

Vilas Prabhu
Wipro Consulting Services

I rest my (business) case

22nd Enterprise Architecture Practitioner's Conference, London - April, 2009







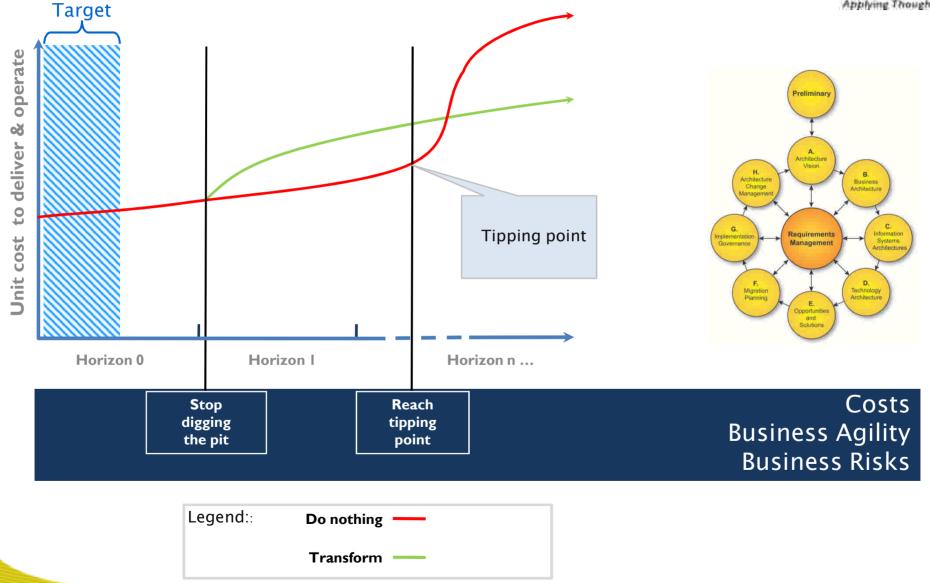
- Target audience
- What is the issue?
- Why there is an issue?
- Explaining the approach



- Target audience
- What is the issue?
- Why there is an issue?
- Explaining the approach

Target audience



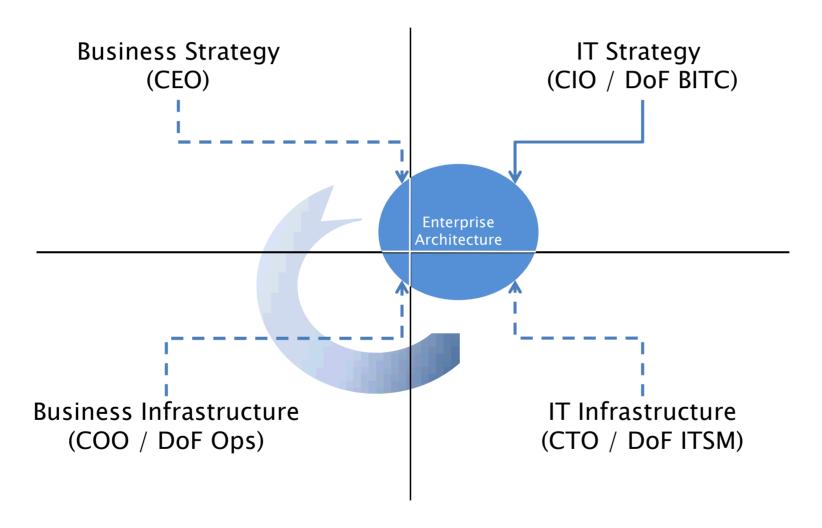




- Target audience
- What is the issue?
- Why there is an issue?
- Explaining the approach

How IT change happens?

