

# Lifecycle Models

What software lifecycle models define

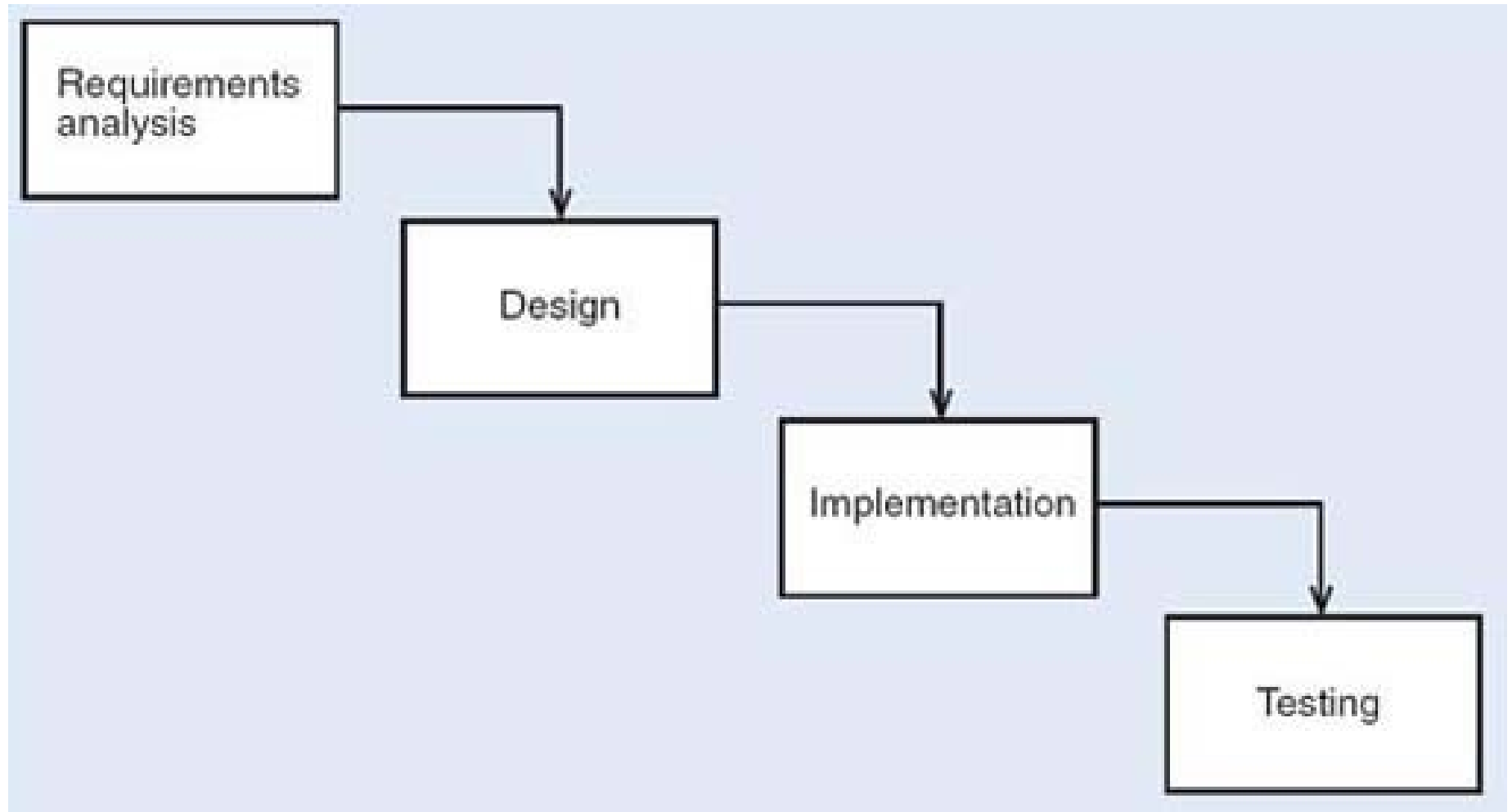
- The economics of the project
- Project risk management
- The nature and scope of the project (size, timing, risks)
- Product maintainability
- Project architecture
- Error detection/removal rates
- Product completeness degree

# Lifecycle Models

Lifecycle models

- Build-and-Fix model
- Incremental model
- Waterfall model
- Rapid prototyping model
- Stabilization and synchronization model
- Object-oriented model
- Spiral model

