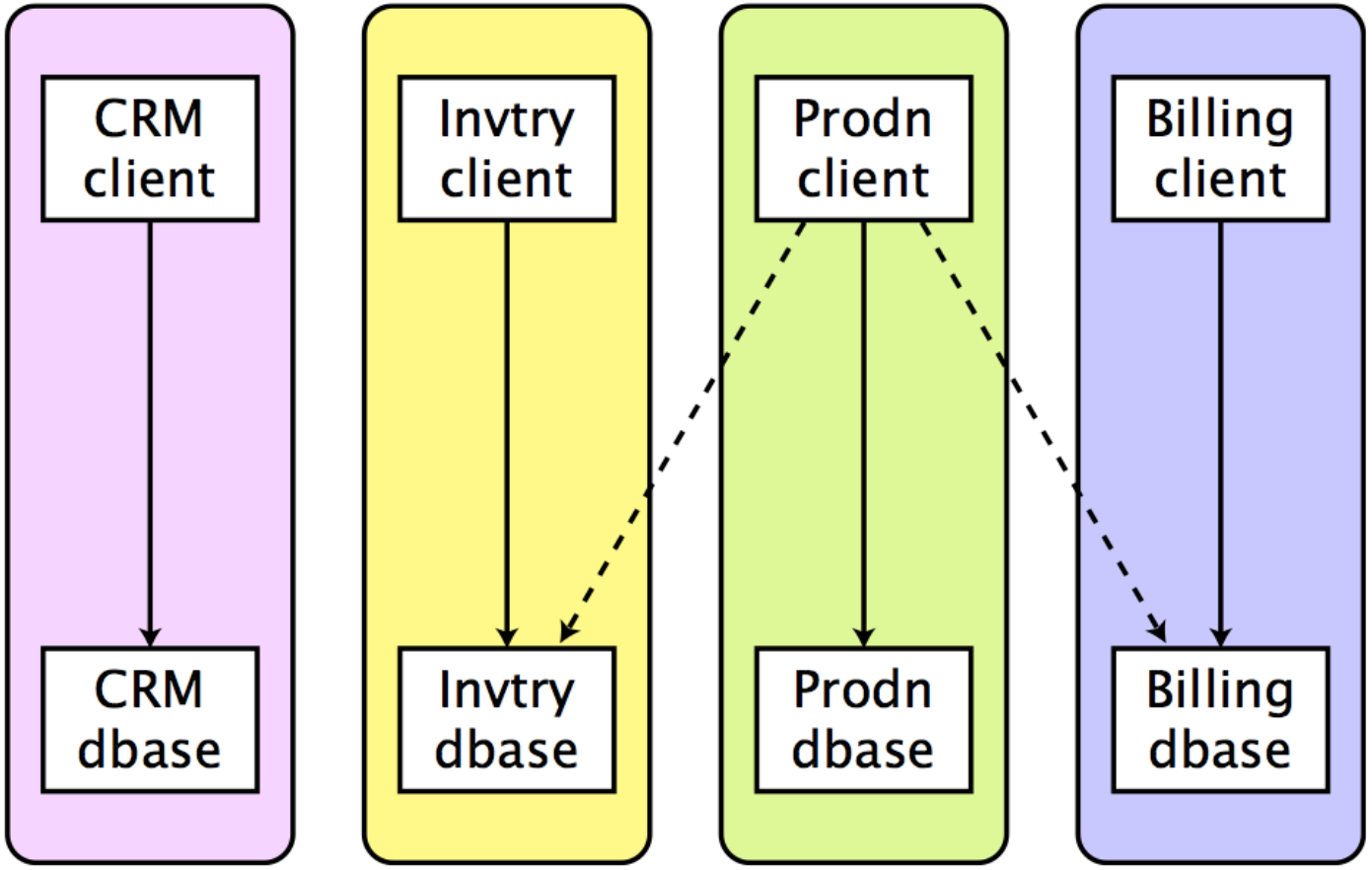
Governance, Registries and Monitoring

Oxford University
Software Engineering
Programme
January 2018

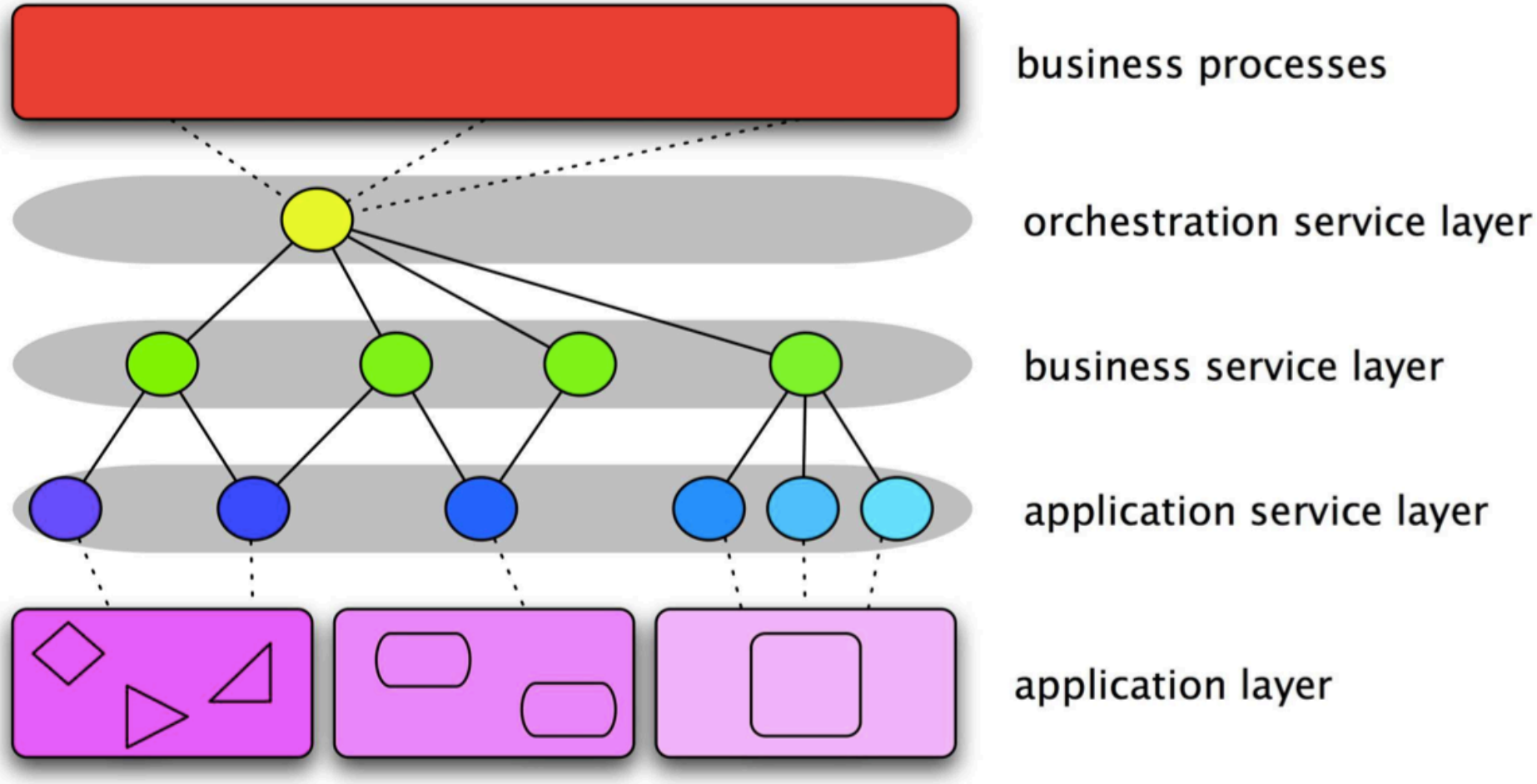
Contents

- Software Development Lifecycle
- Registries
- Design Governance
- Runtime Governance

Before SOA



With SOA



SOA has an impact on organization

- Refactoring of fiefdoms:
  - backend departments
  - cross-domain departments - frontend departments
  - "solutions managers"
- Requires collaboration and trust
- May change the funding model
  - That will pull in resistance