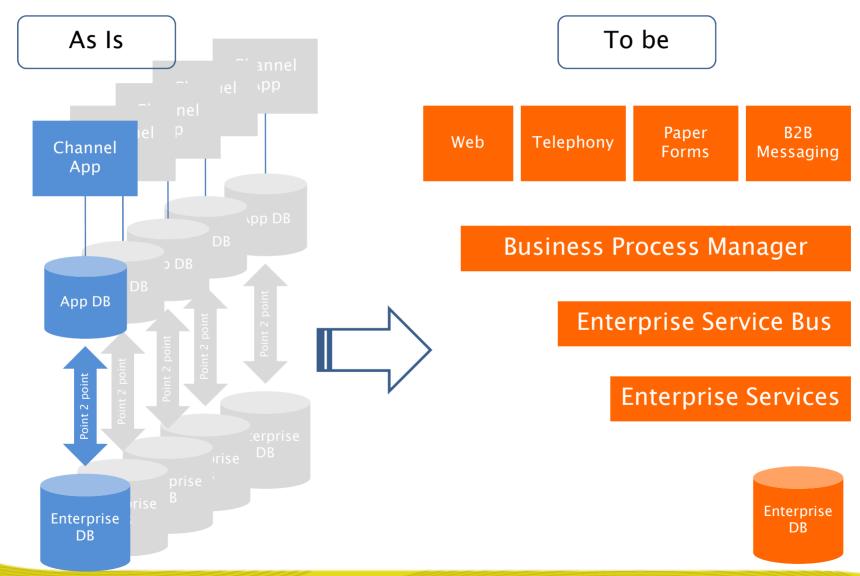




BITC - Business & IT Change, ITSM - IT Service Management, Ops - Business Operations

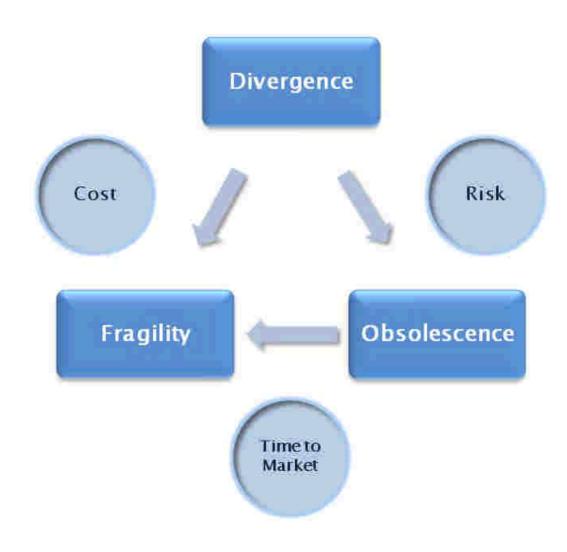
Information systems architecture divergence













- Target audience
- What is the issue?
- Why there is an issue?
- Explaining the approach

Why there is an issue?









Divergence is affordable

Divergence draws on 'Architectural debt'

Architectural debt is not measured



- Target audience
- What is the issue?
- Why there is an issue?
- Explaining the approach

Explaining the approach



 Divergence is affordable because it draws on 'Architectural Debt'

Highlight Debt

Measure Debt

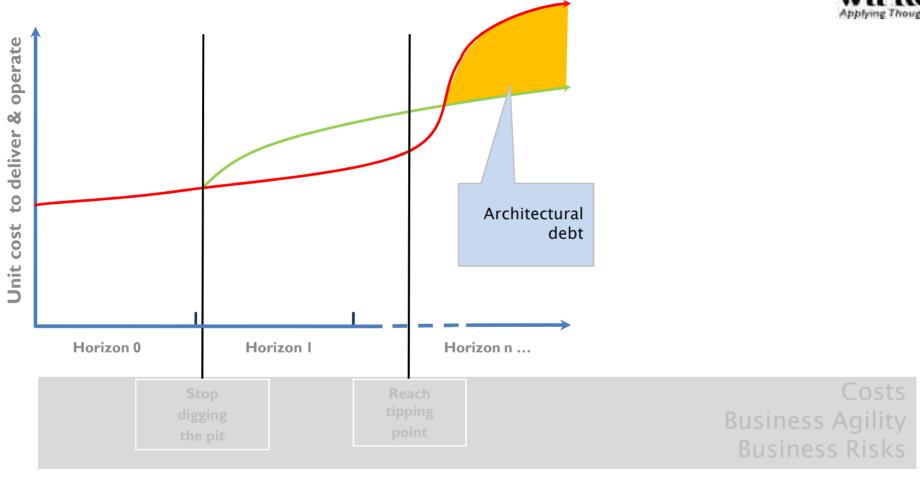
- More divergence means more debt
- Make debt explicit by measuring

- · Use accumulated debt
 - To pay for transformation
 - To influence behaviour

Retire Debt

Architectural debt





Legend: **Do nothing**Transform

Explaining the approach



 Divergence is affordable because it draws on 'Architectural Debt'

Highlight Debt

Measure Debt

- Atake debt explicit ta measuring
- Por architecture life= dydle commit implace

- Use accumulated debt
 - To pay for usin stormation;
 - To influence behaviour

Retire Debt

What is it?