# Waterfall Model



# Waterfall Model features

- Sequential change of all lifecycle phases
- Software Quality Assurance (SQA) group verifies/tests results after phase (sometimes client side participates)
- Feedback with earlier lifecycle phases

## Benefit

Cost reduction for software correction (due to feedback)

## Drawbacks

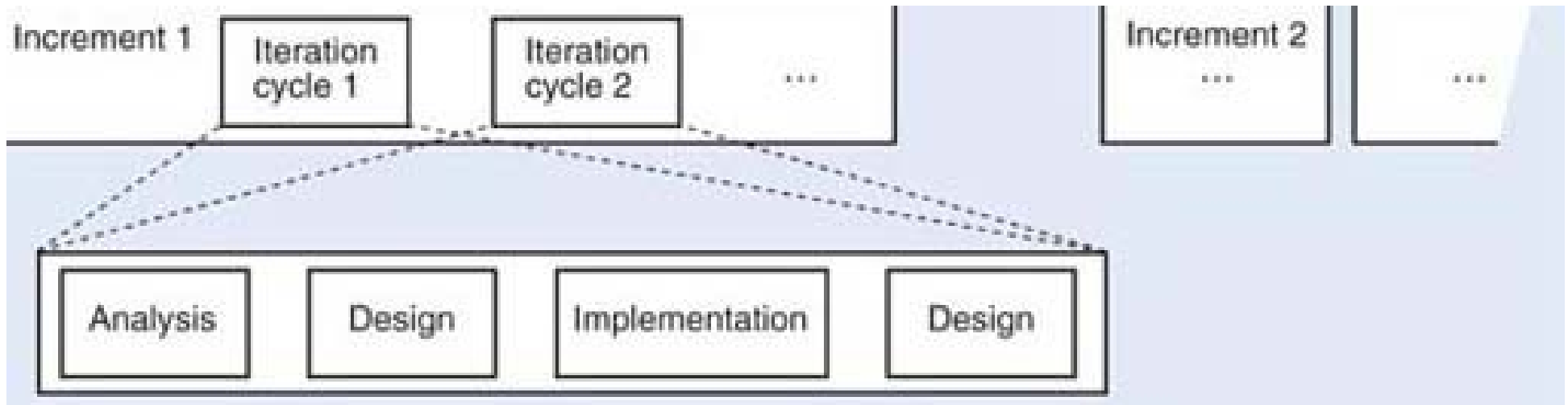
Requires technically «literate» client to create satisfactory specifications

Under high risks, should be combined with rapid prototyping

# Waterfall Model

- Clear project discipline
- Document-driven model
- The software may not meet the customer's requirements
- Changes become complicated: terminated phases get «frozen»
- Does not include iterations and evolution

# Incremental Model



# Incremental Model

## **Benefits**

- The maximal earliest return on investments
- It facilitates the maintainability

## **Disadvantages**

- It requires an open architecture
- It can generate into Build-and-Fix

# Incremental Model

- Product decomposition to sequential releases (each development cycle gives an operational product)
- Operational product at every development step
- Flexible introduction of the new functionality at the client's site
- Easy maintenance due to «straightforward expanding» of the major product modules



