

Architecture in Large/Critical IT projects

Vinay Krishna
and
Dr. Anirban Basu

Agenda

- Introduction
- Project Failure Rate and Factors
- The major challenges with large IT project
- EA and Requirement Change
- Need for Agility in Architecture
- Data Collection
- Data Analysis
- Summary
- Q&A

Introduction

- Software Architecture
 - Inherited from Civil engineering
 - Lots of challenges (new and old) are there
 - Yet unknown world for others
 - Projected as a silver bullet by technical people for success of project
 - Still this is in pre-mature stage comparing to other engineering streams
 - It started evolving during Mid 1980

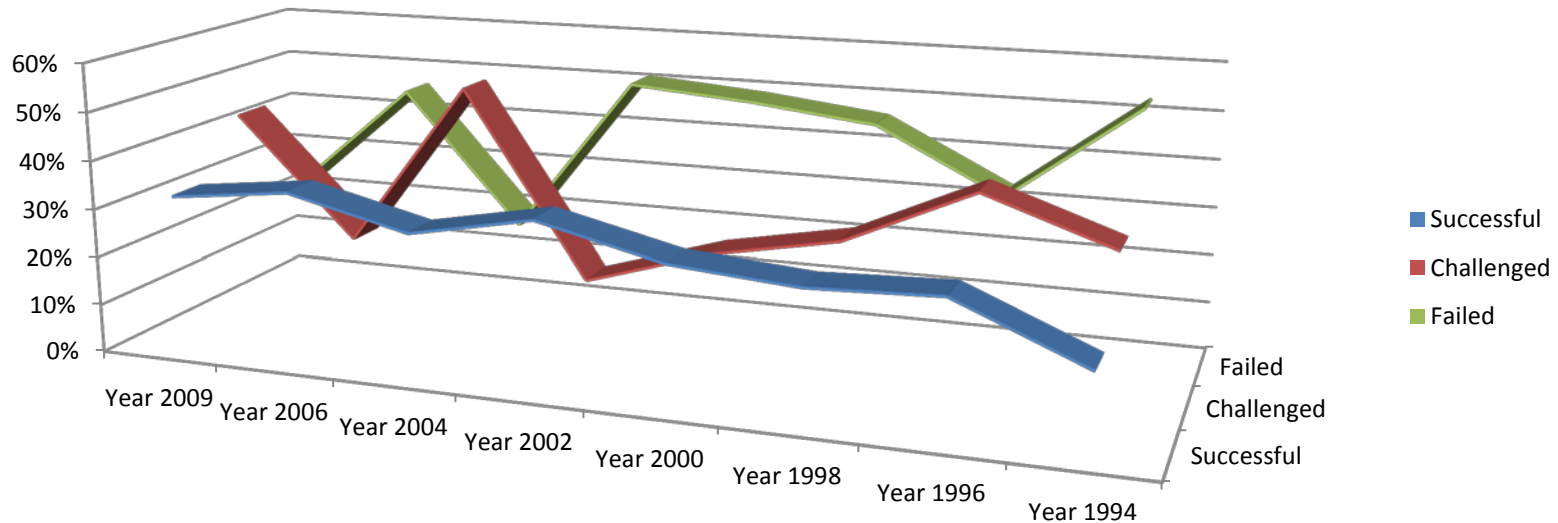
Introduction

- Architecture needs to be planned well in advance to adequately meet all functional and **non-functional requirements**
- This is important in case of Large/Critical application
- Design of the architecture for Large/Critical applications continues to be a challenge
 - Since it is very likely that requirements, technology, business case etc. will change within the planned duration of the project
- All these offer challenges to the Software Architects, who have to foresee these changes when architecting the system

Large/Critical application

- Characteristics
 - Requires more time to develop
 - More chance to change business, technology etc
 - Requires more time to analyze various scenarios
 - Requires more Anticipation
 - Introduce un-necessary complexity
- At the end
 - Architecture is not capable to handle the new request
 - Poor architecture. Why?

