Overview & Why Software Project Management?

Jamal Madni

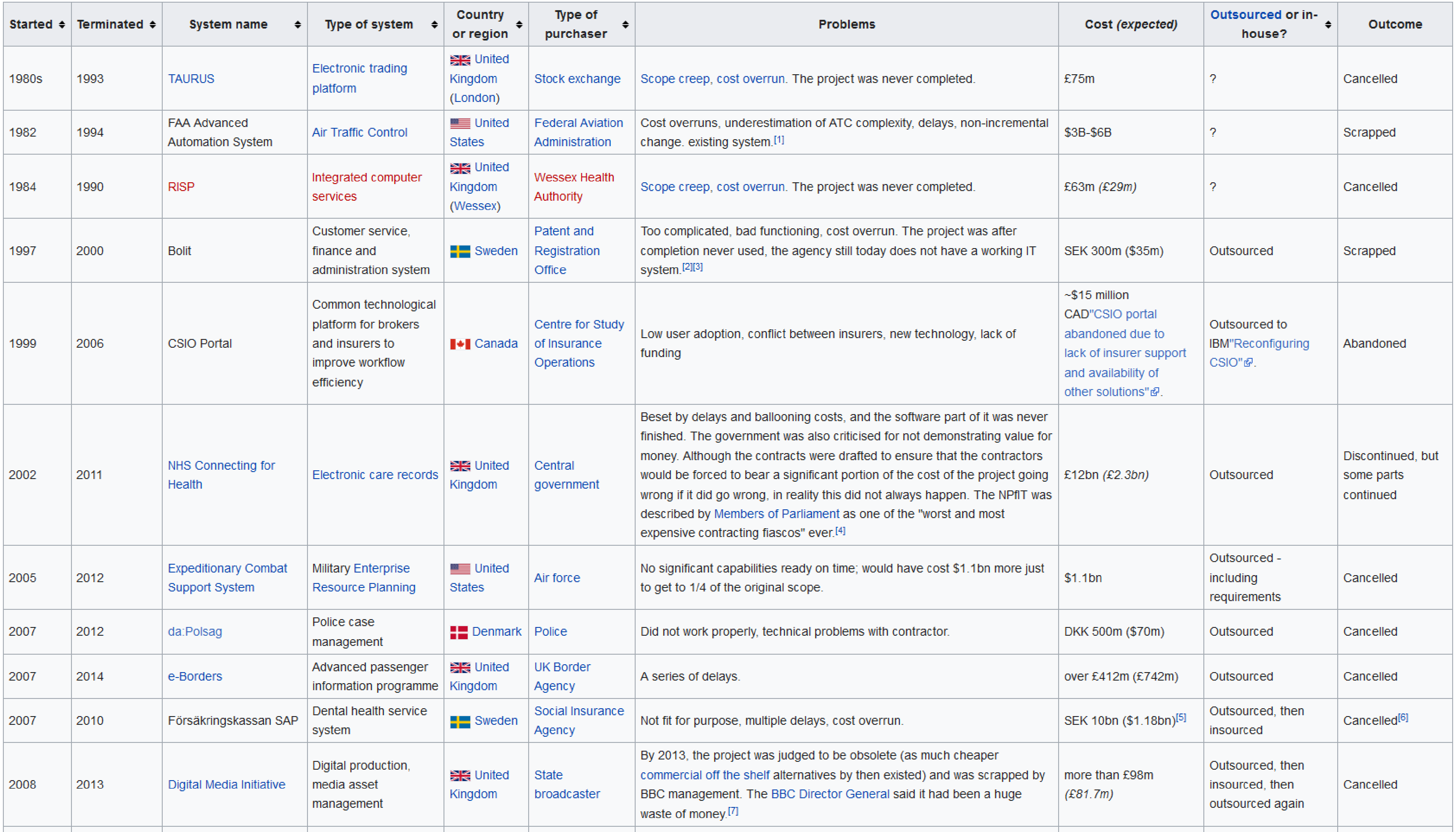
CECS 445

Lecture 1: January 26th, 2021

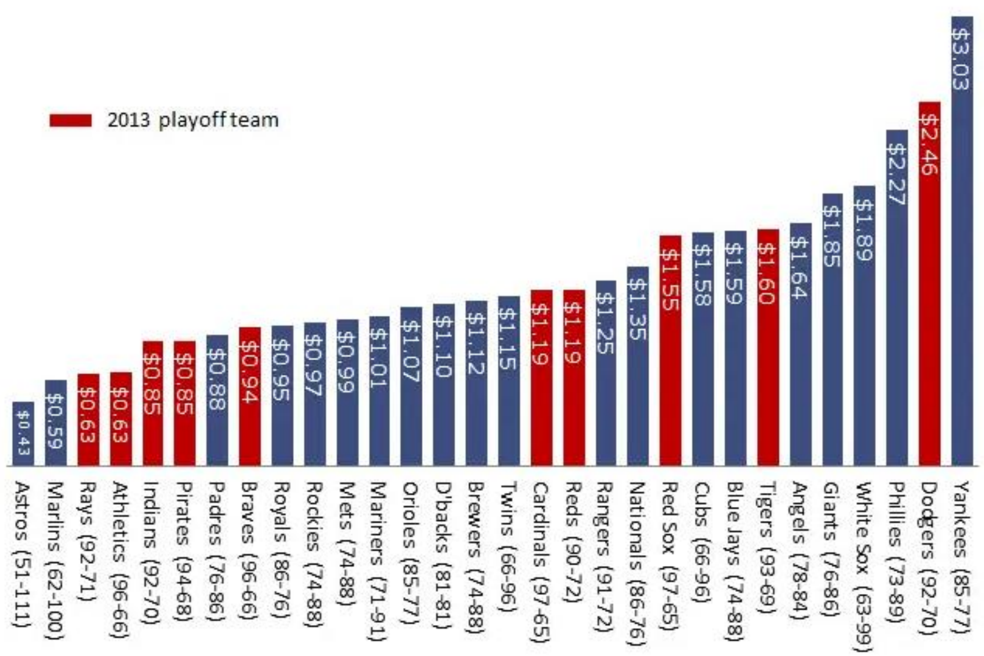
CALIFORNIA STATE UNIVERSITY

1949

LONG BEACH

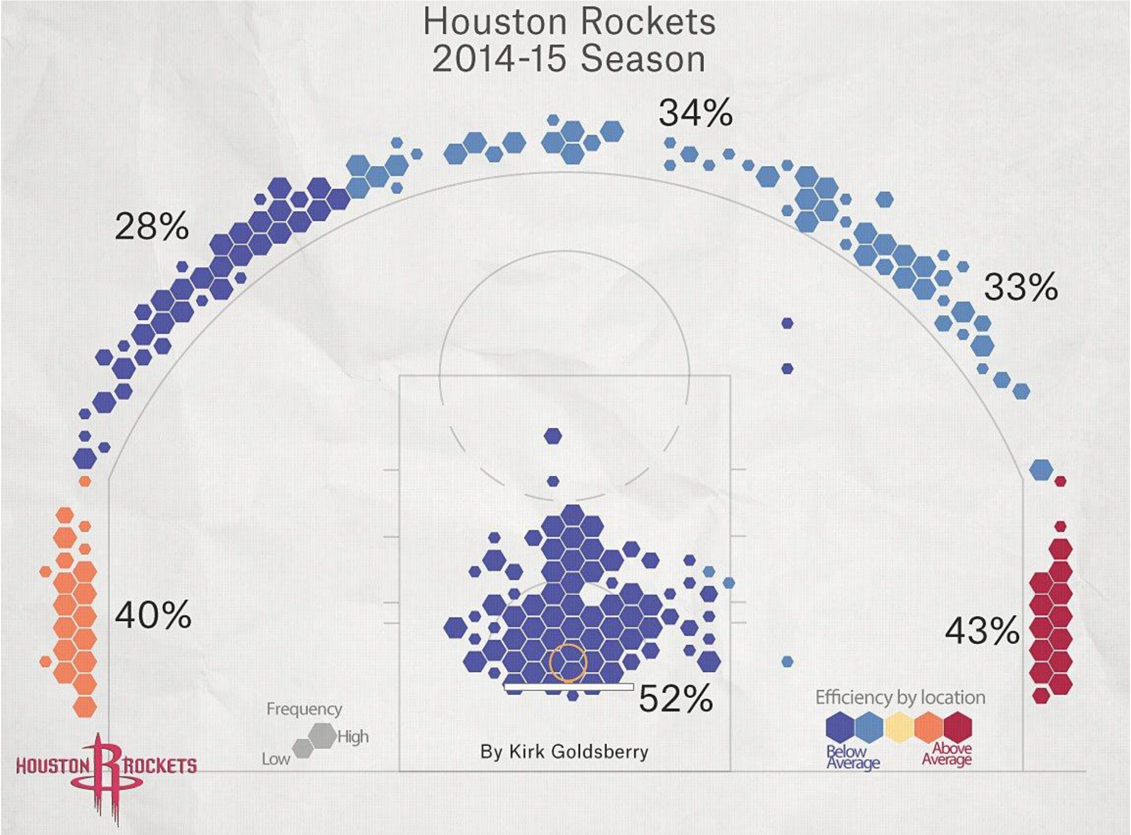


Started	Terminated	System name	Type of system	Country or region	Type of purchaser	Problems	Cost *(expected)*	Outsourced or in-house?	Outcome
1980s	1993	TAURUS	Electronic trading platform	United Kingdom (London)	Stock exchange	Scope creep, cost overrun. The project was never completed.	£75m	?	Cancelled
1982	1994	FAA Advanced Automation System	Air Traffic Control	United States	Federal Aviation Administration	Cost overruns, underestimation of ATC complexity, delays, non-incremental change. existing system.[1]	$3B-$6B	?	Scrapped
1984	1990	RISP	Integrated computer services	United Kingdom (Wessex)	Wessex Health Authority	Scope creep, cost overrun. The project was never completed.	£63m *(£29m)*	?	Cancelled
1997	2000	Bolit	Customer service, finance and administration system	Sweden	Patent and Registration Office	Too complicated, bad functioning, cost overrun. The project was after completion never used, the agency still today does not have a working IT system.[2][3]	SEK 300m ($35m)	Outsourced	Scrapped
