UCL

Software Engineering Governance: a briefing

Anthony Finkelstein
Computer Science

What I Intend to Do

- Making the case for a new research arena
- Reviewing challenges and contributions
- Presenting examples

Why Projects Fail …

User Involvement
Clear Business Objectives
Controlled Scope
Standard Software Structure
Firm Basic Requirements
Formal Methodology
Reliable Estimates

From Standish Group CHAOS Reports

...1991...1992...1993...1994...1995...1996...1997...1998...1999...2000...2001...2002...2003...2004...2005...2006...2007...2008...

Why does history repeats itself?

One of the large unanswered questions in software engineering

Flawed techniques – Inherent difficulty

Response: new techniques

Ignorance – Poor training

Response: make it easier, improve transfer

Laziness – Ill will

Response: improved control frameworks

An Alternative *Theory*

That organisations are unable to avoid these problems because of structural issues and in particular problems (mismatches) at the interface between the structure of the business organisation and the organisation of software development

Specifically …

- Complex system 'ownership'
- Misalignments in incentives
- Difficulties in securing 'accountability' for critical decisions

This *theory* is supported by some informal observations … illustrated later in this briefing

… the relationships between business structures and software engineering are poorly understood and under-researched, for example the relationship between commercial procurement practice and software development

The core area of concern here is what has become known as 'governance'

I will use the term Software Engineering Governance to capture my focus on software development

Definition(s)

Software Engineering Governance is the set of structures, processes and policies by which the software development and deployment function within an organisation is directed and controlled so as to yield business value and to mitigate risk

Often erroneously thought to be principally about regulatory compliance

Related to …

Software Engineering Governance is a component part of Corporate Governance - *the set of structures, processes and policies by which an organisation is directed and controlled so as to …*

align interests and incentives in the interest of the organisation as a whole within a framework of openness and transparency

Key Themes

A shared notion of business value

Mitigation of risk

Alignment of interests and incentives

Legislation …

Large corporate failures in the late 1990s focused attention on governance, giving rise to legislation (eg SOX). This attention necessarily 'trickles down' to the software function as a major means by which a business obtains value and a locus of cost and risk

Observation …

The centrality of software systems to organisational performance is increasing significantly faster than development risk is decreasing

It is a critical organisational arena in which misalignments of interests and incentives manifest themselves

Regulatory Pressure is Important

This is one of the few arenas where senior executive management are directly engaged.

Looking at governance is timely… changes in enterprise architectures and software development methods raise new challenges and existing structures and processes are failing.

New enterprise architectures (based for example on SOA) decouple services, processes and platform cutting across existing business structures.

Federated data management, integration and messaging change patterns of information ownership and control that have been a dominant means of structuring enterprises.

Outsourcing and external service provisioning move control across enterprise boundaries and alter the ‘touch-points’ within enterprises.

Agile development changes lines of management control and accountability. Self organising teams present particular governance difficulties.

Software Engineering (research at least) tends to adopt a project by project, product by product focus

It is important to distinguish governance from the *direct* managerial control mechanisms necessary to ensure 'low-level' good practice is followed

Adherence to mandated processes, use of libraries and configuration management, interface control, metrics gathering and so on

This only becomes a governance concern where their absence reflects some underlying differences in the determination of risk or in the incentives of the parties engaged.

Hence audit and monitoring

The State-of-the-Art … ‘standards’ and ‘best practice frameworks’

ISO/IEC 38500: 2008 Corporate governance of information technology

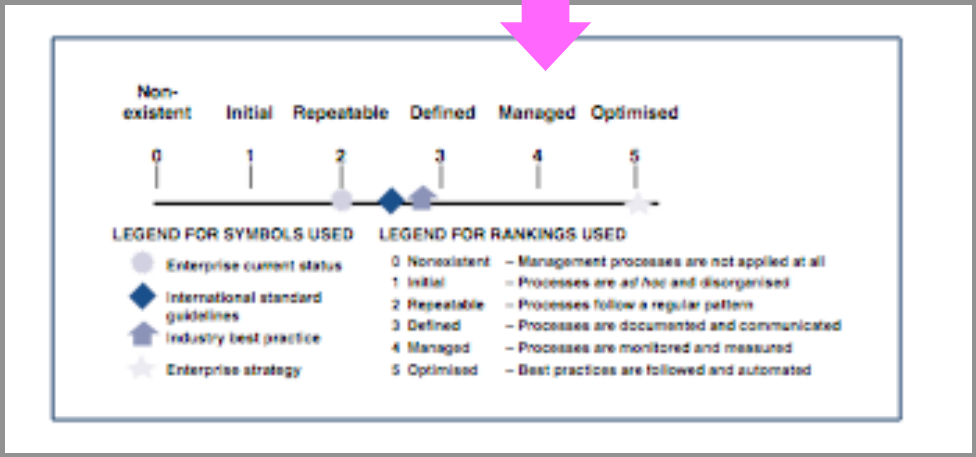
and national variants and precursors

COBIT: Control Objectives for Information and Related Technology (ISACA - Information Systems Audit & Control Association and ITGI - IT Governance Institute