Project Failure Rates



[Figure 1: Based upon Standish Report]

Success rate improved from 16% in 1994 to 32% in 2009 😊

Failure rate (including both Challenged and Failed), is still quite high (68% in 2009). 😞

Project Failure Rates

Financial Analyses done by **Roger Sessions** based upon Standish data reveal that when indirect failures are added to write-offs, **the world's GDP suffered a \$6T USD loss due to IT failures in 2008.**

Size of project	Early	On-Time	Delayed	Cancelled	Sum
1 function point	14.68%	83.16%	1.92%	0.25%	100.00%
10 function points	11.08%	81.25%	5.67%	2.00%	100.00%
100 function points	6.06%	74.77%	11.83%	7.33%	100.00%
1,000 function points	1.24%	60.76%	17.67%	20.33%	100.00%
10,000 function points	0.14%	28.00%	23.83%	48.00%	100.00%
100,000 function points	0.00%	13.67%	21.33%	65.00%	100.00%
Average	5.53%	56.94%	13.71%	23.82%	100.00%

Table 1: Percentage of projects early, on-time, late, canceled
(from *Patterns of Software Systems Failure and Success*, by Capers Jones)

Cause



- It is very difficult to analyse the actual causes responsible for failure of software projects, as there are so many factors which cause failure.
- However, it is established that inadequate requirements development is a major contributor to failure of a large number of software projects irrespective of their size

Cause

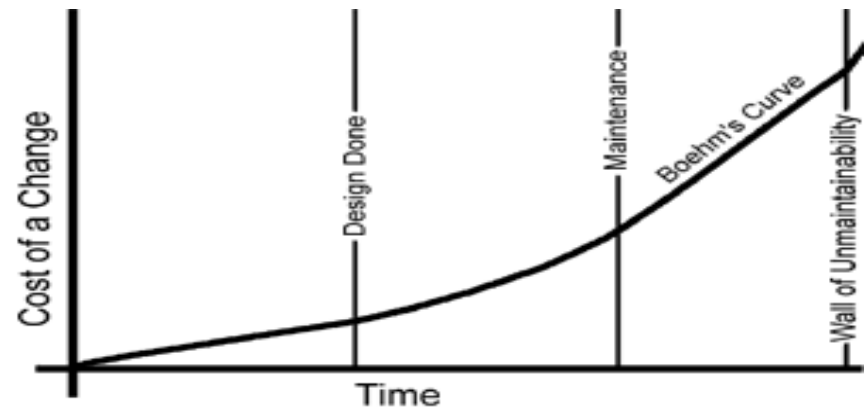
- System Requirements mature with time
 - Research of many projects shows that 45% of features proposed in the initial requirements were not used and an additional 19% were rarely used.
- Business changes over time
 - In a large organization, business changes happen every day.
 - The common business drivers for architecture change are business-as-usual development, business exceptions, business innovations and strategic changes.
- Rapid evolution of Technology
 - There are many technology-related drivers for architecture change. Request for new technology reports, asset management cost reductions, technology withdrawal and standards initiatives.

60% of all IT budgets are invested in projects that are at risk for unnecessary complexity.

EA and Requirement change

- Enterprise Architecture (EA) evolved twenty years ago to address two problems - System complexity and Poor business alignment
- The objective is to provide support to Business agility by use of IT whenever there is a need.
- EA is a top down process that starts with a strategy, vision and ends with implementation plan.

EA and Requirement change



[Figure2: Boehm's curve]

- Nevertheless it requires extra effort and cost and still adoption of change is a major challenge for EA.

Needs for Agility in Architecture



Needs for Agility in Architecture

- The only factor which is constant in any enterprise is change.
- Due to many factors both internal and external, one needs to cater to changes and this may require changes in architecture and design.
- The future needs must be well thought out early during requirement analysis phase. – **Not possible**

