planning:** direction regularly adjusted

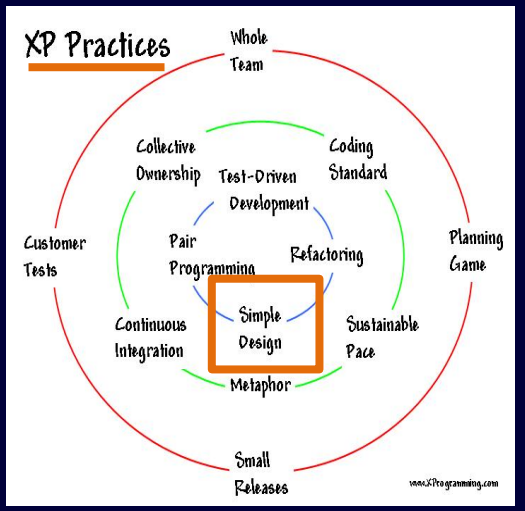


XP: eXtreme Programming (4)

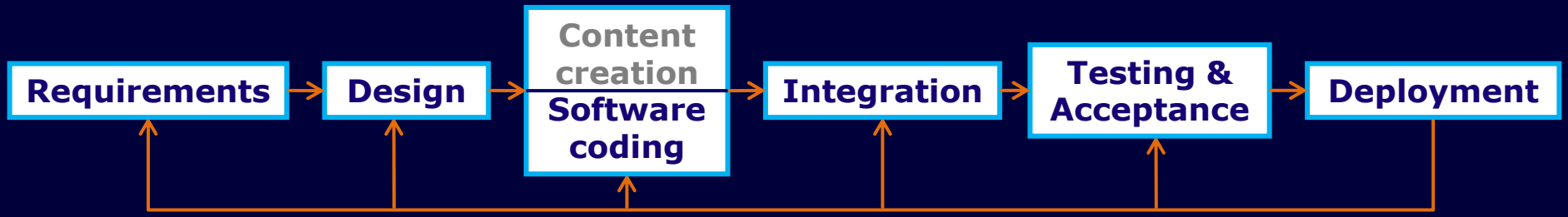


Simple design:

- always match required functionality
- do not waste time on features not really needed

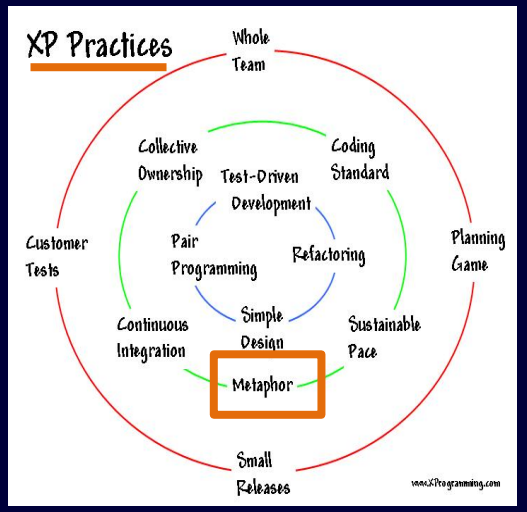


XP: eXtreme Programming (5)

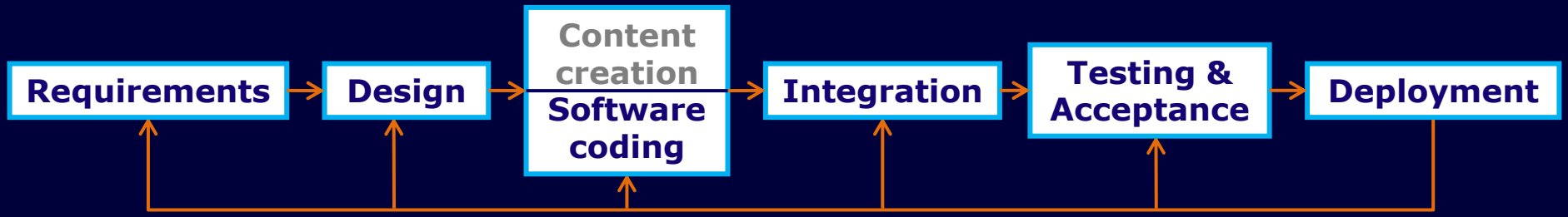


Metaphor:

- describes in very simple and evocative terms how the software should work
- requires agreed-upon vocabulary