# Rapid prototyping Model

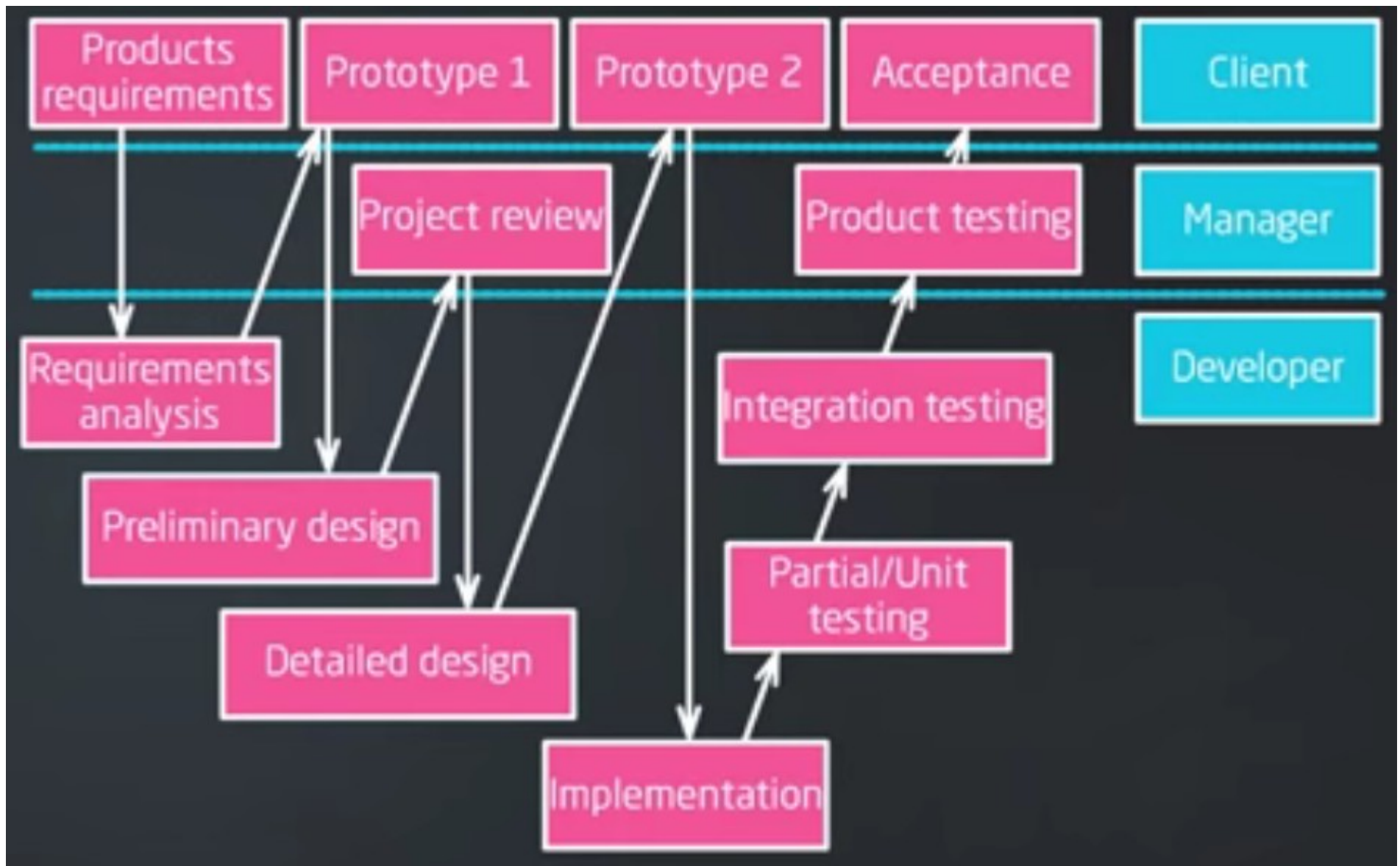
Jim Highsmith offered three life cycle phases:

- Speculation
- Collaboration
- Learning

## **Features**

- Rapid prototype has limited functionality and reliability/performance
- Client has not technical knowledge to discuss requirements
- Requirement analysis and specifications are generally possible before coding and testing

