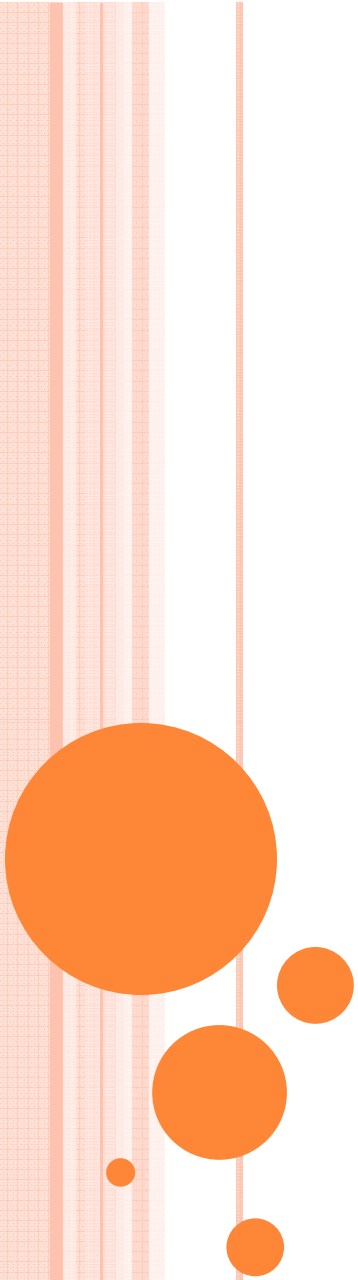


SOA Governance Optimizes the Business and Evolution of Service-Oriented Systems

Hausi Müller,[1] Priyanka Gupta,[1] Leho Nigul,[2]
Ron Desmarais,[1] Alexey Rudkovskiy,[1]
Norha Villegas,[1] Qin Zhu[1]
[1] University of Victoria, Canada
[2] IBM Center for Advanced Studies, Canada

MESOA 2009
September 21, 2009
Edmonton, Alberta, Canada

OUTLINE

- Need for SOA governance
- SOA vs. IT governance
- Characterizing SOA governance
- SOA governance pillars
- SOA governance mechanisms facilitate the optimization of business and evolution objectives of service-oriented systems
- Traditional evolution mechanisms
- Feedback loops
- Levels of indirection
- Research challenges

Need For Governance & Adaptation in Service-Oriented Systems

- Great benefits can be realized when an enterprise transforms its architecture to SOA because of the distributed and flexible nature of services
- But chief architects have a hard time to manage the entire service portfolio across various business lines
- Understanding, controlling, and managing uncertainty and run-time dynamics is crucial given the ever-changing business environment
- As a result, service-oriented systems resort to self-adaptation and self-management for dynamic service management and service composition

Controlling Dynamic Service Selection

- The value chains of today are the result of linked individual business tasks that come together to form a valuable end product
  - Just like a physical assembly line, each participant in the value chain contributes something to increase the value of the end product for the end user
  - Although these value chains have become increasingly distributed, they still tend to be predictable, structured and linear
- The combination of participating service providers is changing dynamically based on who is in the best position to perform a given task at a given time
  - These service providers themselves are becoming interconnected with one another to the point that mapping their relationships yields more of a net than the traditional linear chain
  - Familiar value chains are morphing into dynamic value nets
- The workings of the value net are orchestrated by the organization that delivers the end product to market under its own brand name
  - This orchestration itself may very well be the lead brand organization's unique value added

Steve Mills, VP IBM Software Group: The Future of Business, White Paper, June 2007

SOA GOVERNANCE

- Governance has been rated as the main inhibitor of SOA adoption
- SOA governance provides a set of policies, rules, and enforcement mechanisms for developing, using and evolving service-oriented systems, and for analysis of their business value
- SOA governance includes policies, procedures, roles and responsibilities for design-time governance and runtime governance

G. Lewis, D. Smith: SOA and its Implications for Software Maintenance and Evolution, ICSM FoSM 2008

SOA vs. IT Governance

- IT governance
  - Establishing rules, regulations, and policies for working with these modern, decentralized, distributed socio-technical IT ecosystems.
  - Specifically governance involves establishing chains of responsibility, measurements to gauge effectiveness, goal policies to meet business objectives, feedback and control mechanisms to assess compliance, and effective communication among stakeholders.
- SOA governance
  - Deals specifically with the lifecycle of services, service information models, service components and registries, service providers, service consumers, and service processes and with cross-cutting aspects such as service development, identification, agreements, quality attributes, selection, composition, versioning, management, and orchestration.
  - Guides the development, operation and evolution of services and establishes agreements between providers and consumers of services.
  - Service level agreements (SLAs) provide a specification of the verifiable quality characteristics that a user can expect from a provided service.

SOA Governance Types

- Design-time governance
  - Includes elements such as rules for strategic identification of services, development, and deployment of services; reuse; and legacy system migration to services
  - Enforces consistency in use of standards, SOA infrastructure and processes
- Runtime governance
  - Enforces rules to ensure that services are executed only in ways that are legal and that important runtime data is logged
  - Service level agreements (SLAs) including runtime validation of contractual specifications on performance, throughput, and availability; the use of automated metrics for tracking and reporting; and problem management

G. Lewis, D. Smith: SOA and its Implications for Software Maintenance and Evolution, ICSM FoSM 2008

CHARACTERIZING SOA GOVERNANCE

- SOA life cycle models
  - SEI life cycle (Lewis/Smith)
    - Strategic analysis
    - Planning
    - Constructions
    - Transition
    - Production
  - Waterfall model
    - Early planning stages
    - Design time
    - Deployment time
    - Run-time
    - Steady-state time
  - Spiral model (IBM)
    - Plan
    - Define
    - Enable
    - Measure
- SOA solution providers
  - People-centric (TIBCO)
  - Policy-centric (Oracle)
  - Process-centric (IBM)
  - Technology-centric
  - ITSM-centric (IBM more & more)
- Application
  - SLAs
  - Service composition
  - Dynamic service attributes
  - User contexts
  - Evolution processes
  - ITSM processes
- Design views
  - Entities and relationships