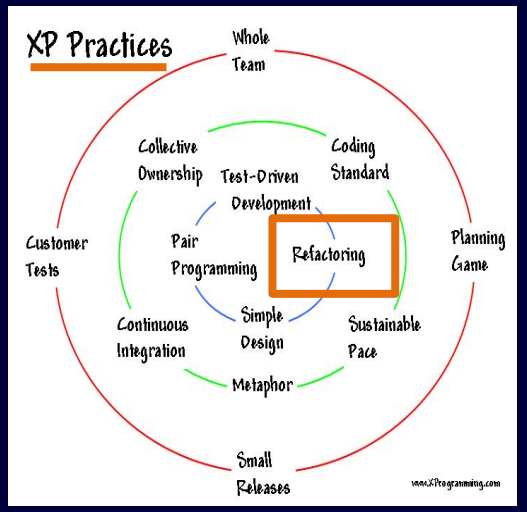


XP: eXtreme Programming (6)

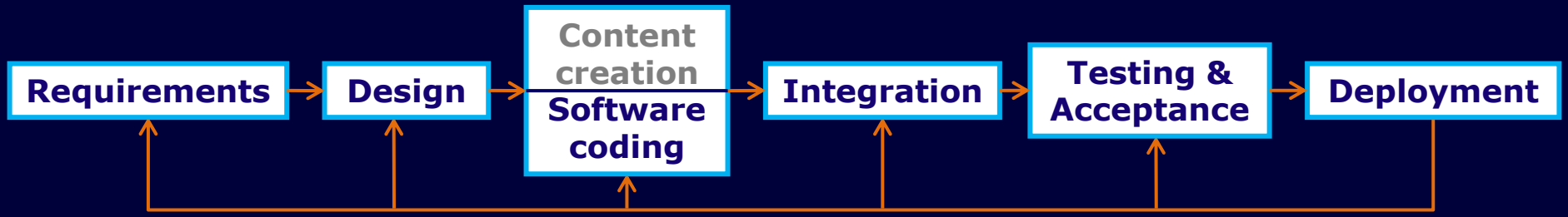


Refactoring:

- continuous process of design improvement
- focuses on avoiding duplication and achieving full “cohesion” of the code

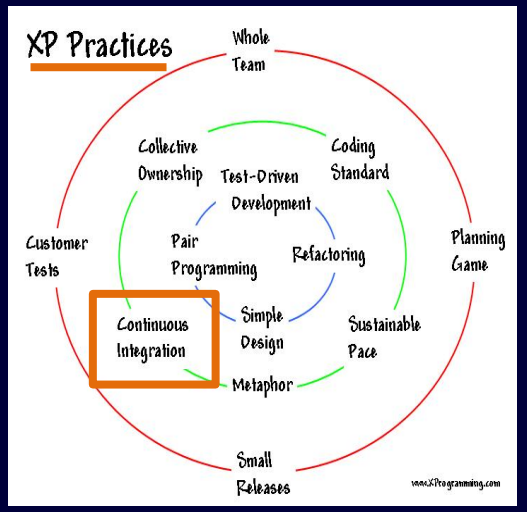


XP: eXtreme Programming (7)

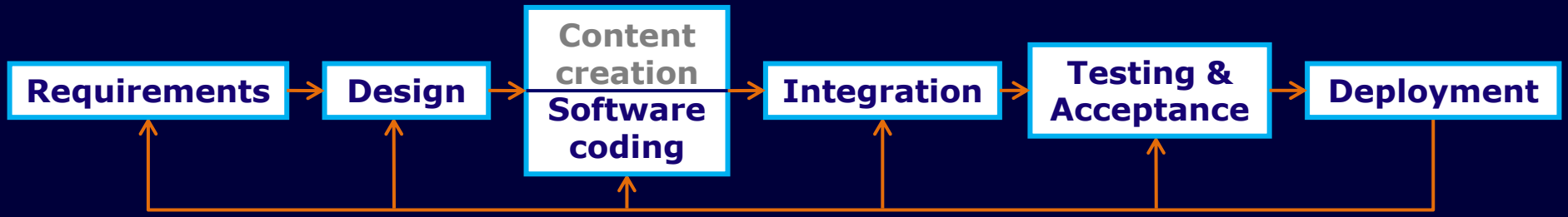


Continuous integration:

- system kept “fully integrated” at all stages of development in order to maintain its cohesion
- system builds produced on a very frequent basis

