software studio

software development processes

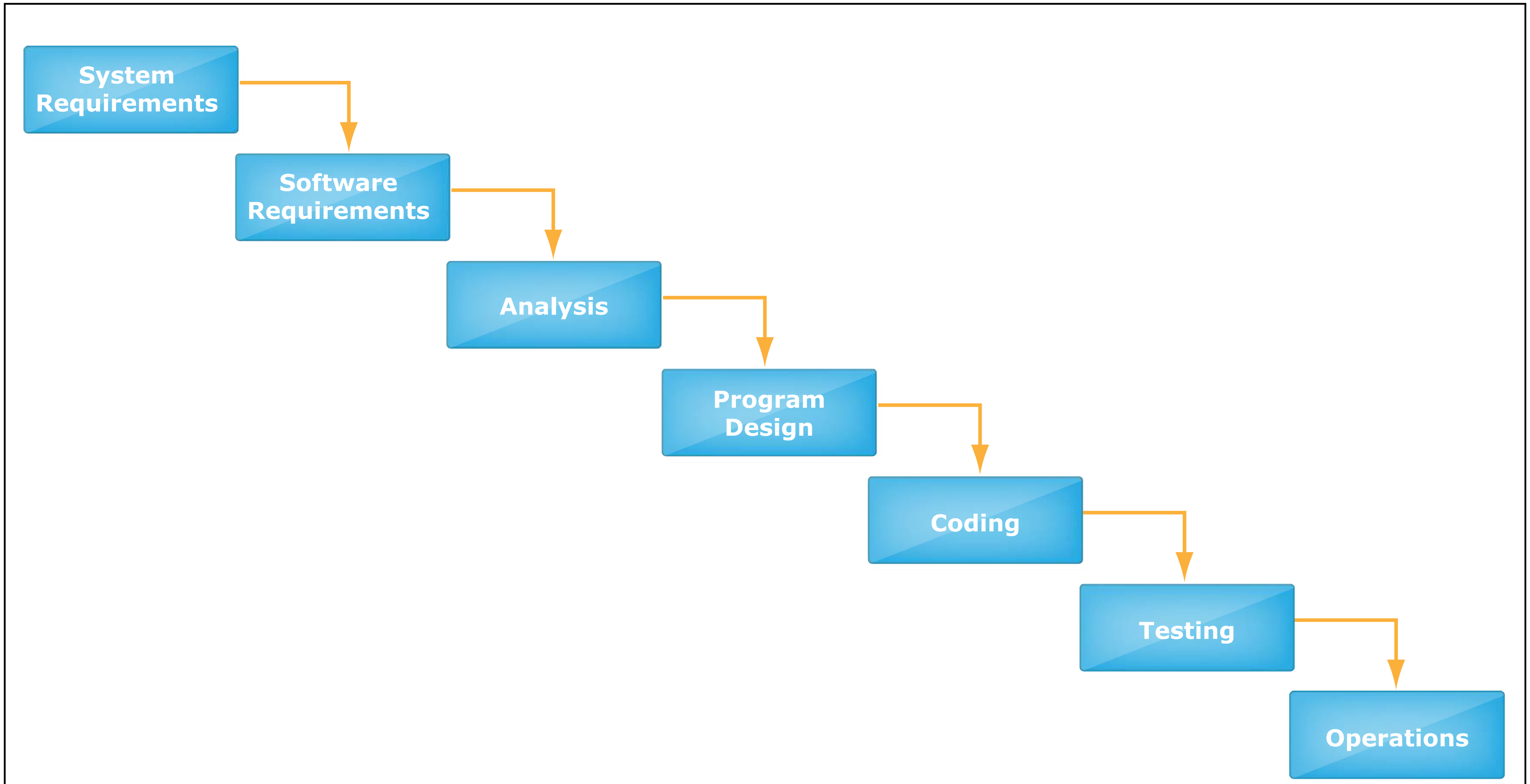
Daniel Jackson

One of the planning documents for software research revealed --in a parenthetical remark only-- an unchallenged tacit assumption by referring to "the tradeoff between cost and quality". Now in all sorts of mechnical engineering it may make sense to talk about "the tradeoff between cost and quality", in software development this is absolute nonsense, because poor quality is the major contributor to the soaring costs of software development.