Our Study

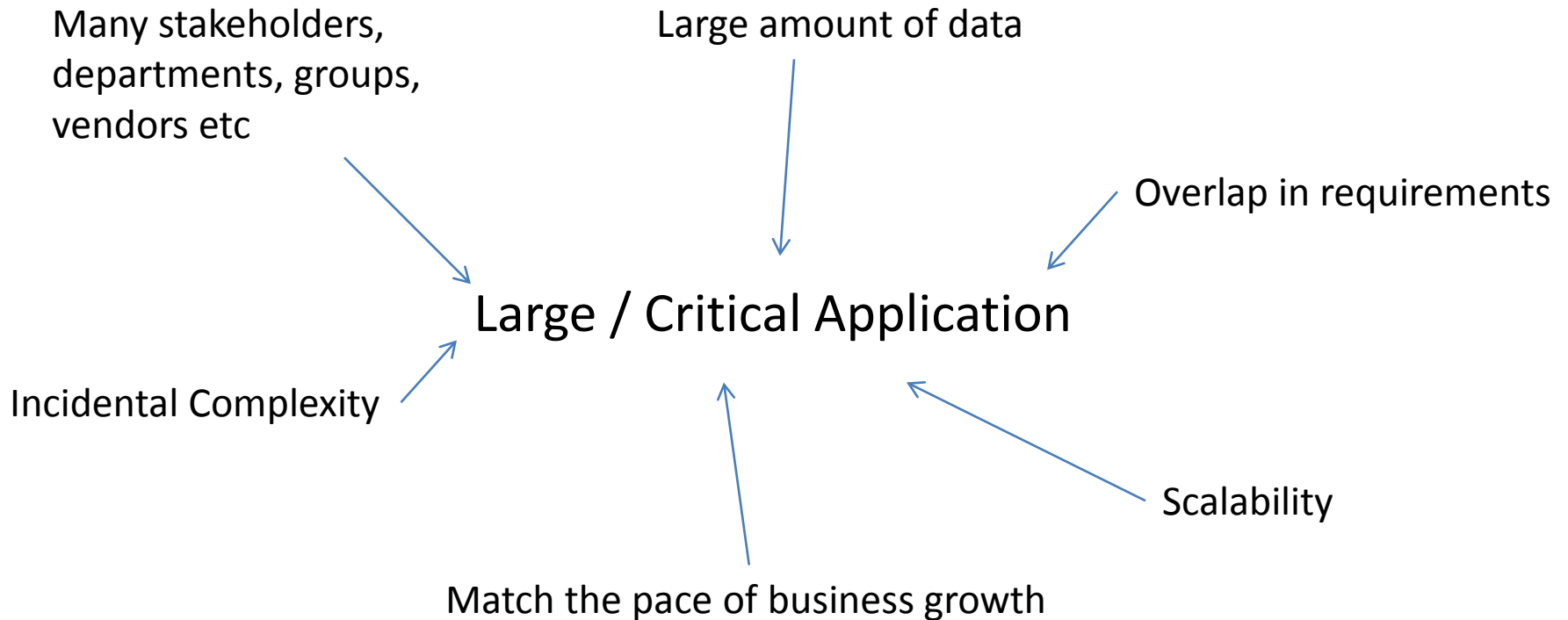
- We conducted a survey with open ended questions and collected data from different organizations from various countries such as USA, Netherlands, South Africa and India.
- We interacted with participants with different roles within large/critical IT projects: enterprise architects, software architects, technical leads, CTOs, COOs, and consultants.
- Data collection is still in progress.
- **Project Size:** Most of the projects belong to large category and project duration is between 1-3+ years.
- **Team Size:** 25-100+

Summary of Results so far

Category	Yes	No
Frozen Architecture & Design prior to Development	33%	67%
Evolutionary Architecture & Design approach	84%	26%
Just Enough Architecture	67%	33%
Architecture require agility	98%	2%

Analysis of Survey Data

Below are some common challenges encountered which have severe impact on architecture:



Challenges in Large Applications

- Many stakeholders, departments, groups, vendors etc
 - It's a very common problem in case of large/critical application
 - Create more chaos, confusion in terms of requirement

Challenges

- Large amount of data
 - Handling large amount of data is important
 - Need extra care, since its growing very fast
 - Possibility of noise

Challenges

- Overlapping of requirements
 - Multiple groups, departments involved
 - Noise are present in requirement
 - Redundant requirements