# Rapid prototyping Model



# Rapid prototyping Model

Rapid prototyping should be used in conduction with some other models

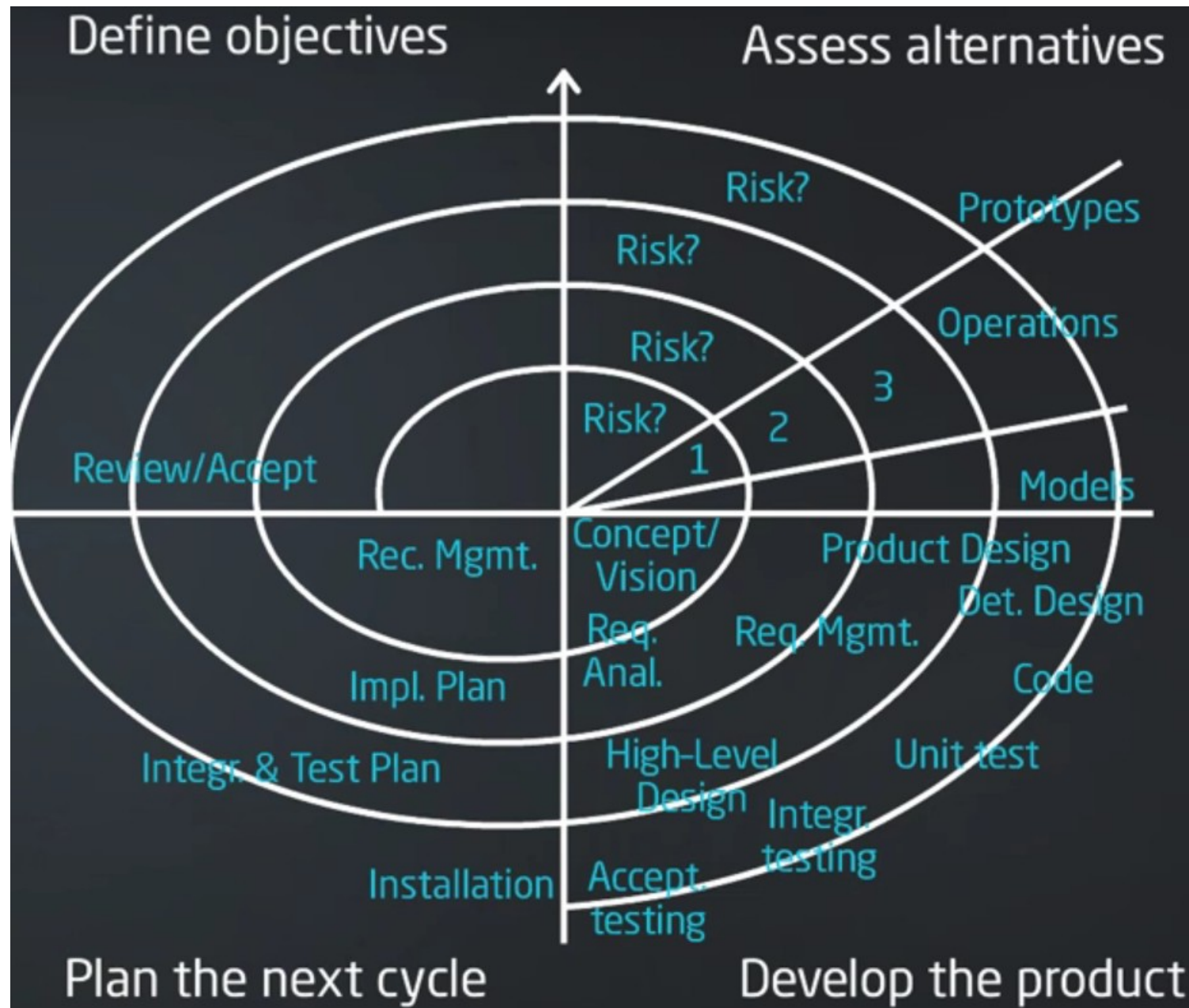
## **Benefit**

- Rapid prototyping ensures compliance with customer requirements

## **Drawback**

- User confusion of prototype and finished system

# Spiral Model



# Spiral Model

## **Features**

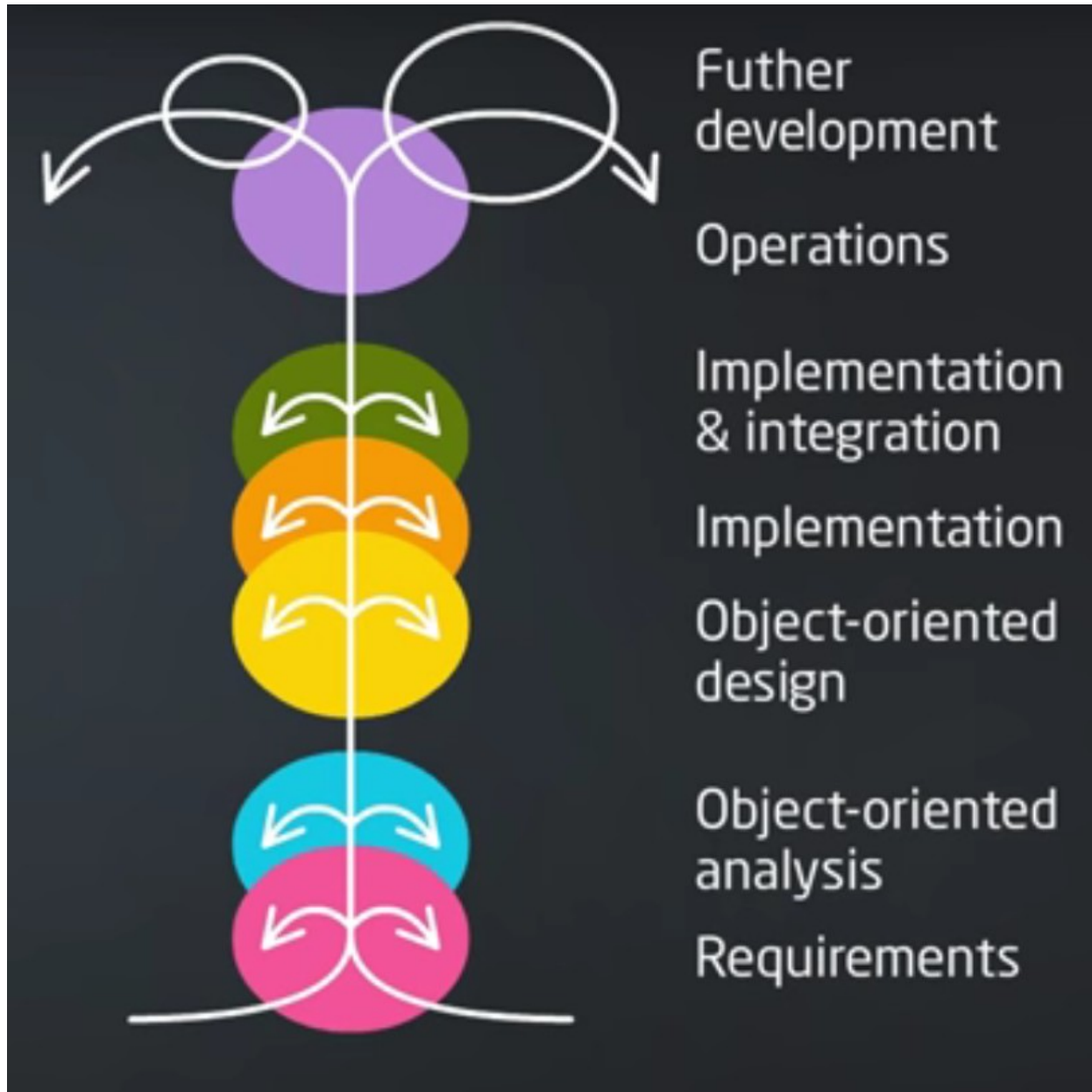
- Rapid prototyping advantages application to the entire lifecycle
- Based on waterfall model and risk analysis
- Risk analyzed at the start of each phase (detection and mitigating of the most serious project risks)
- Project terminated if risks cannot be eliminated
- Several prototyping steps and unlimited iterations number allowed

# Object-Oriented Model

## **Features**

- Intensive interaction between lifecycle phases
- Iterative lifecycle phases change
- Phases interlap
- Object Oriented Design (OOD) usually include Object Oriented Analysis (OOA) phases
- Backtrack to earlier phases is possible

# Object-Oriented Model



# Object-Oriented Model

## **Benefit**

- It provides interaction and parallelism between the phases

## **Disadvantage**

- It can degenerate into CABTAB (Code a bit, test a bit) Build-and-Fix



# Similarities of software lifecycle models

- Include all stages of the software lifecycle (except Build-and-Fix)
- It involves several iterations of project development
- Software lifecycle stage is clearly distinguishable (except OO)