—Dijkstra, EWD690

Quotation from Fred Brook's *The Mythical Man Month*, 1975 removed due to copyright restrictions.

the waterfall model, 1970

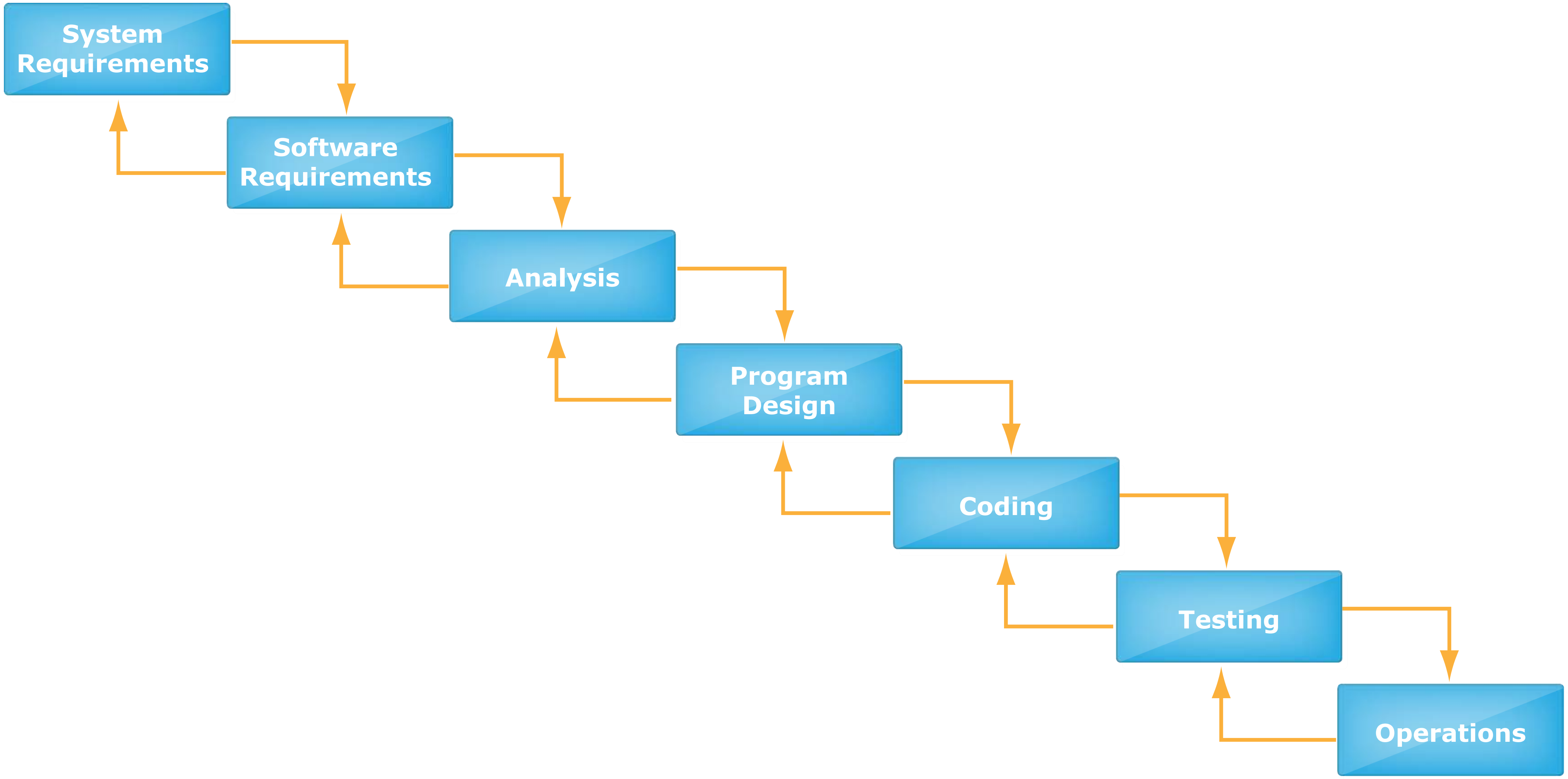


Ideal waterfall model.

Image by MIT OpenCourseWare.

from: Winston Royce, “Managing the Development of Large Software Systems”, Proceedings of IEEE WESCON 26 (August): 1–9, 1970.