Architectural Debt is invisible credit received per unit of output It will be repaid as

Delivery

- · Future additional costs
- · Future additional time
- · Future additional risks

Operation

- · Future additional costs
- · Future additional risks

Capability

- · Future lack of capability
- · Future lack of agility

What is it? (contd.)

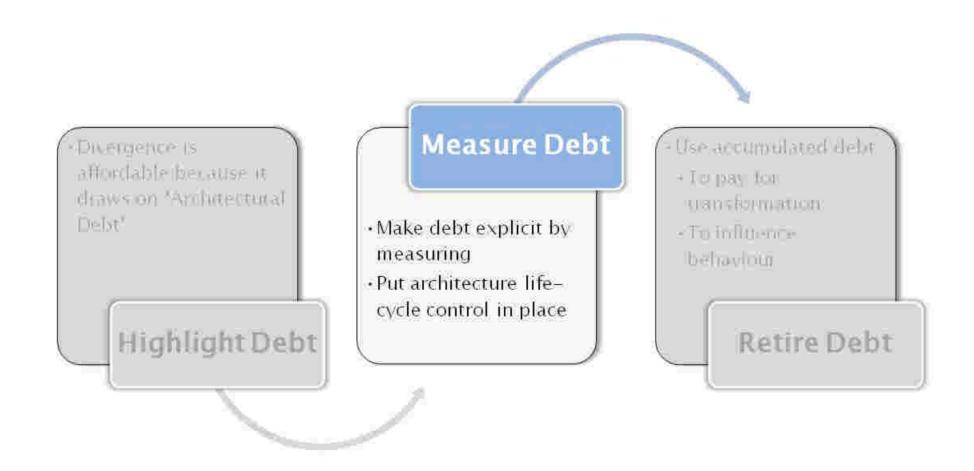


E.G. Divergence from information systems and technology architecture

- Impacts in terms of
 - Development costs
 - Development efforts
 - Integration costs
 - Integration efforts
 - Regression testing costs
 - Regression testing efforts
 - Potential migration costs
 - Potential migration efforts
- Impacts operation
 - Capacity enhancement costs
 - Risk response (BCP, DR) costs

Explaining the approach





Architecture life-cycle control



Effect change

- Make component ready for deployment
- Compute debt per unit to be carried in case of divergence

Effect lifecycle changes Track Requirements

Track requirements as BAU

- Business requirements
- Regulatory requirements
- External relevant events
 - Mergers, acquisitions
 - Technology announcements
 - Supplier announcements
- Competitor actions

Assess Impact

Assess Impact

- Analyse and filter requirement
- Prioritise requirements
- Propagate requirements to impacted architectural components
- Assess whether change in lifecycle stage is warranted (TCO, risks, effect on TTM etc.)

Architecture component's lifecycle



Exit Entry Component need to be ·Component fitness established researched and identified ·Starter skills, processes for use Skills and infrastructure and infrastructure needed are to be established ·Architecture definition established and developed **Evaluate** fneeded Iteration complete Entry Entry ·Component evaluated and fit: *External event forcing for adoption retirement ·Starter skills, processes and *Planned obsolescence due to Infrastructure developed escalating costs and risks Adopt Track Retire Requirements Exit Exit Component adopted *Component usage stopped successfully on small scale or in across board ab environment ·Architecture governance ·Infrastructure, skills and iteration complete processes ready •Transition planning iteration complete Exit Entry Component ready for Maintain *External relevant event Exit evaluation Entry · Necessary started skills in trigger ·Planned retirement due ·Replacement component. *Lifecycle phase trigger place has reached Adopt state to obsolescence · Necessary starter ·External event infrastructure in place triggering unplanned · Necessary starter retirement processes in place Architecture context Iteration complete

Business cases without and with debt



Compliant / Conformant

- Delivery
 - Build effort
 - Quality Assurance effort
 - Change management effort
 - Infrastructure



- Operation
 - Infrastructure
 - Support effort
 - Maintenance effort
 - Risk mitigation

Divergent

- Delivery
 - +
- Operation
 - +
- Replacement
 - Build effort
 - Execution effort
 - Quality assurance effort
 - Risk mitigation effort (in case of retired component)
 - Infrastructure
 - Effort due to impact from/onto other divergent components