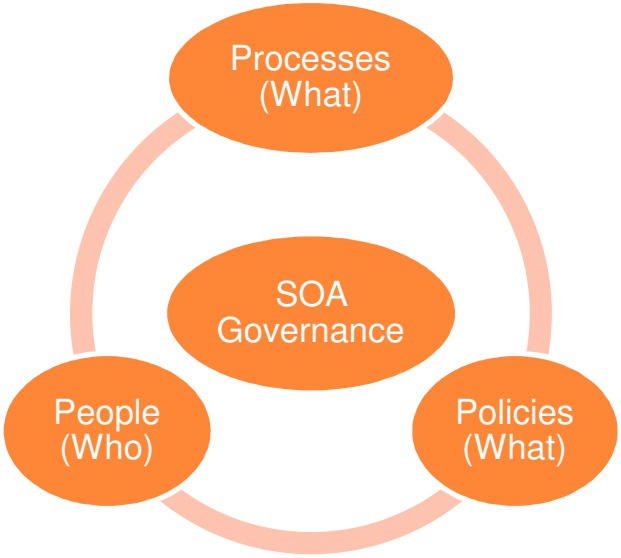
SOA Governance Objectives

- SOA governance
  - ensures that the concepts and principles of service orientation and distributed architecture are
    - Managed appropriately
    - Deliver on stated business goals
  - controls the evolution of service-oriented systems

- Most papers dealing with SOA governance methodologies and best practices concentrate on how to deliver and satisfy business objectives and often do not cover maintenance and evolution challenges systematically.

CHARACTERIZING SOA GOVERNANCE

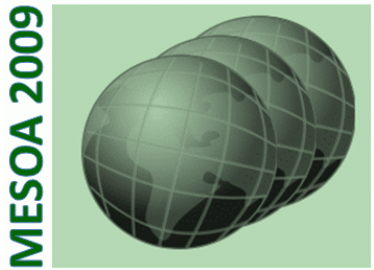
- Developers of service-oriented infrastructure have identified **people, policies, processes** and **technology** as the key governance pillars
- These pillars are used to control, manage, and maintain distributed systems of services and resources to optimize business and evolution objectives
- Services are
  - Guided by policies
  - Managed by people
  - Implemented by processes



SOA Governance Pillars

- **People**
  - Establish chains of responsibilities & decision rights
  - Establish control, policy, and process mechanisms to facilitate people to carry out their roles and responsibilities
  - Measure results and provide feedback to strategists
  - Orchestrate and execute process steps by adhering to, observing, and optimizing specified policies and procedures
- **Policies**
  - Created at design-time and adapted at run-time—manually by people or automatically by processes,
  - Higher level policies define the rules for lower level policy adaptations to optimize business and evolution
- **Processes**
  - Facilitates orchestration of service-oriented business and the evolution of its applications and infrastructure
  - SOA governance comprises a set of well-defined processes, which enact policies and are executed by people and technology