1999	2006	CSIO Portal	Common technological platform for brokers and insurers to improve workflow efficiency	Canada	Centre for Study of Insurance Operations	Low user adoption, conflict between insurers, new technology, lack of funding	~$15 million CAD"CSIO portal abandoned due to lack of insurer support and availability of other solutions".	Outsourced to IBM"Reconfiguring CSIO".	Abandoned
2002	2011	NHS Connecting for Health	Electronic care records	United Kingdom	Central government	Beset by delays and ballooning costs, and the software part of it was never finished. The government was also criticised for not demonstrating value for money. Although the contracts were drafted to ensure that the contractors would be forced to bear a significant portion of the cost of the project going wrong if it did go wrong, in reality this did not always happen. The NPfIT was described by Members of Parliament as one of the "worst and most expensive contracting fiascos" ever.[4]	£12bn *(£2.3bn)*	Outsourced	Discontinued, but some parts continued
2005	2012	Expeditionary Combat Support System	Military Enterprise Resource Planning	United States	Air force	No significant capabilities ready on time; would have cost $1.1bn more just to get to 1/4 of the original scope.	$1.1bn	Outsourced - including requirements	Cancelled
2007	2012	da:Polsag	Police case management	Denmark	Police	Did not work properly, technical problems with contractor.	DKK 500m ($70m)	Outsourced	Cancelled
2007	2014	e-Borders	Advanced passenger information programme	United Kingdom	UK Border Agency	A series of delays.	over £412m (£742m)	Outsourced	Cancelled
2007	2010	Försäkringskassan SAP	Dental health service system	Sweden	Social Insurance Agency	Not fit for purpose, multiple delays, cost overrun.	SEK 10bn ($1.18bn)[5]	Outsourced, then insourced	Cancelled[6]
2008	2013	Digital Media Initiative	Digital production, media asset management	United Kingdom	State broadcaster	By 2013, the project was judged to be obsolete (as much cheaper commercial off the shelf alternatives by then existed) and was scrapped by BBC management. The BBC Director General said it had been a huge waste of money.[7]	more than £98m *(£81.7m)*	Outsourced, then insourced, then outsourced again	Cancelled



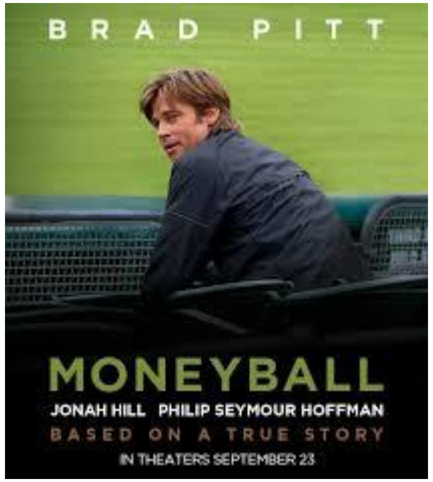
MLB 2013 Cost per Win (in millions)