Challenges

- Scalability
 - Need to scale the application in terms of data and function
 - Most challenging

Challenges

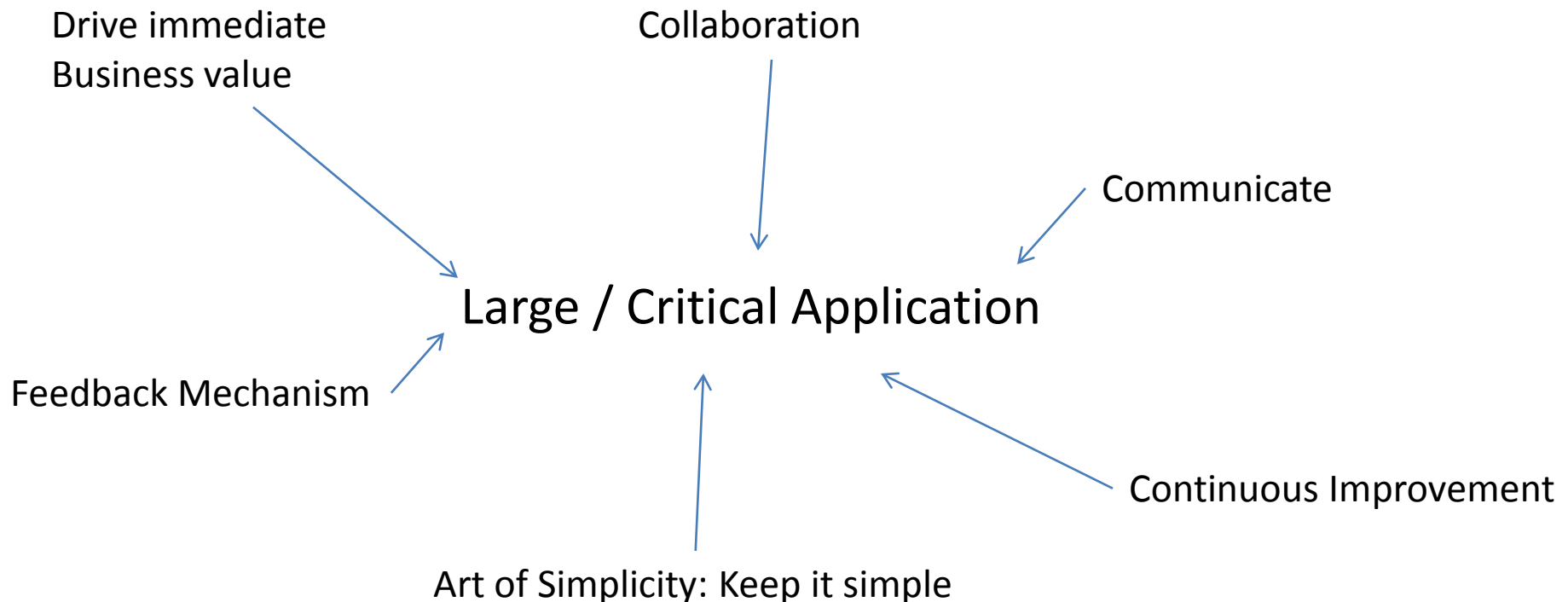
- Matching the pace of business growth
 - The only factor which is constant in any enterprise is change
 - A common belief is architecture must be able to handle it gracefully

Challenges

- Incidental Complexity
 - Also known as “Accidental complexity”
 - Arises from choices made in terms of technology, hardware etc to be used
 - Anticipation introduces more incidental complexity
 - In large-scale software, though, removing accidental complexity while retaining the solution to the essential complexity is challenging.