SOA Governance People Roles

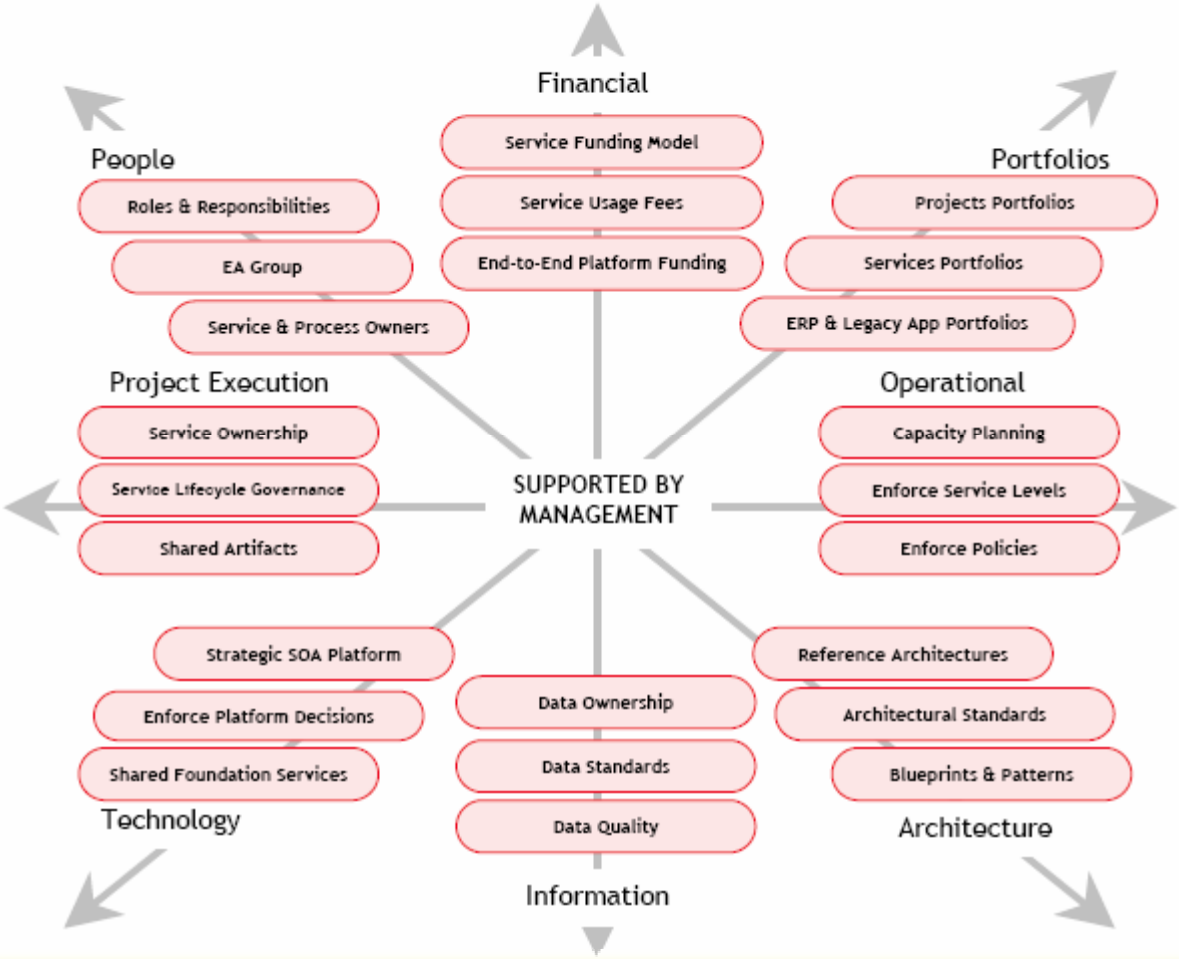


– Responsible

	Senior Level IT Steering Committee	Program Management Office	Enterprise Architecture Group	Integration Center of Competency	Services Development Group
Vision, Strategy, Priorities	■				
Business Services Portfolio Definition	■				
Services and Implementation Life Cycles	■				
Service Policies	■				
Services Funding Management		■			
Portfolio Planning and Management		■			
Schedule, Staff, and Manage Projects		■			
Change Management		■			
SOA Strategy and EnterpriseArchitecture			■		
Infrastructure Services Portfolio Definition			■		
IntegrationArch and EIF[1]				■	
Services Librarians					■

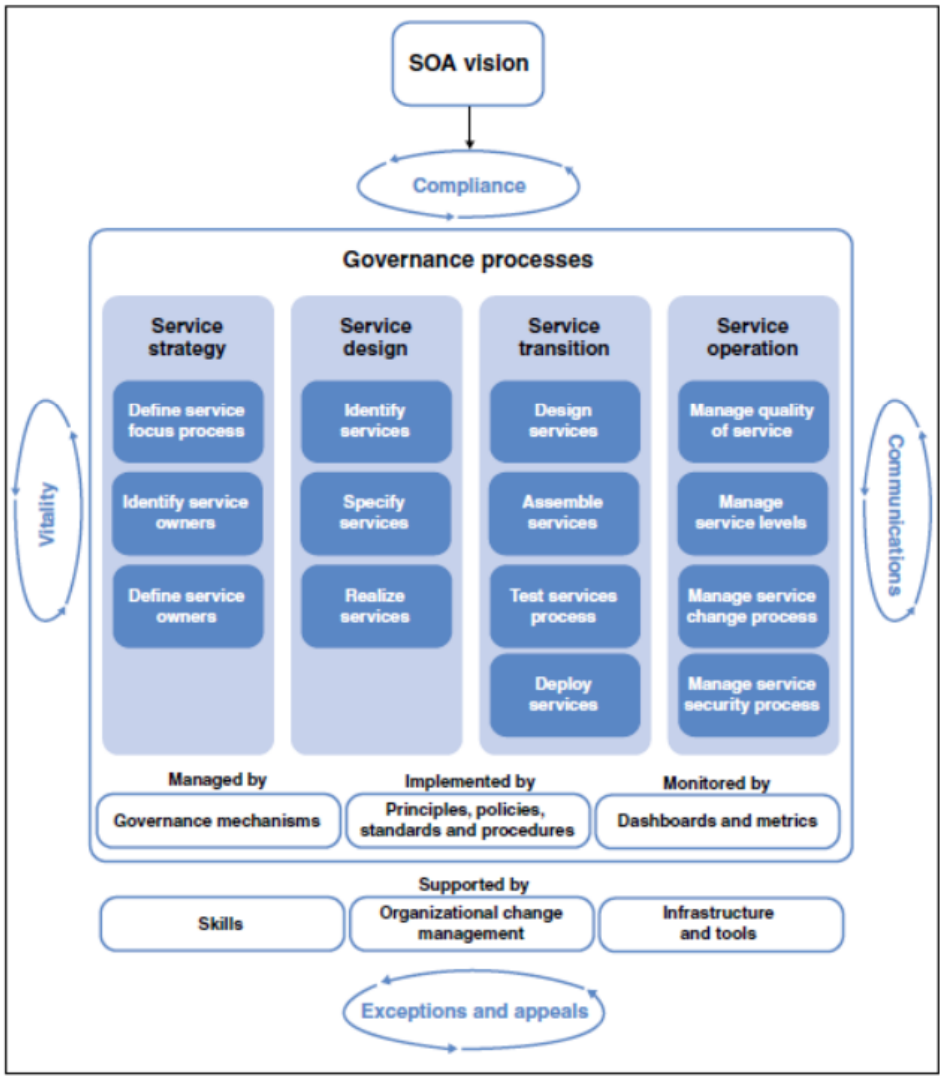
TIBCO: SOA Governance Best Practices, 2005

SOA Governance Policies



M. Afshar: SOA Governance: Framework and Best Practices, Oracle White Paper, May 2007

SOA GOVERNANCE PROCESSES



B. Brown
Introduction to SOA Governance and Service Lifecycle Management: Best practices for Development and Deployment,
IBM SOA White Paper, March 2009.