Business Insider

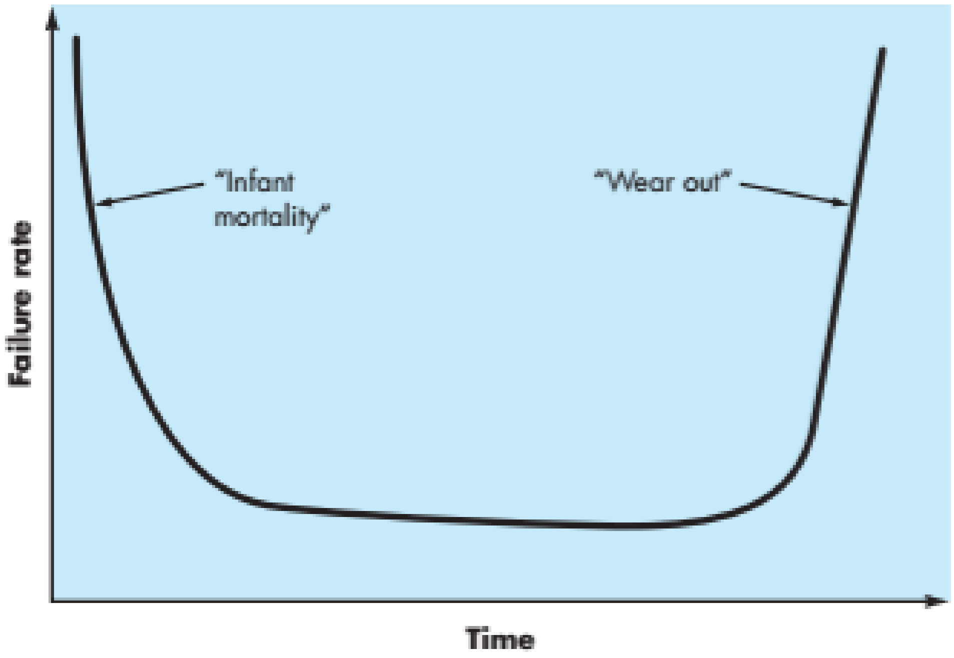


Houston Rockets
2014-15 Season

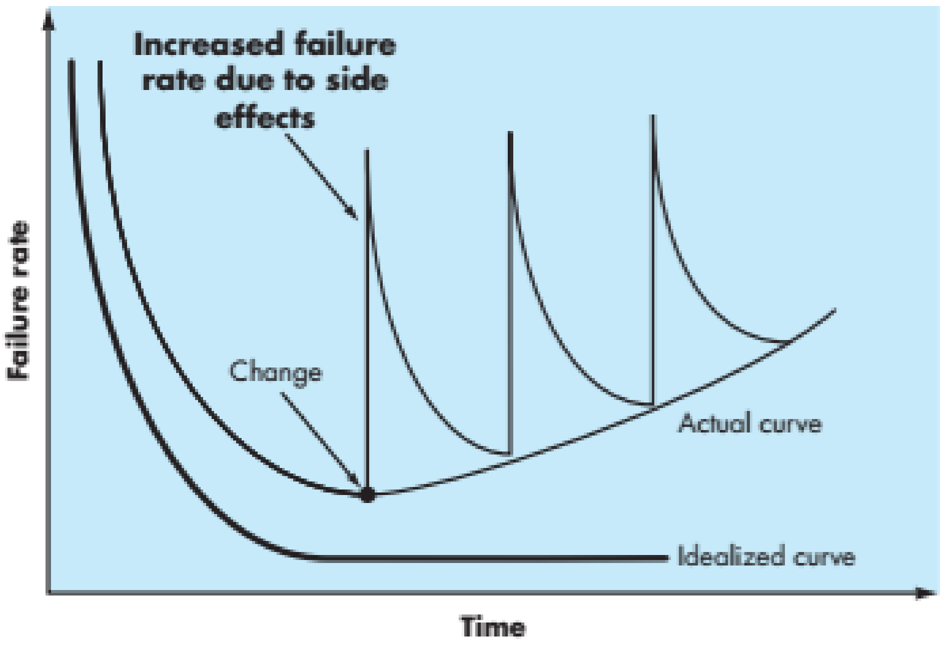
By Kirk Goldsberry

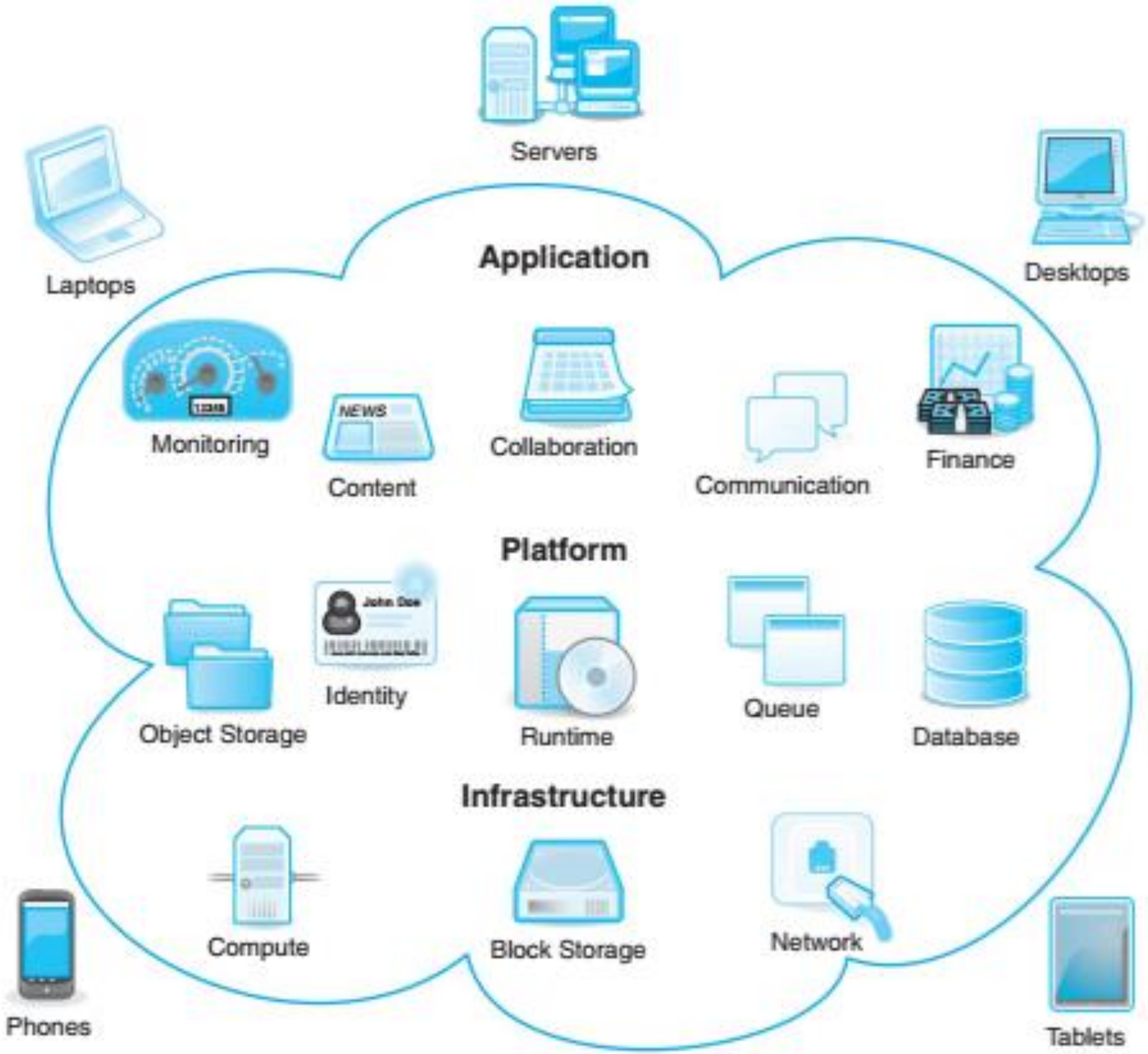


Hardware



Software





Cloud Computing

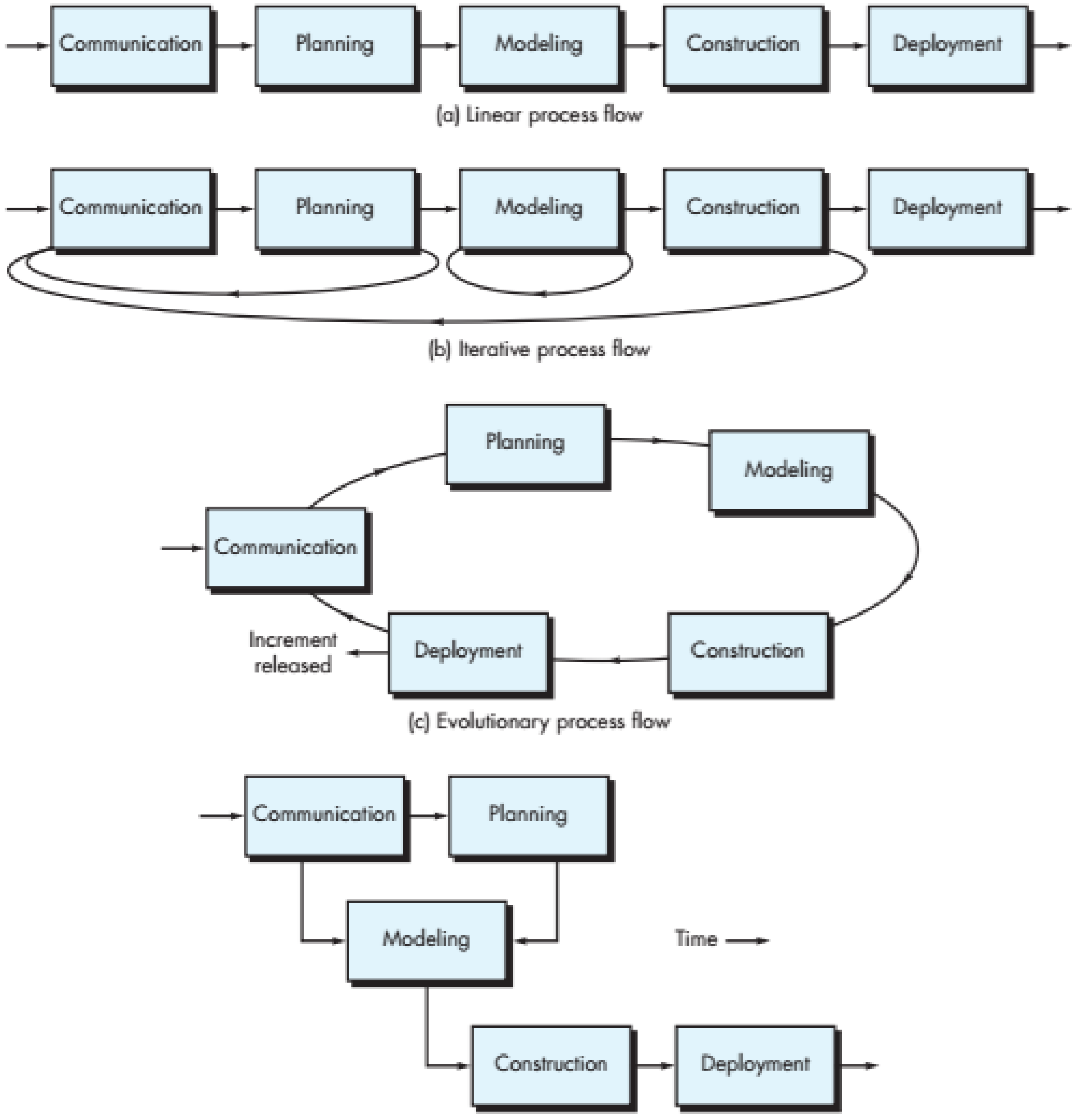
Communication. Before any technical work can commence, it is critically important to communicate and collaborate with the customer (and other stakeholders).[3] The intent is to understand stakeholders' objectives for the project and to gather requirements that help define software features and functions.

Planning. Any complicated journey can be simplified if a map exists. A software project is a complicated journey, and the planning activity creates a "map" that helps guide the team as it makes the journey. The map—called a software project plan—defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule.