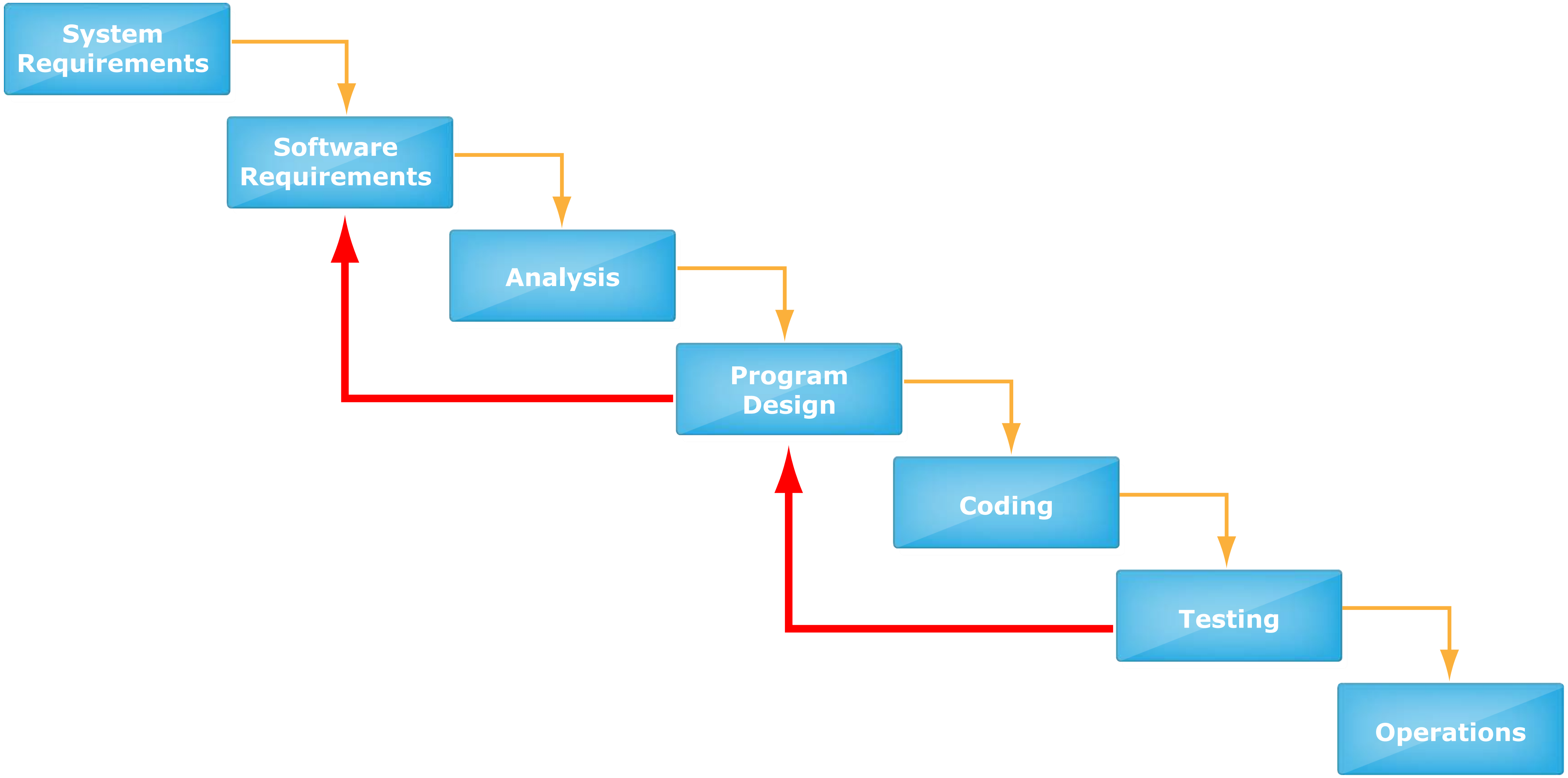
what Royce actually said



Waterfall model with iterative interactions restricted to successive steps.

Image by MIT OpenCourseWare.

what happens in practice



Waterfall model in practice, iterations are not restricted to successive steps.

Image by MIT OpenCourseWare.

Royce's fixes

program design comes first

- do some design between requirements and analysis phases

document the design

- how much? "my own view is quite a lot"

do it twice

- "If the computer program in question is being developed for the first time, arrange matters so that the version finally delivered to the customer for operational deployment is actually the second version insofar as critical design/operations areas are concerned"