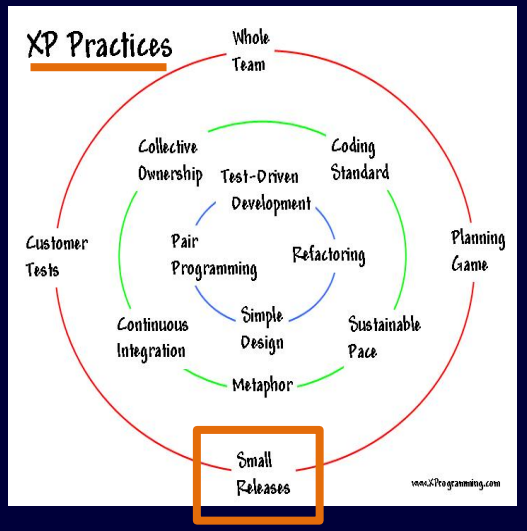


XP: eXtreme Programming (8)

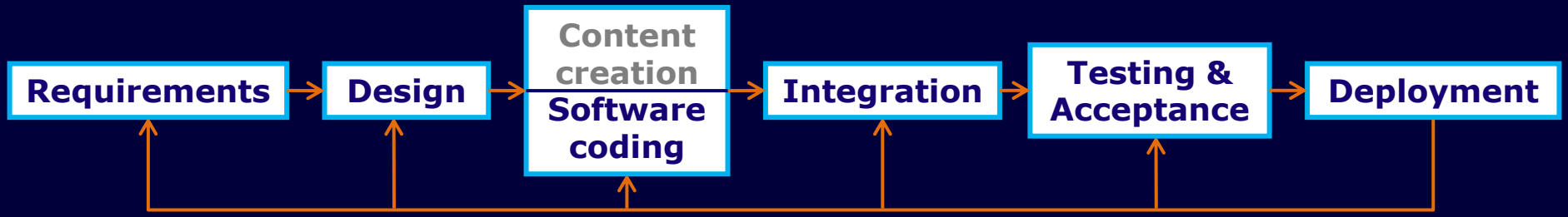


Small releases:

- working version of software delivered to customer after each iteration
- may even be put into operation for maximum feedback

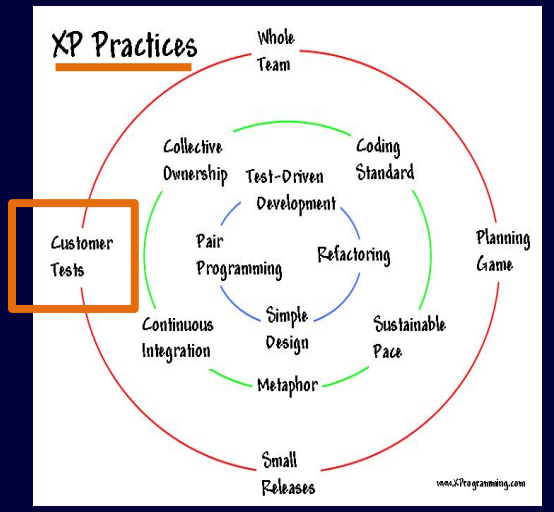


XP: eXtreme Programming (9)

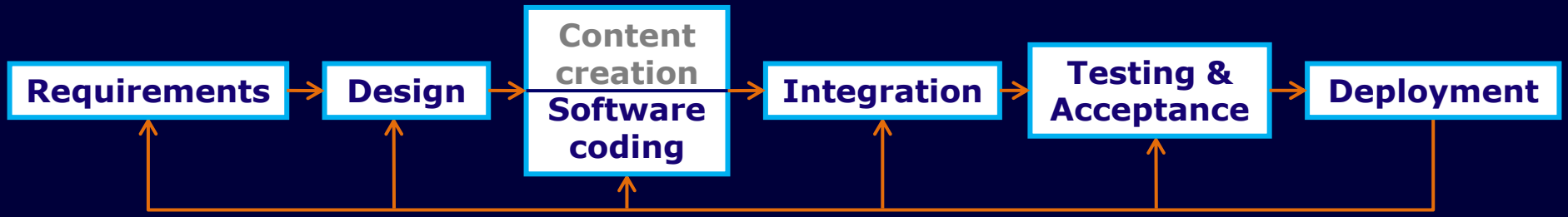


Customer tests:

- performed for each “small release” of software, preferably with automatic acceptance programs

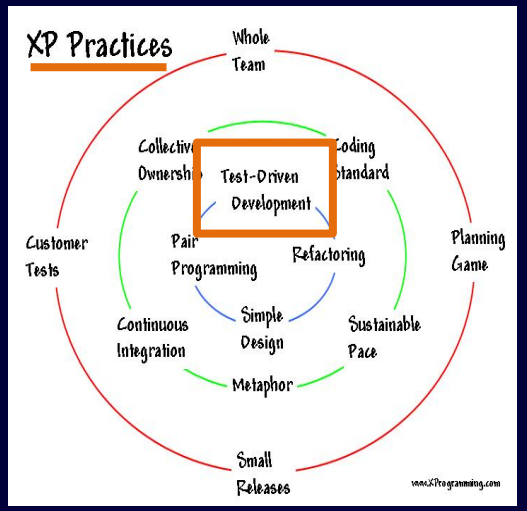


XP: eXtreme Programming (10)