Modeling. Whether you're a landscaper, a bridge builder, an aeronautical engineer, a carpenter, or an architect, you work with models every day. You create a "sketch" of the thing so that you'll understand the big picture—what it will look like architecturally, how the constituent parts fit together, and many other characteristics. If required, you refine the sketch into greater and greater detail in an effort to better understand the problem and how you're going to solve it. A software engineer does the same thing by creating models to better understand software requirements and the design that will achieve those requirements.

Construction. What you design must be built. This activity combines code generation (either manual or automated) and the testing that is required to uncover errors in the code.

Deployment. The software (as a complete entity or as a partially completed increment) is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation.



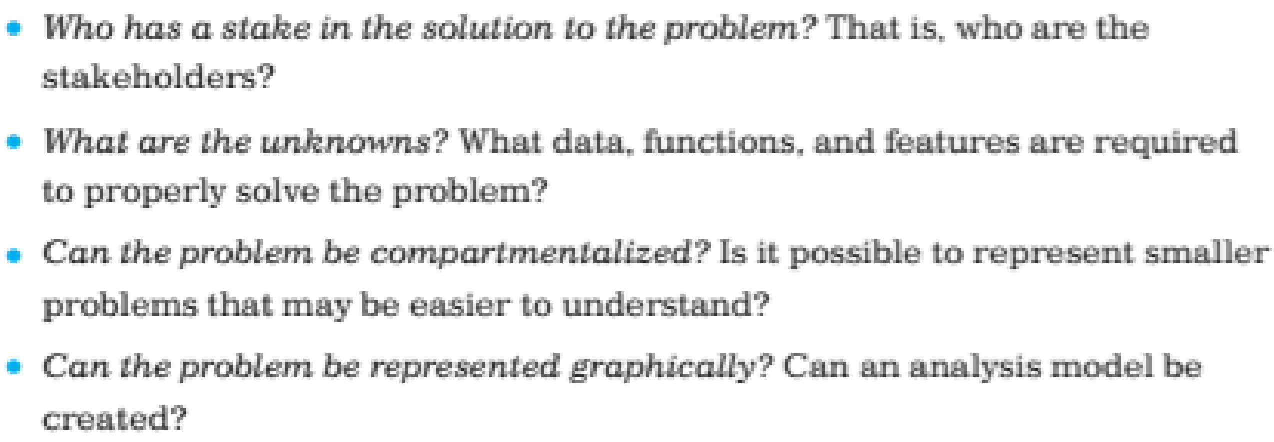
(a) Linear process flow

(b) Iterative process flow

(c) Evolutionary process flow

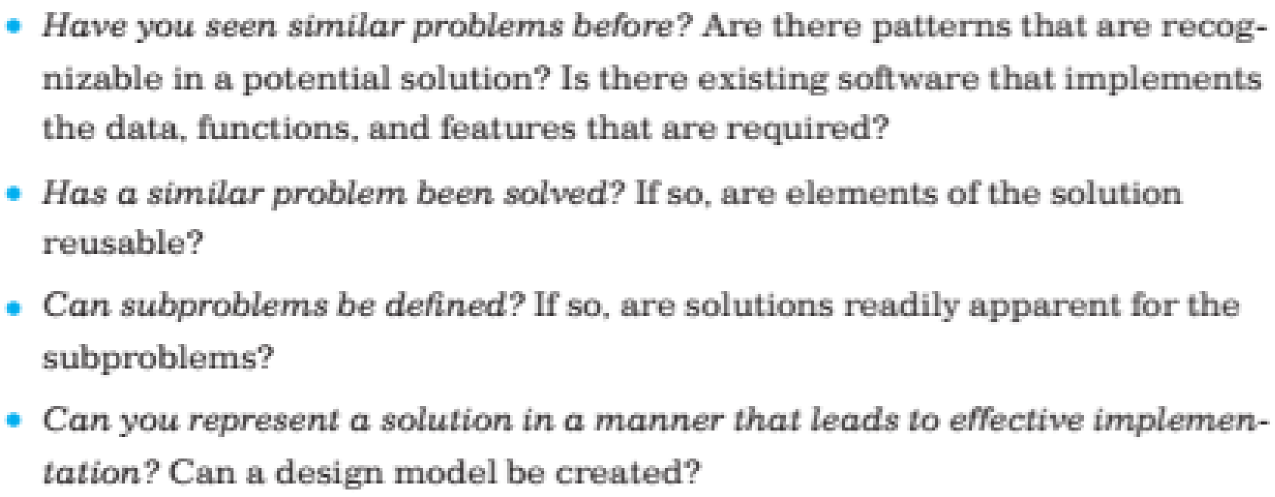
(d) Parallel process flow

Communication



- *Who has a stake in the solution to the problem?* That is, who are the stakeholders?
- *What are the unknowns?* What data, functions, and features are required to properly solve the problem?
- *Can the problem be compartmentalized?* Is it possible to represent smaller problems that may be easier to understand?
- *Can the problem be represented graphically?* Can an analysis model be created?