plan, control and monitor testing

- with a separate testing team

involve the customer

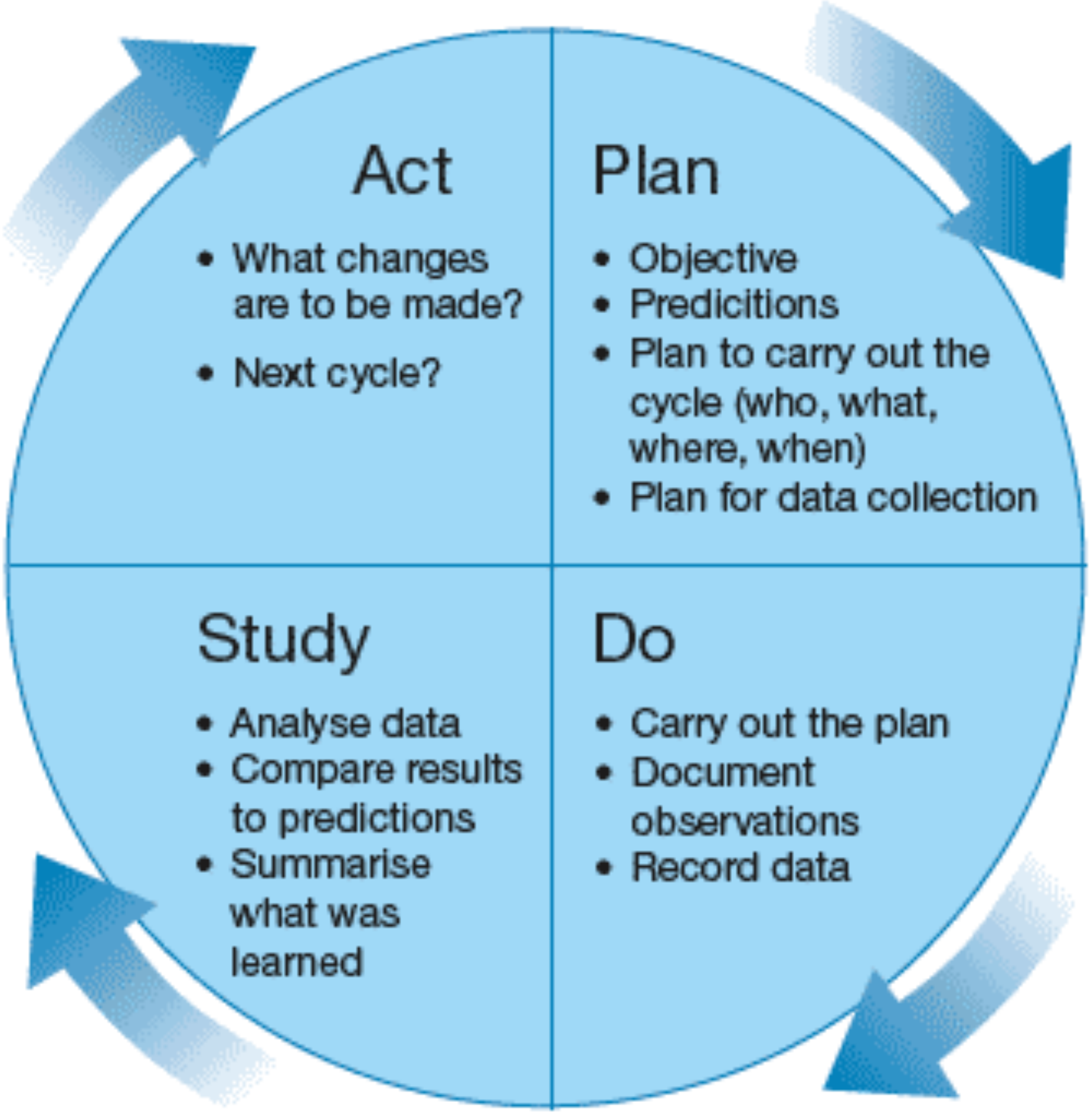
- "in a formal way, committed... at earlier points before final delivery"

spiral model, 1986

Diagram of the spiral model removed due to copyright restrictions.

Reference: Figure 1 from Boehm B. "A Spiral Model of Software Development and Enhancement." _$&0 6,*62)7 6RIWZDUH (QJLQHHULQJ 1RWHV_ 11, no. 4 (1986): 14-24.

origins of iterative approaches


Act

- What changes are to be made?
- Next cycle?

Plan

- Objective
- Predicitions
- Plan to carry out the cycle (who, what, where, when)
- Plan for data collection