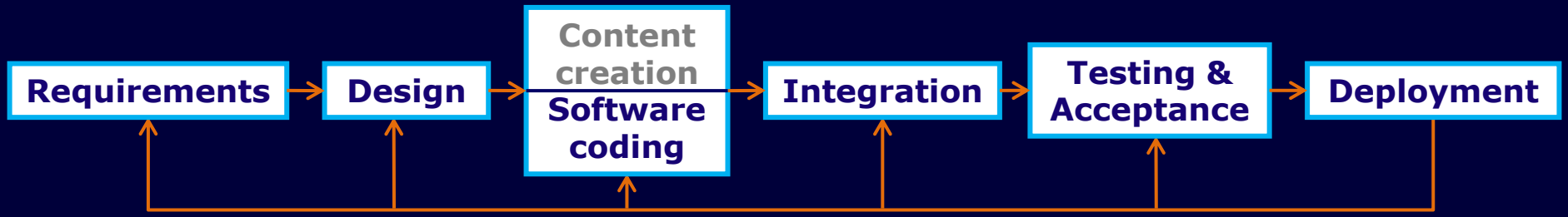


Test-driven development:

- systematic unit tests, with full coverage of features developed
- as the system grows, so does the number of unit tests which need to be run successfully
- feedback from tests drives further development work

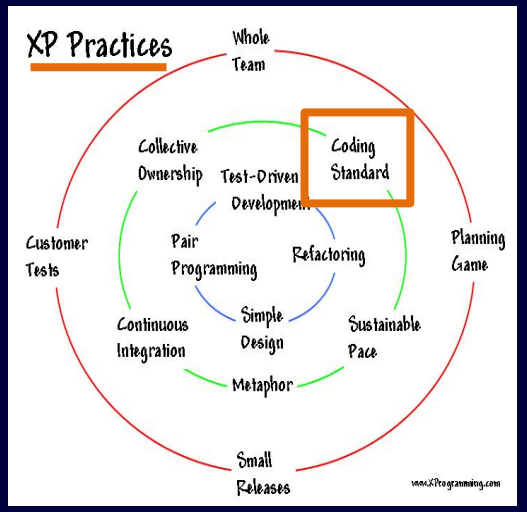


XP: eXtreme Programming (11)

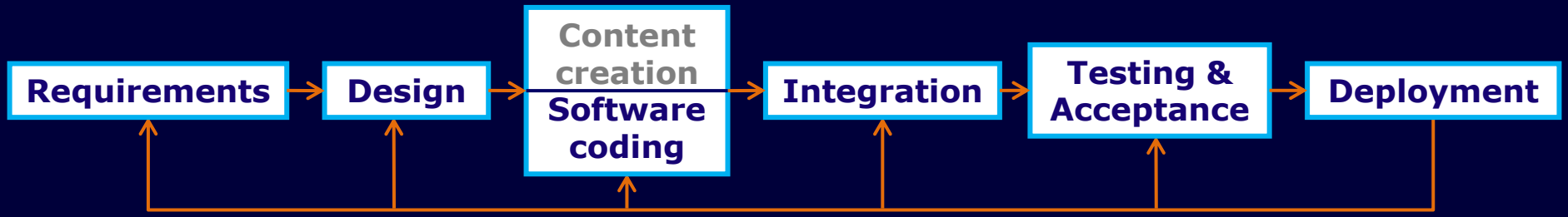


Coding standard:

- code written by any member of the development team complies with a general, unique standard
- ensures cohesion of the system and facilitates code maintenance

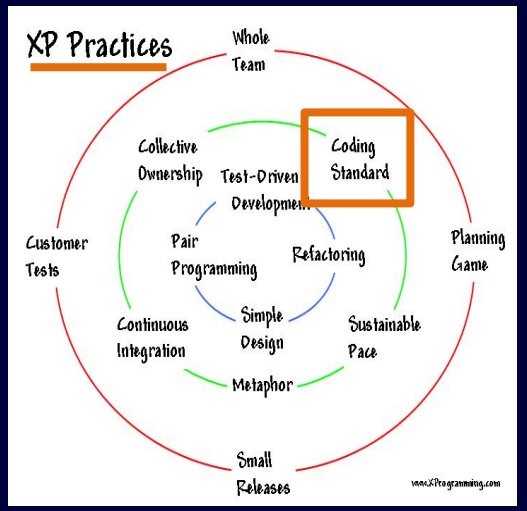


XP: eXtreme Programming (12)