Planning



- *Have you seen similar problems before?* Are there patterns that are recognizable in a potential solution? Is there existing software that implements the data, functions, and features that are required?
- *Has a similar problem been solved?* If so, are elements of the solution reusable?
- *Can subproblems be defined?* If so, are solutions readily apparent for the subproblems?
- *Can you represent a solution in a manner that leads to effective implementation? Can a design model be created?*

Myth vs. Reality

- Add more programmers if you get behind schedule **(WRONG)**

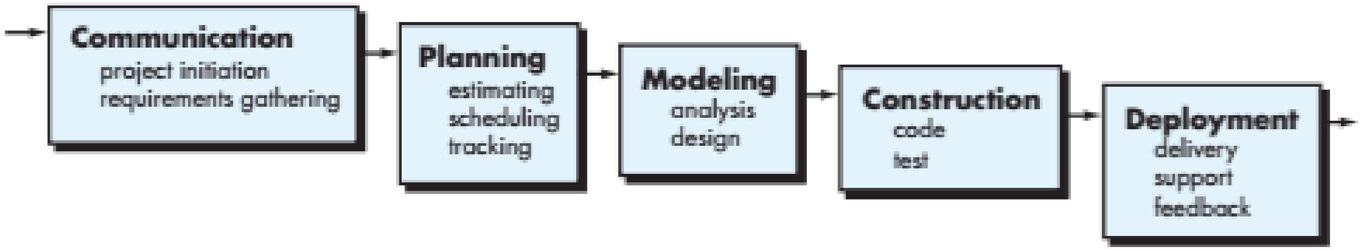
- A general statement of objectives is sufficient to write software programs – can fill the details later **(ABSOLUTELY NOT)**

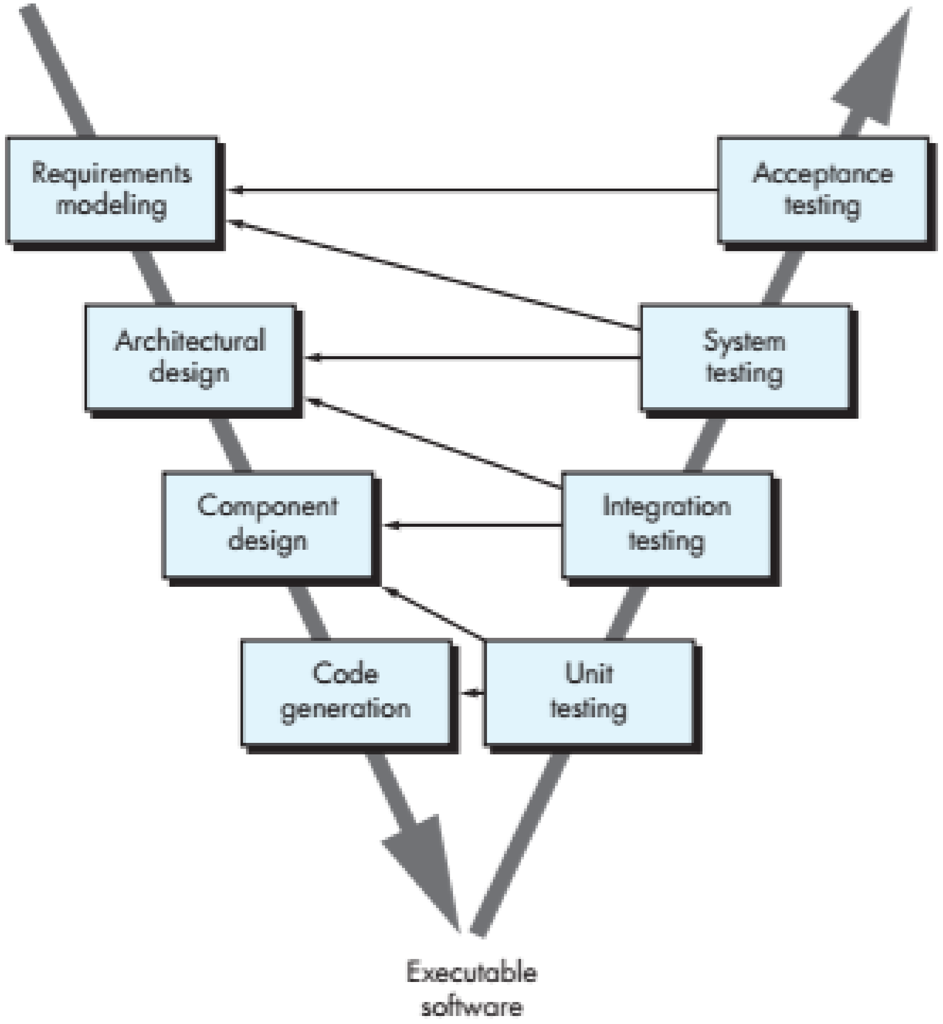
- Once we write the programs, our job is done **(NOT EVEN CLOSE)**

- The only deliverable work product is a successful program **(QUITE THE CONTRARY)**