- Related to the design methodology
- Require a high organizational maturity and project development team discipline (can degenerate into CABTAB)



Lifecycle Model	Advantages	Disadvantages
Build-and-Fix (Code-and-Fix)	Good for small projects that do not require maintenance	Absolutely not suitable for non-trivial projects
Waterfall	Clear project discipline, document-driven	The software may not meet the customer's requirements
Rapid prototyping	It ensures compliance with customer requirements software	It is tempting to reuse code to be re-implemented
Incremental	The maximal return on investments earlier; It facilitates the maintainability	It requires an open architecture; can degenerate into Build-and-fix
Synchronization and Stabilization	Meets future needs of the customer; provides integration component	Not widely used outside of Microsoft
Spiral	It combines the features of all the above models	Suitable only for large-scale domestic projects; developers must own risk management
Object-oriented	It provides iteration within phases and the parallelism between the phases	It can degenerate into CABTAB

# References

- Sommerville I. Software Engineering (9 ed.), Addison Wesley, 2011, 790 pp.
- Schach S.R. Object Oriented and Classical Software (8 ed.), McGraw-Hill, 2011, 688 pp.
- McConnell S. Code Complete. Microsoft Press, 1993
- Zykov S.V. Crisis Management for Software Development and Knowledge Transfer, Springer, 2016