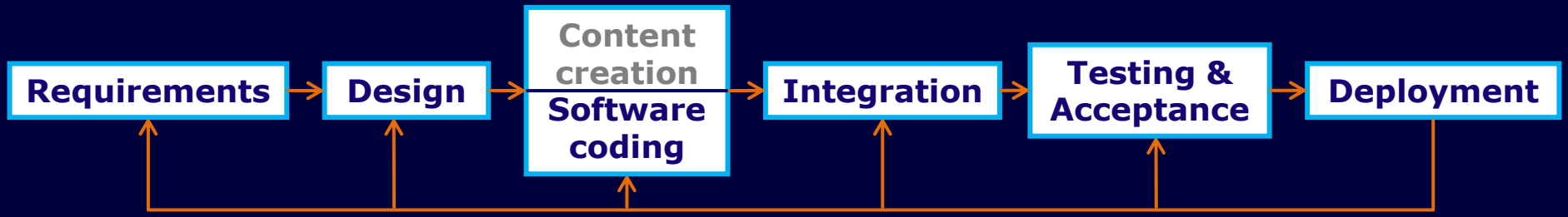


Pair programming:

- each software unit is developed by two programmers working together, to produce better code than would two programmers working singly

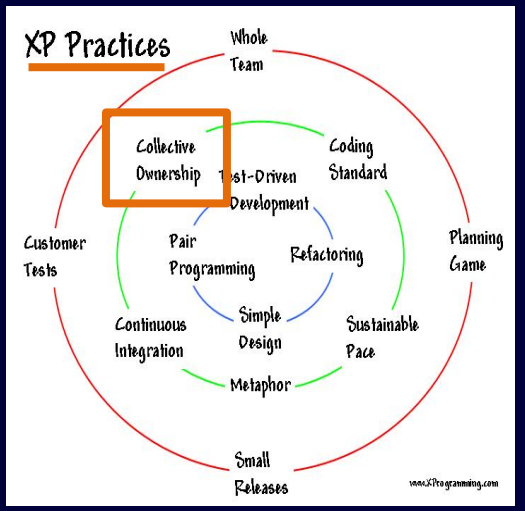


XP: eXtreme Programming (13)

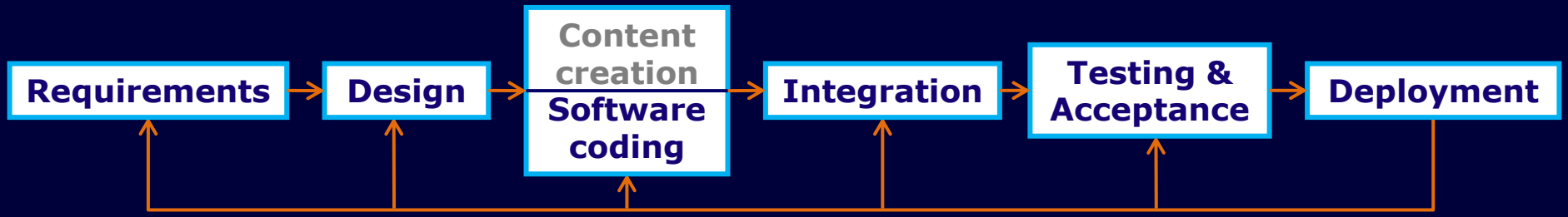


Collective ownership:

- code produced by programmers is owned by all members of the development team
- each person pays attention to code written by others and contributes to improving its quality

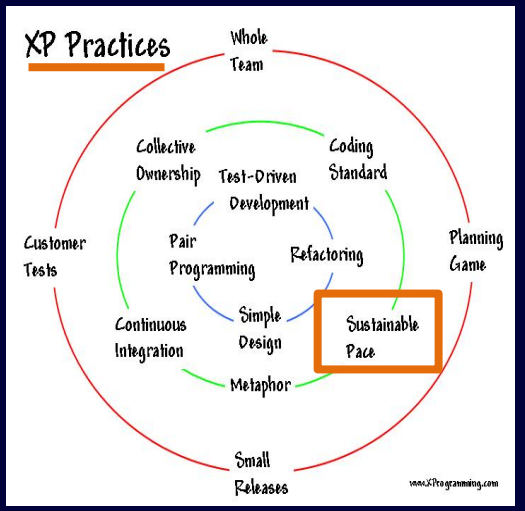


XP: eXtreme Programming (14)



Sustainable pace:

- developers should work with maximum productivity at a sustainable pace, avoiding "burn-out"...
  - ✓ easier said than done!



Questions?