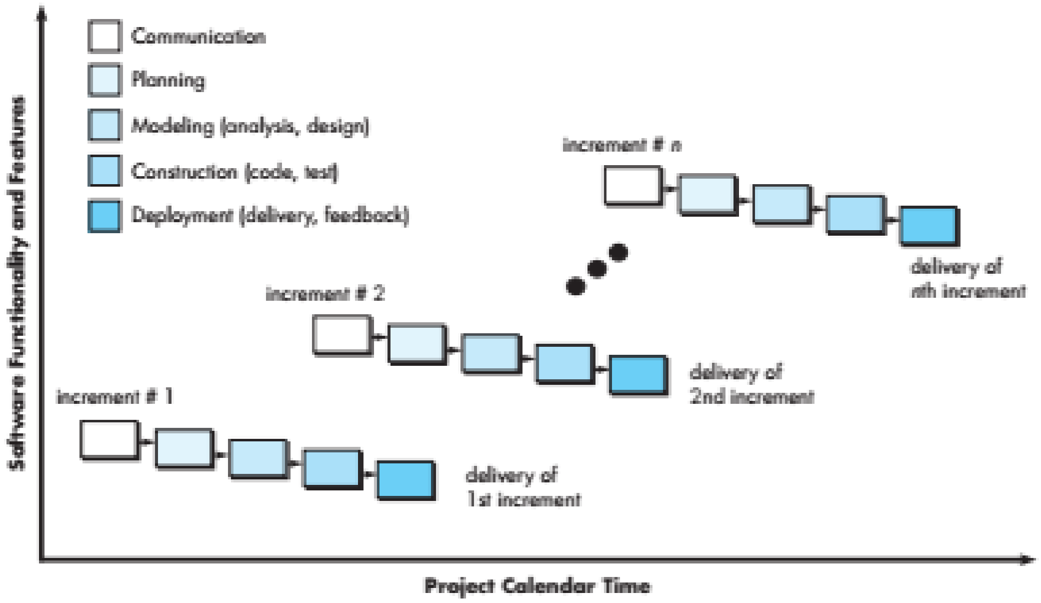
- Software engineering creates lots of documentation that slows us down **(MISSING THE FOREST FOR THE TREES)**

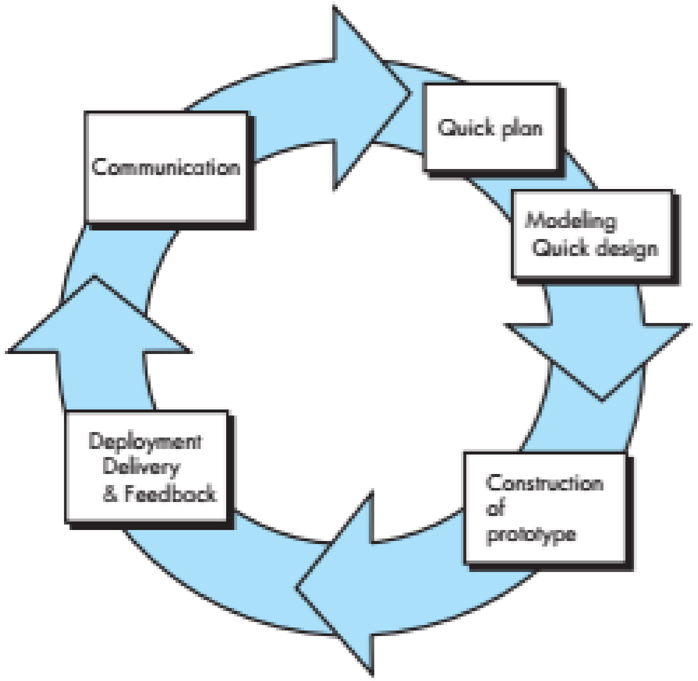
Waterfall Method & Systems Engineering V

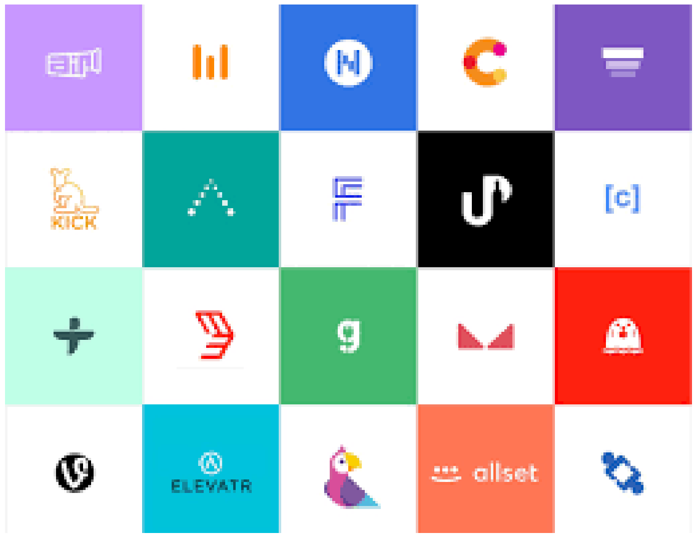




Incremental Method & Iterative Prototyping







General Principles

- “The Reason It All Exists”
- KISS (Keep It Simple, Stupid)
- Maintain Your Vision
- What You Produce, Others Will Consume
- Be Open to the Future
- Plan for Re-Use
- Think!