And of course …

The inevitable maturity model

IT Governance Institute 'Board Briefing on IT Governance'



ITGI focal areas for governance

Strategic alignment

Value delivery

Resource management

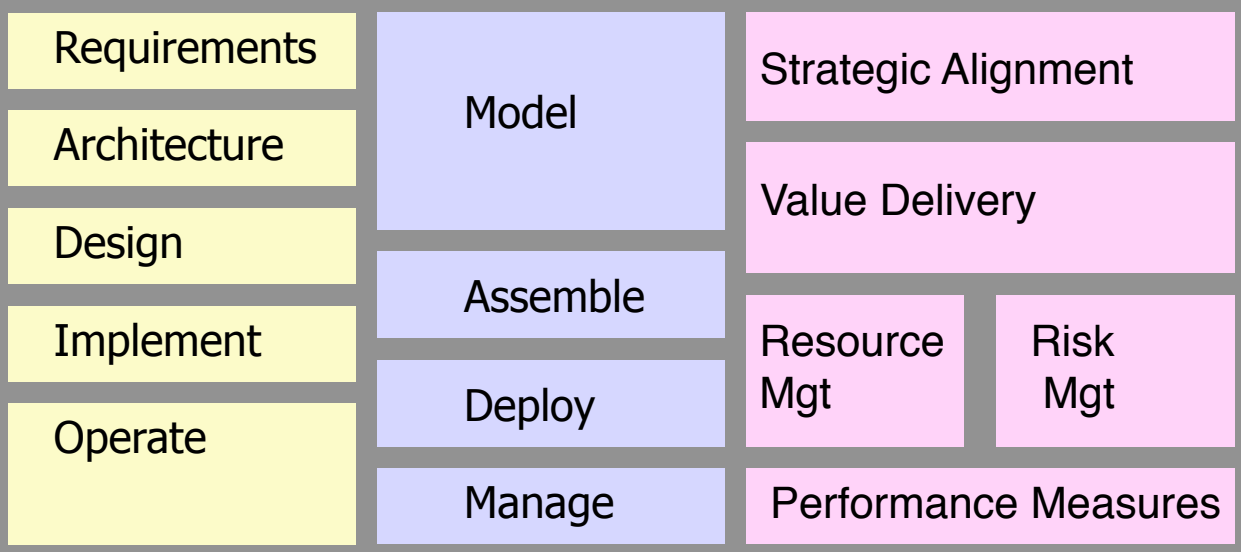
Risk management

Performance measures

All of which directly impinge on Software Engineering

Lifecycle

There is a need for governance at every stage of the life of the system. The balance of attention shifts across focal areas as development proceeds.



Software Development Governance 2008 & 2009:
Yael Dubinsky & Phillipe Kruchten

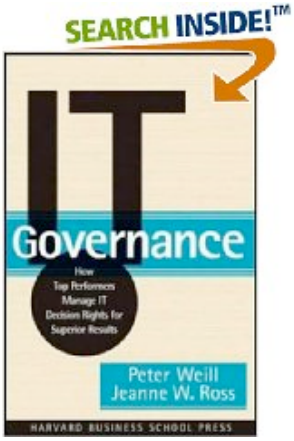
Emerging definitions and scoping challenge
Bottom-up vs Top-down tension
Small number of 'agreed principles'
Slightly large number of useful techniques

Key research contribution:

Peter Weill & Jeanne Ross

'IT Governance: How Top Performers Manage IT Decision Rights for Superior Results, Harvard Business School Press (2004).

Note the connection between performance and governance



10 Principles of IT Governance

1. Actively design governance
2. Know when to redesign
3. Involve senior managers
4. Make choices
5. Clarify the exception handling process

10 Principles of IT Governance

6. Provide the right incentives
7. Assign ownership and accountability for governance
8. Design governance at multiple organisational levels
9. Provide transparency and education
10. Implement common mechanisms across assets

Implications for Software Engineering

- Incorporate governance design in process configuration and management activities
- Consider governance when introducing significant architectural or process changes
- Direct senior management attention to implications of changes

Implications for Software Engineering

- Provide a structure for highlighting conflicting goals
- Develop coherent structures from Board-level downwards
- Expose rather than hide governance
- Lead the governance debate within the enterprise

Structures typically in place

Board level - strategic investment management

Executive level - business case scrutiny and requirements management

Group level - technical authority

Operational level - monitoring execution of key decisions, risk and compliance

Operational level - design review and architecture compliance

Enterprise Architecture Challenges

Because business logic is shared outside traditional silos the potential company -wide impact of any given service becomes greatly increased

Complex ownership of services and relationships

Difficulties of aggregating services on a shared platform that delivers the appropriate non-functional properties

Why is SOA governance particularly difficult?