Recommendations

Based up on data analysis we determine below points gave rise to better architecture and design in case of large and critical IT projects:



Suggestions

- Drive immediate Business value
 - Just enough architecture
 - ~~Big UpFront Design or No UpFront Design~~
 - Priority: Time to market/value
 - Don't tell me, Show me

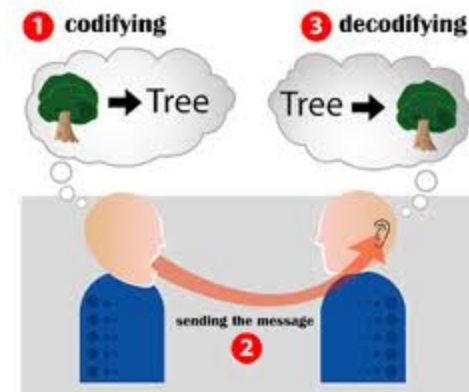
Suggestions

- Collaboration
 - Encourage collaboration with all stakeholders
 - Reduce noise in requirements



Suggestions

- Communicate
 - Communicate in the language of business
 - Build Strong trust

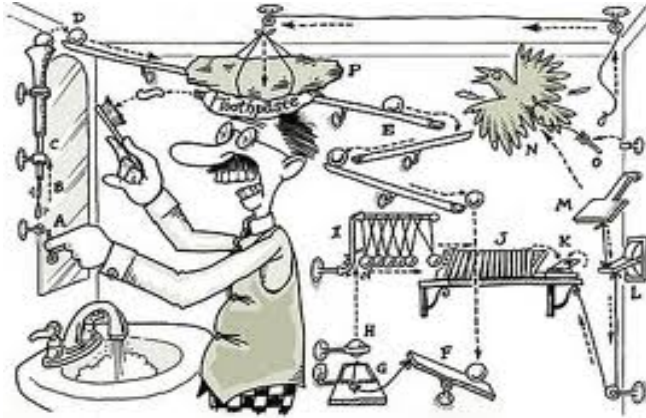


Suggestions

- Continuous Improvement
 - Measure, adapt and become the best
 - Develop mechanism to capture knowledge and share



Suggestions



- Art of simplicity – Keep it Simple
 - Don't choose any technology unless it critical or prove it
 - Justify essential complexity over incidental complexity

Suggestions

- Feedback Mechanism
 - Define mechanism to encourage early feedback from end user
 - Adoption over anticipation
 - Evolutionary architecture



References

- Software Engineering Institute (SEI) <http://www.sei.cmu.edu/architecture>
- S. Caddel, *"Software Architecture and the Use of Patterns: How Christopher Alexander's The Timeless Way of Building can be applied to Software Design"*, 2001, URL <<http://library.colorado-tech.com/papers/2001/tr01caddel006.pdf> >
- M. Fowler, *"Is Design Dead?"* Articles July, 2000 [cited on May, 2004], URL <<http://martinfowler.com/articles/designDead.htm>>
- Rotem-Gal-Oz, *"Software Architecture – 5 years later"*, 17, July, 2010, URL <[http://architects.dzone.com/news/software-architecture-%E2%80%93-5?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+zones%2Fruby+\(Ruby+Zone\)](http://architects.dzone.com/news/software-architecture-%E2%80%93-5?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+zones%2Fruby+(Ruby+Zone))>
- Sessions, Roger, *"The Cost of IT Failure"*, URL <http://simplearchitectures.blogspot.in/2009_09_01_archive.html>
- J. Donaldson, *"SYSTEMS FAILURES: An approach to understanding what can go wrong"*, July, 2002, URL <http://www.eis.mdx.ac.uk/research/SFC/Papers/AJMD_EuroMicro00.pdf>
- R. Veryard, *"Business Change and Process Improvement"*
- <http://www.standishgroup.com/>
- B. Danette Allen, *"Designing for Change: Minimizing the impact of changing requirements in the later stages of a spaceflight software project"*, NASA/TM-1998-208466
- TOGAF, Architecture change management, <http://pubs.opengroup.org/architecture/togaf8-doc/arch/>
- MSDN, <http://msdn.microsoft.com/en-us/library/bb466232.aspx>
- N. Brown et al, *"Enabling Agility Through Architecture"*, CrossTalk-Nov/Dec 2010
- A. West, *"Final Report, NASA Study on Flight Software Complexity"*, May, 2009
- P. Kruchten et al, *"Agility and Architecture: Can They Coexist?"*, March/April 2010, IEEE
- C. Larman et al, *"Agile & Iterative Development – A manager's guide"*



Questions?