IBM Global Services Service Integration Maturity Model (SIMM)

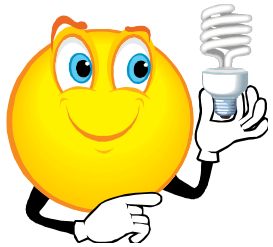
- Ultimate goal: dynamically configurable services

Service Integration Maturity Model	Silo	Integrated	Components	Service Integration	Composite Services	Virtualized Services	Dynamically Configurable Services
Business View of IT	Function Oriented	Function Oriented	Function Oriented	Service Oriented	Service Oriented	Service Oriented	Service Oriented
Methods	Structured	Object-Oriented	Component Based Development	Service Modeling	Service Modeling	Service Modeling	Grammar Modeling
Applications	Modules	Modules	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Application Assembly
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re Configurable Architecture
Infrastructure	Platform Specific	Platform Specific	Platform Specific	Platform Specific	Platform Specific	Platform Neutral	Dynamic Sense & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

http://www.ibm.com/developerworks/webservices/library/ws-soa-simm/

Traditional Evolution Mechanisms Are At The Core

- Implementations of SOA governance methodologies often embody classic software evolution concepts
  - **Levels of indirection**
  - **Feedback loops**

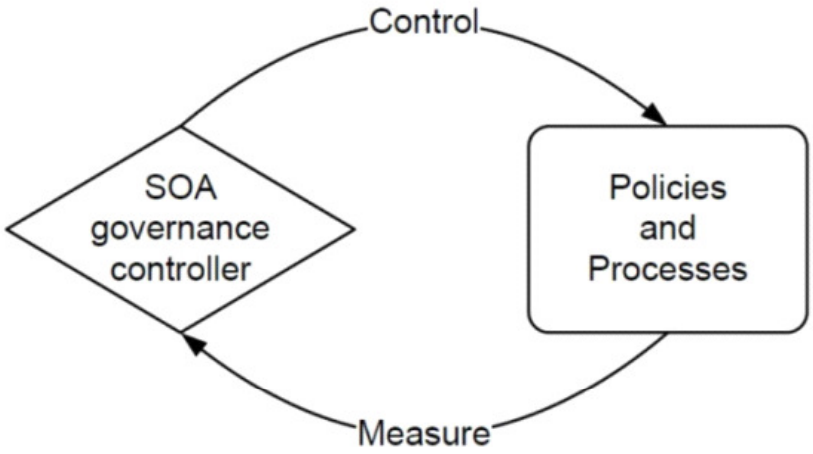


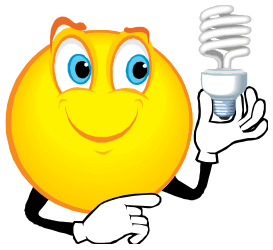
- To ensure that the concepts and principles of service orientation are managed appropriately and deliver on stated business goals
- To control the evolution of service-oriented systems

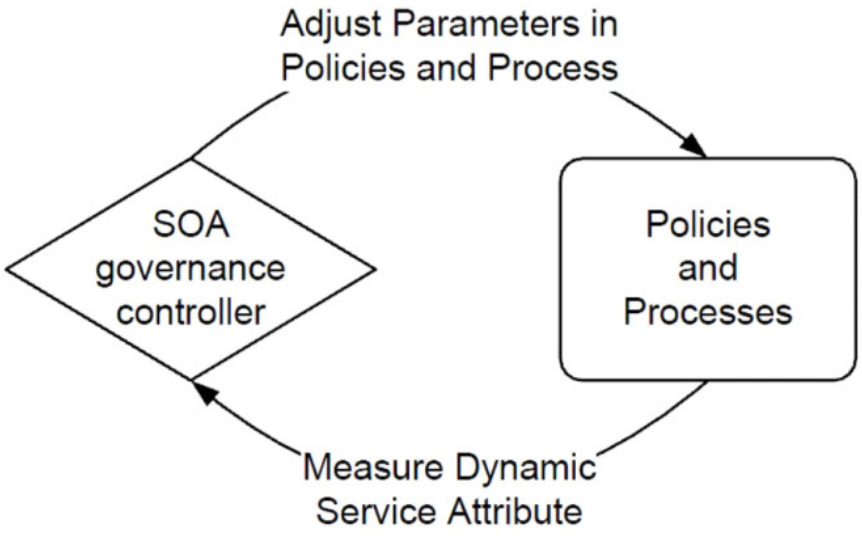

Evolving Software Systems
SOA Systems
Control Systems
Autonomic Systems
ULS Systems
What do these systems have in common?


Evolving Software Systems
SOA Systems
Control Systems
Autonomic Systems
ULS Systems
Self-managing Systems
Feedback Loops