Study

- Analyse data
- Compare results to predictions
- Summarise what was learned

Do

- Carry out the plan
- Document observations
- Record data

plan-do-study-act

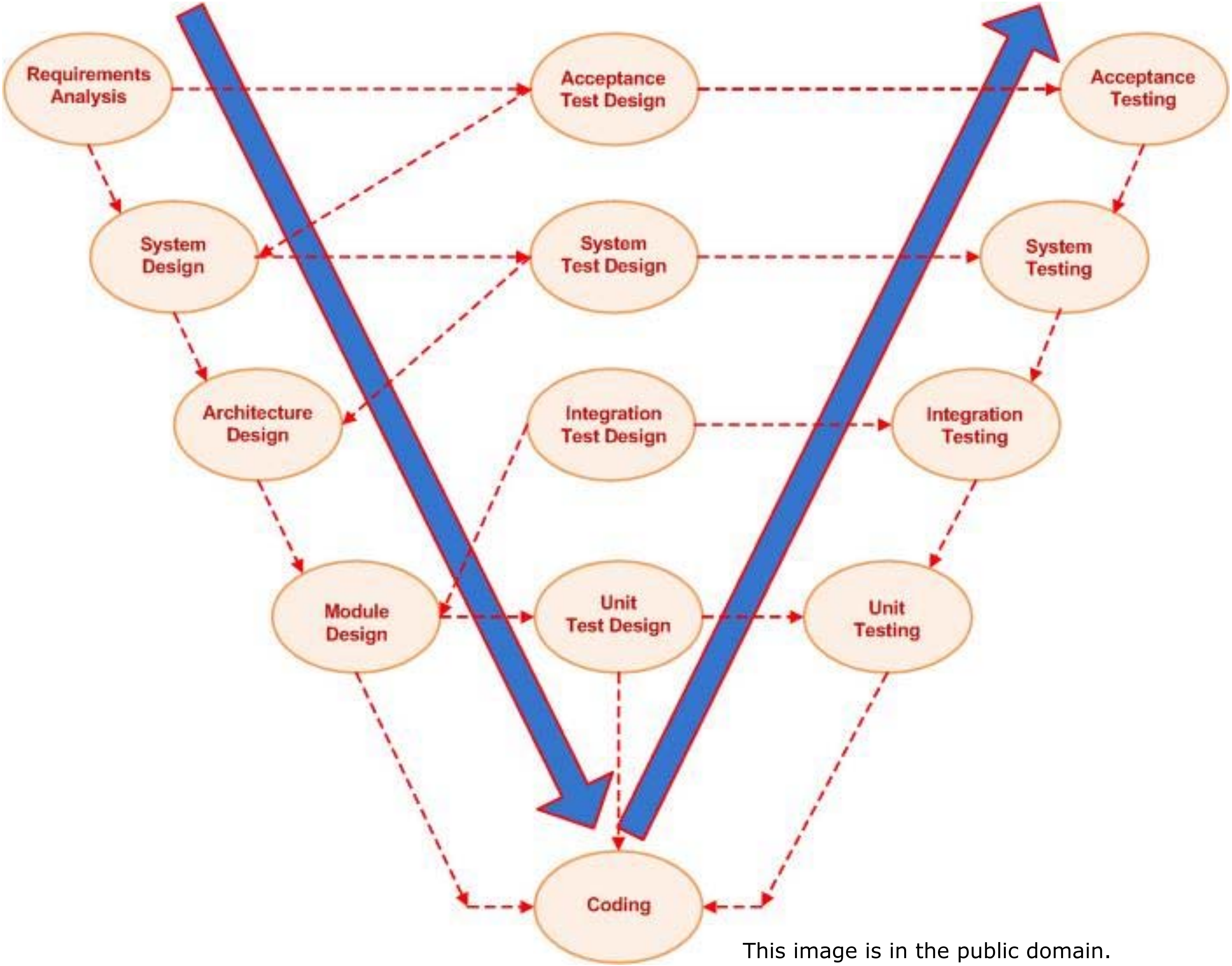
- Shewart, 1930s
- Deming, 1940s

Project Mercury

- NASA, 1960s
- half-day iterations
- tests before each iter
- became IBM Federal Systems Division

See: Craig Larman, Victor R. Basili (June 2003). "Iterative and Incremental Development: A Brief History". IEEE Computer(IEEE Computer Society) 36 (6): 47–56.

V model



This image is in the public domain.

tests developed in early phases, applied in later phases

extreme programming

Kent Beck, 1999

- take best practices to “extreme” levels
- developed during C3 project with Ron Jeffries

a sample of XP practices

- test first: acceptance and unit tests
- continuous integration
- pair programming
- repeated refactoring

Chrysler’s C3 payroll system

- started in 1996, cancelled in 2000
- implemented in Smalltalk
- running payroll took 1000 hours initially
- Chrysler said they abandoned XP after this

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

agile approaches

agile manifesto (2001)

- an articulation of common practices
- a reaction to traditional notions

rejected notions

- upfront design (“BDUF”)
- written documentation (“ceremonial”)
- planning for future modifications

key practices like XP

- continuous integration, test first, refactoring
- features added incrementally (“sprints” and “scrums”)

6.170 Software Studio
Spring 2013