- Ease of creating and using 'rogue' web services
- Incoherent architecture arising from services developed in projects chartered to solve conflicting business problems

adapted from Laurent, 2007

Symptoms of Poor Governance

- Single use services and point-to-point connections
- Proliferation of redundant services and data types
- Inconsistent implementation of cross-cutting capabilities (security, reliability, transactions, logging, routing, filtering)

adapted from Manes, 2007

Case studies (close to home)

‘CAPSA and its Implementation’

Report to the Audit Committee and the Board of Scrutiny of the University of Cambridge (October 2001)

Experience points clearly to the intimate relationship between governance and successful system development and deployment

Lesson learned …

An organisation with a flawed governance structure cannot articulate its requirements, charter a project, identify appropriately skilled staff, manage the concomitant change process, determine if the project has been successful or even deal with the consequences of failure

Case studies (close to home)

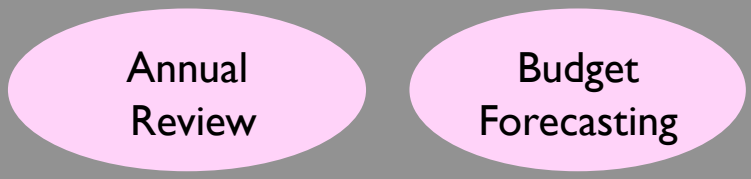
ABC is a large, research-intensive, metropolitan university in the UK. It has a dedicated and professional IT services function that engages in small-scale development and large-scale customisation and deployment projects.

A participant-observer

I have strong sense that the biggest problems I encountered have their origins at the interface between governance and requirements engineering

Example I

'Left Field'



Complex processes with substantial IT implications introduced as it were 'out of left field', that is from other 'lines of governance'.

Challenge: how can process and business governance arrangements be meshed with software governance

Example II

Technical Fix

Common Timetable

Decisions driven down to too low a level in the governance structure leaving the technology to leverage the change. Inadequate intermediate level structures to mediate between strategic intent and execution

Challenge: how to ensure decisions and responsibility for changes are made at the right level within the organisation

Example III

'CEOs iPhone'

Research 'Database'

Failure to maintain the integrity of the planning and governance process in the face of senior management decision making

Challenge: how to find structures that are responsive and preserve strategic leadership but also support a stable, planned and directed programme

Example IV

Nobody's baby

CoI Declarations

'Orphan processes' that are not strongly owned and thus never receive the necessary advocacy to have their requirements heard

Challenge: to identify and to 'promote' orphans, particularly if they are high aggregate value, or low-hanging fruit

Example V

Favoured Sons

Staff Recruitment

Very strong ownership of a cross-cutting process by a single organisational player distorting the governance process

Challenge: to put in place mechanisms that enable collective ownership without diluting value

Example VI

Handling Failure

Student Records

Success has many fathers, failure is an orphan.

Challenge: to build governance arrangements that can take risks and assume responsibility without inducing a 'blame culture'. These arrangements continuing when a project is perceived to have failed.

It seems easier to know what *not* to do than actually what should be done. There are some governance anti-patterns implicit in the examples I have presented.

Known Barriers

- Shifts in decision rights and associated power
- Resistance to accept accountability
- Inability to obtain sufficient business involvement
- Particular complexity with federated and outsourced business structures

What we do know …

Centralised governance for architecture and platform, decentralised for services and applications, lightweight (with central oversight) for processes

With management focusing on business goals that cross-cut system structures … means we need to rethink reporting

Use cost transparency and charge back as a key lever to effect change

Providing a clear mechanism for making business value visible

This is another area that is unexplored from a research standpoint

Substantial growth in risk and compliance audit, most notably in the area of security

Tendency to more 'negative' governance than 'positive' governance

Disaggregated risk management – process risk, architectural risk, operational risk and business risk not correlated

Audit and compliance instruments not compatible with software development methods


Strategic Management
Law & Regulation
Software Economics
Stakeholder Analysis
governance a new research challenge?
Software Development Methods
Corporate Governance
Risk Management
Security Engineering
Policy Modelling & Analysis