SOA Governance Feedback Loop







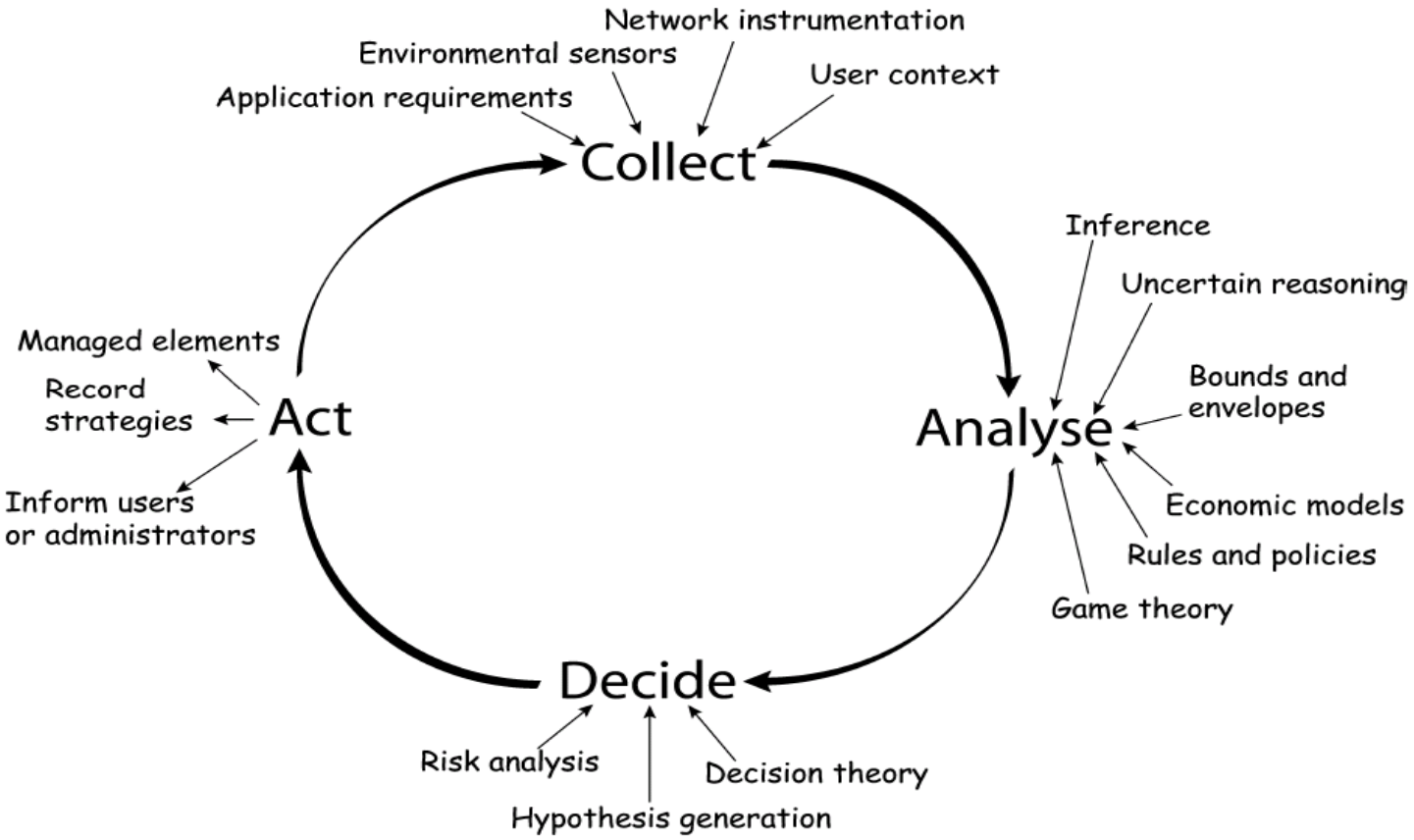
SOA Process Measures To Optimize Business & Evolution Objectives

- Run-time monitoring of SOA processes
- Gathering key measures
  - Execution time
  - Availability
  - Throughput
  - Latency
  - Resource consumption
- Controlling dynamic service selection
- Controlling the evolution of
  - Services
  - Metadata
  - Applications

Manage SOA Properties Using Feedback Loops

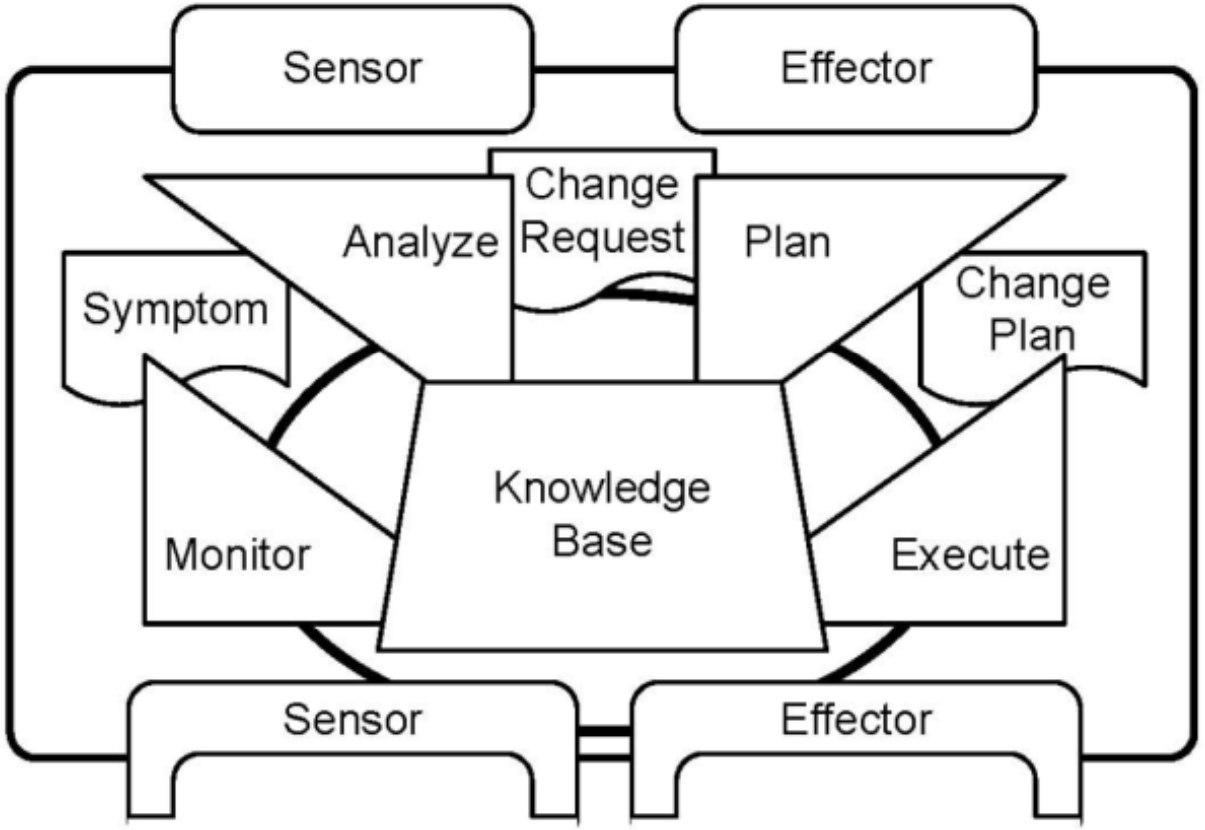
- Control compliance and conformance
- Perform critical regression tests dynamically to observe satisfaction of requirements
  - Testing run-time (and design-time) governance
  - Govern and enforce rules and regulations
- Perform V&V operations regularly to ascertain V&V properties
  - Monitor compliance and conformance
  - Assess whether services are used properly
  - Recognizing normal and exceptional behaviour
- Monitor functional & non-functional requirements when environment evolves
  - SLAs
  - Assess and maintain quality of service (QoS)
  - Manage tradeoffs
- Conformance: check how well a given implementation matches a reference
  - A conformance testing suite that returns results that certain aspects of an implementation match a reference implementation
  - A Web services implementation can conform with the WS-I basic interoperability profile
  - Service delivery is conformant with an SLA depends on the importance of the customer