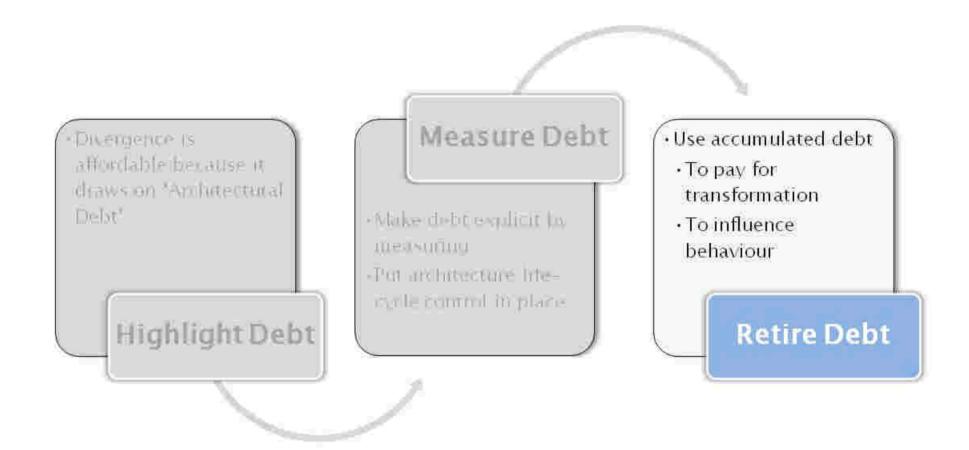
How to measure size?



- Information system's change size can be calculated as normalised testable wants and needs for all stakeholders (users, operators, decision makers etc.)
- Complexity measure can be used to normalise the size
- User acceptance testing efforts gives an indication of complexity
 - Estimated test set up time
 - Configurations
 - Test data complexity
 - Estimated test execution time
 - Estimated test verification time
- Operational changes to be sized empirically as it is easier to estimate

Explaining the approach





Retiring architectural debt



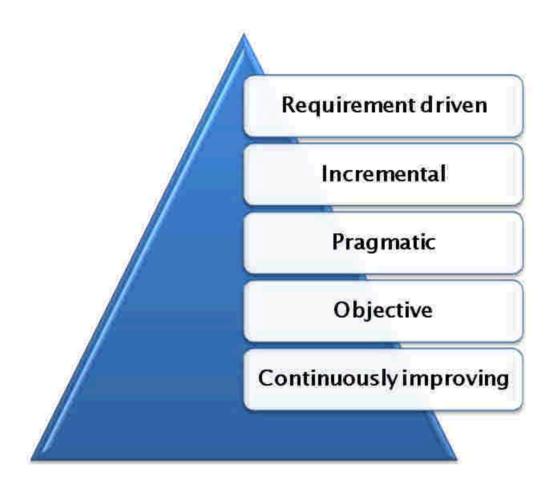
In case of divergence, future replacement costs to be paid into debt retirement pot (during delivery and operation)

Accumulated debt retirement pot to be used to migrate non-compliant usage when it crosses fixed cost threshold

After migration Architecture component moved to 'Retired' state. All subsequent users will pay additional risk premium for usage of component, and pay even more to debt retirement pot.

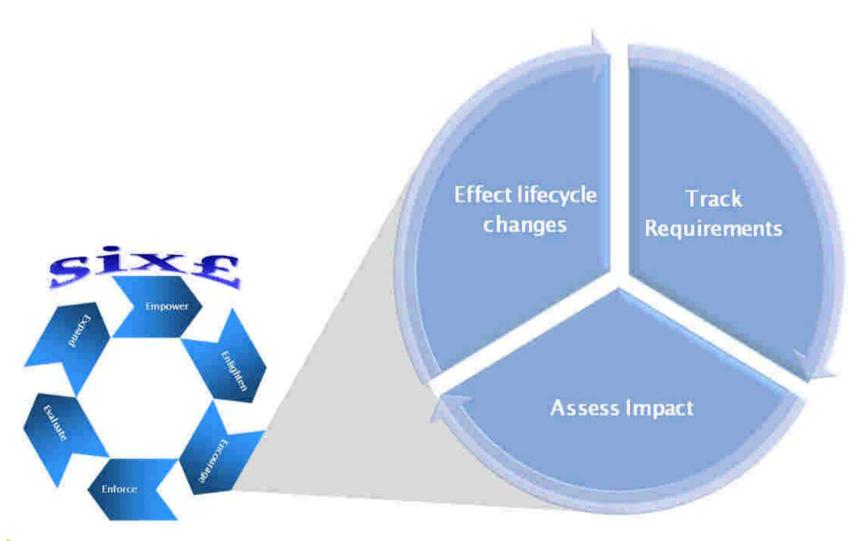






SixE framework fit









Thank You

Vilas Prabhu

http://vilasp.blogspot.com