AUTONOMIC FEEDBACK LOOP



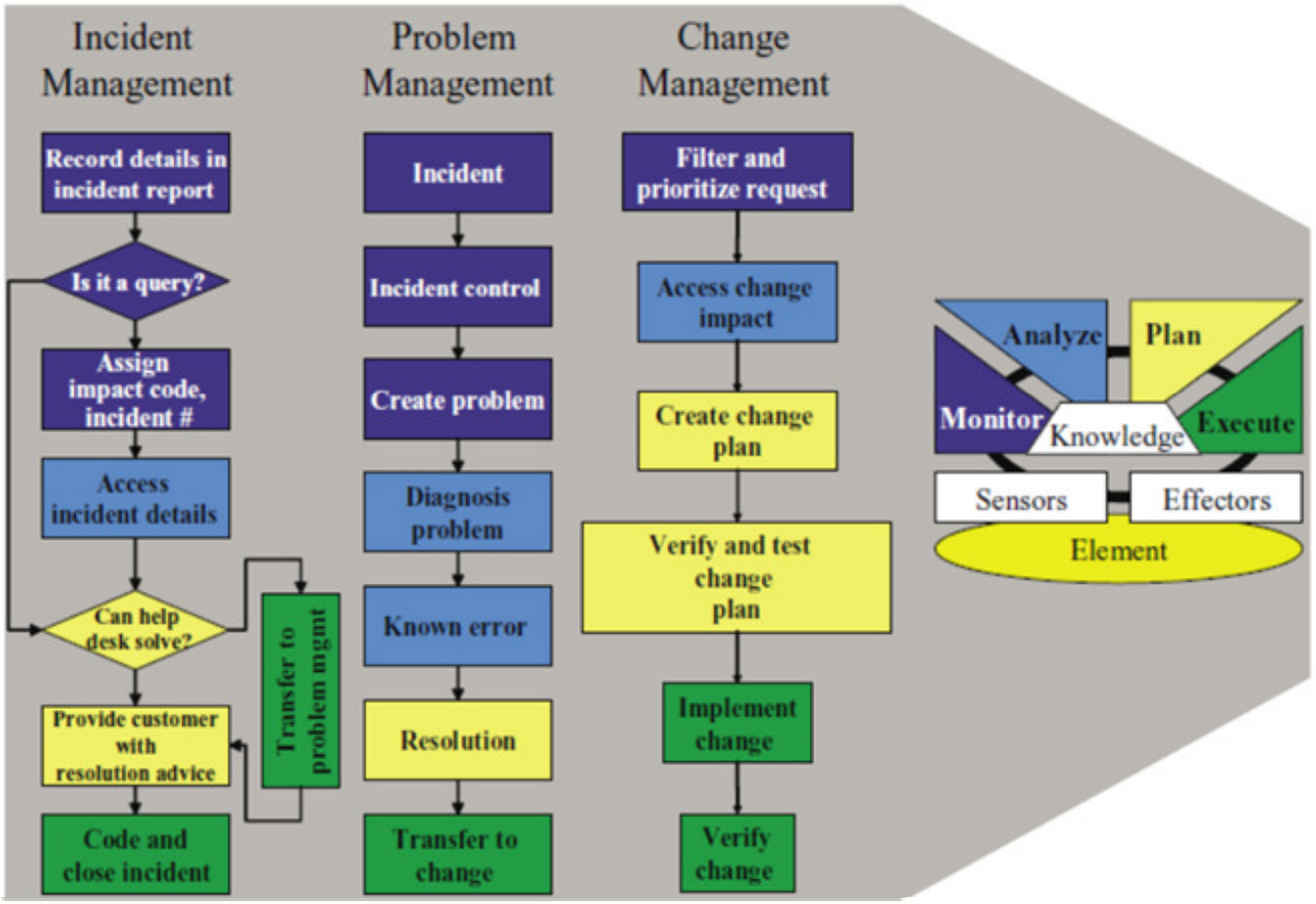
Dobson, S. et al.: A Survey of Autonomic Communications. ACM Trans. on Autonomous and Adaptive Systems (TAAS) 1(2):223-259 (2006)

AUTONOMIC ELEMENT



IBM: An Architectural Blueprint for Autonomic Computing, IBM White Paper 4th Ed., June 2006.

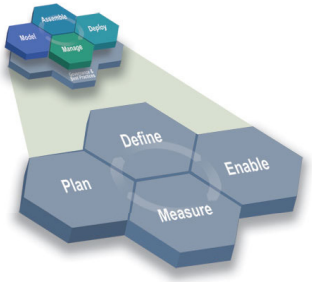
ITSM PROCESSES AS FEEDBACK LOOPS

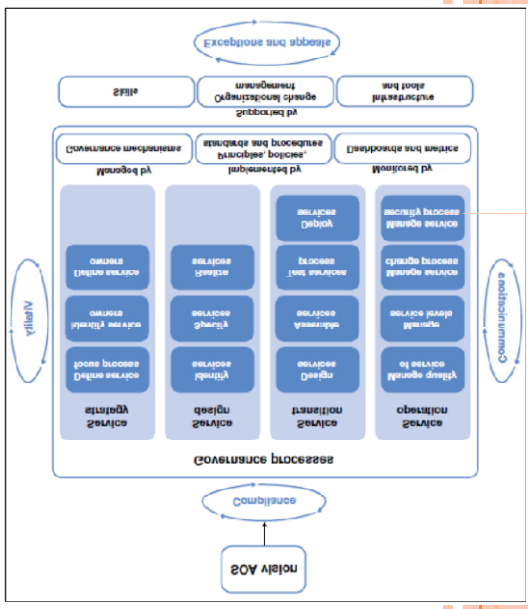


IBM: An Architectural Blueprint for Autonomic Computing, IBM White Paper 4th Ed., June 2006.

Levels Of Indirection

- Another mechanism to manage flexibility is to introduce levels of indirection
- IBM governance process model constitutes a level of indirection

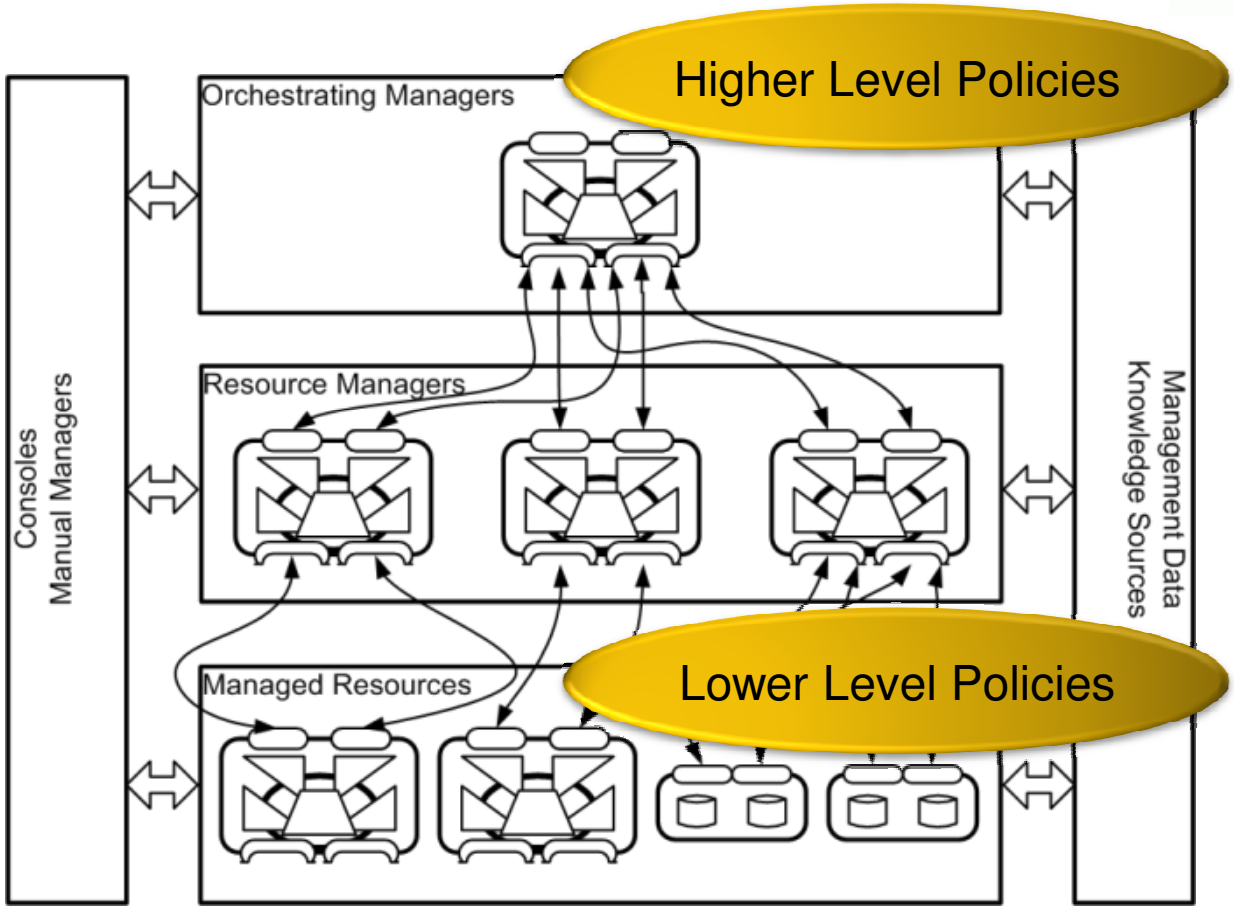




- Levels of indirection can be designed for
  - IBM's SGMM key process components
  - Oracle's policy leverage points
- Organize policies into layers as in IBM's Autonomic Reference Architecture (ACRA)

Any problem in computer science can be solved with another layer of indirection. —David Wheeler

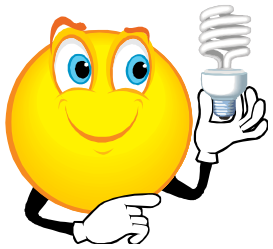
ACRA To Orchestrate SOA Governance Policies & Processes

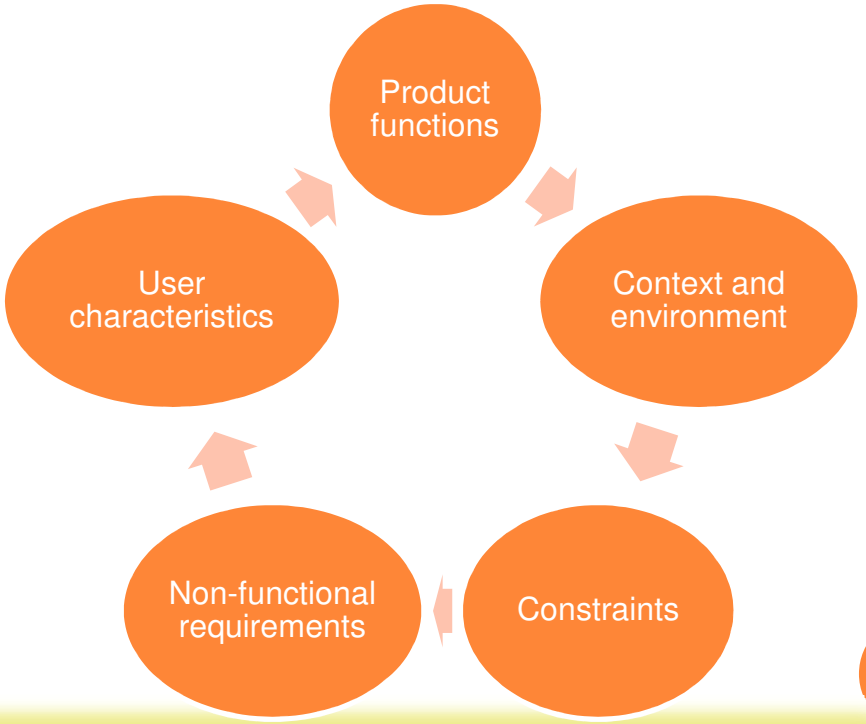


ACRA: Autonomic Reference Architecture

CO-SOFTWARE EVOLUTION

- Do we concentrate too much on the code?
- Are we too reactive and not enough pro-active?
  - Can we apply and inject our evolution experience into new development paradigms (e.g., service oriented systems)?
- Is co-evolution harder for dynamical systems?





Co-evolve requirements, context, constraints, policies, code

Research Challenges

- Model construction
- Managing and leveraging uncertainty
- Making control loops explicit

Research Challenges: Model construction

- The process of designing feedback-based computing systems requires the construction of models which quantify the effect of control inputs on measured outputs
  - While performance engineering and queuing theory have developed advanced models for many different applications, we need models for many other quality-of-service indicators that come into play in dynamical applications
  - For some of these criteria (e.g., trust) quantification is difficult
  - What system-wide, end-to-end, and local quality-of-service indicators are relevant to meeting user needs?
- Models are also needed to design trade-off analyses schemes for combinations of quality-of-service indicators
- Developing feedback models for quality-of-service indicators for various application domains is a major challenge
  - Models and quality-of-service indicators related to governance, compliance, and service-level agreements are of particular importance for service-oriented business processes and applications

RESEARCH CHALLENGES: MANAGING & LEVERAGING UNCERTAINTY

- When we model potential disturbances from the environment of an SOA system (e.g., unexpected saturation of the network) or satisfy requirements by regulation (i.e., trade-off analysis among several extra-functional requirements), we introduce some uncertainty
- Therefore, designers and maintainers of such dynamical systems should manage uncertainty because the environment may change in unexpected ways and, as a result, the system may adapt in such a way that was not foreseeable at design time

- Introducing uncertainty requires trade-offs between flexibility and assurance
- For a maintainer it is critical to know which parts of the environment are assumed to be fixed and which are expected to introduce uncertainty
- Moreover, assurance and compliance criteria should be continuously validated at run-time—not just at system acceptance time

- Thus, understanding, managing, and leveraging uncertainty is important for delivering SOA systems with reliability and assurance guarantees

Research Challenges: Making Control loops explicit

- Investigate architecture-centric vs. control-centric design and run-time views for SOA systems
- Software engineers are trained to develop abstractions that hide complexity
- Designers of SOA systems will likely realize significant benefits by raising the visibility of control loops and specifying the major components and characteristics of the control loops explicitly
  - When arrangements of multiple control loops interact, system design and analysis should cover their interactions
  - As control grows more complex, it is important for the control loops to be explicit in design and analysis
- Investigate the trade-offs between hiding the complexity of feedback loops and treating feedback loops as first class objects with respect to the construction and operation of SOA systems
- Further benefits could be realized by identifying common forms of adaptation and then distilling design and V&V obligations

CONCLUSIONS

- "Introduced" two SOA governance mechanisms
  - Feedback loops
  - Levels of indirection
- These mechanisms are equally applicable to optimize business and evolution of service-oriented systems
- IT vs. SOA governance
- Is maintenance for service-oriented systems easier than for traditional information systems?
- Is software co-evolution harder for service-oriented systems easier than for traditional information systems?

To Probe Further

- Hellerstein, J.L., Diao, Y., Parekh, S., Tilbury, D.M.: Feedback Control of Computing Systems. John Wiley & Sons (2004)
- P. Brittenham, R.R. Cutlip, C. Draper, B.A. Miller, S. Choudhary, M. Perazolo: IT Service Management Architecture and Autonomic Computing, IBM Systems Journal 46(3):565-581, 2007.
- Mills, S.: The Future of Business, White Paper (2007)
- Dobson, S., Denazis, S., Fernandez, A., Gaiti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., Zambonelli, F.: A Survey of Autonomic Communications. ACM TAAS 1(2):223-259 (2006)
- Lewis, G., Smith, D.: SOA and its Implications for Software Maintenance and Evolution, ICSM FoSM (2008)
- Bianco, P., Lewis, G.A., Merson, P.: Service Level Agreements in Service-Oriented Architecture Environments, SEI CMU/SEI-2008-TN-021, Sep. 2008.
- Müller, H.A., Pezzè, M., Shaw, M.: Visibility of Control in Adaptive System. In: 3rd ACM/IEEE International ICSE ULSSIS 2008, pp. 23-26 (2008)
- Müller, H.A., H.M. Kienle, H.M. Stege, U.: Autonomic Computing: Design and Evolution of Autonomic Software Systems, *International Summer School on Software Engineering (ISSE) 2006–2008,* LNCS 5413, A. De Lucia and F. Ferrucci (Eds.), Springer-Verlag Berlin Heidelberg, pp. 32–54, 2009.

Questions, Feedback, Comments, Ideas, Aha-experiences, Insights, ...