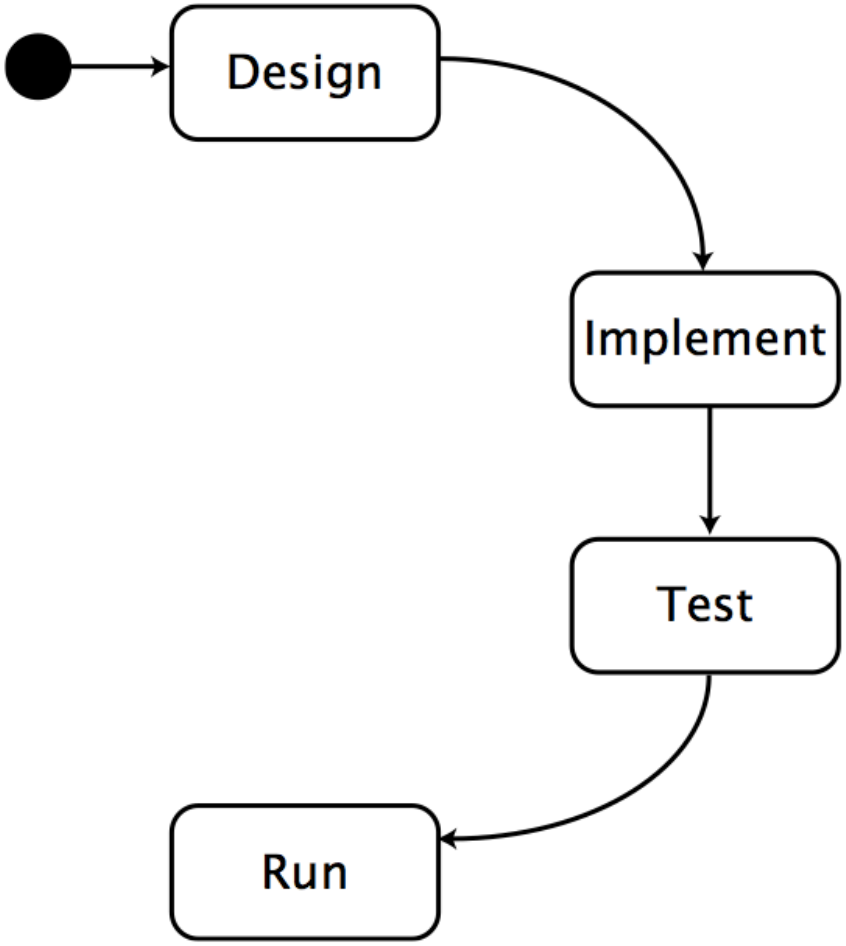
Conway's Law

- Any organization that designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure.

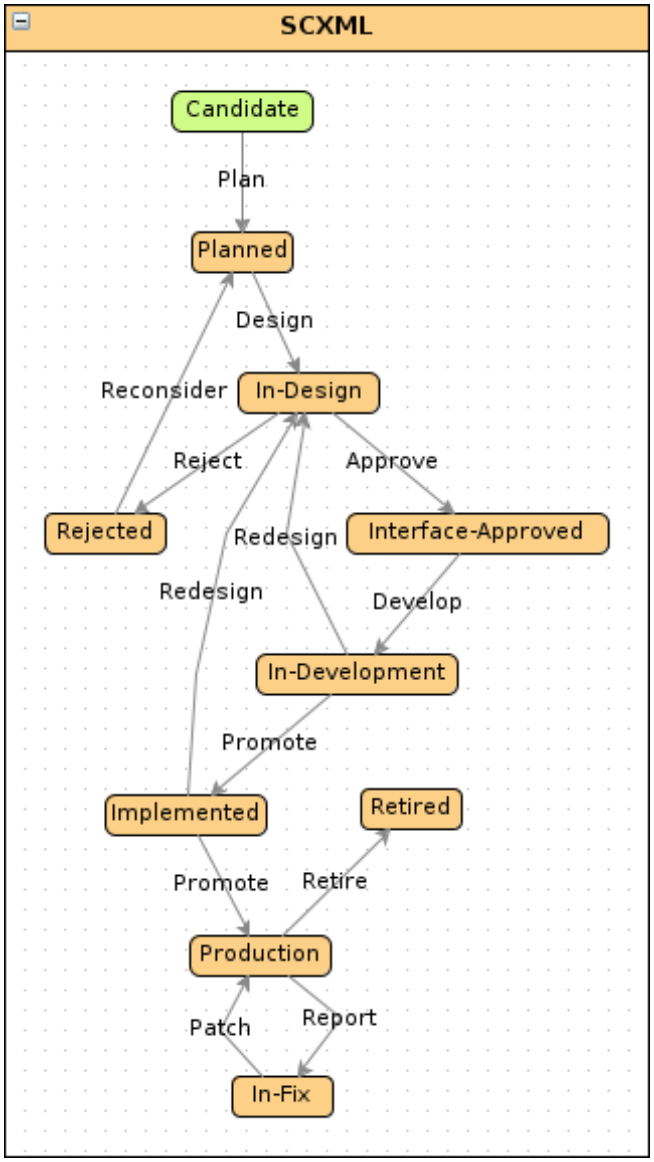
  Melvin Conway, *How Do Committees Invent?*, Datamation Apr 1968, http://www.melconway.com/law/

- Popularized and named by Fred Brooks in *The Mythical Man-Month*: "If you have four groups working on a compiler, you'll get a 4-pass compiler."

Software Development Lifecycle



Not that simple!



High level governance

- *Visions, objectives, business case, funding model*
  - Why are we doing this? How will we pay for it?
  - *Reference architecture*
    Fundamental decisions: preferred technology, message exchange patterns, metamodel, etc
- *Rules and responsibilities*
  - who drives and cares about issues
- *Policies, standards, formats, processes, lifecycles*
  - decide and document, in standard notations

Technical Governance

- *Documentation*
  - important for transparency; promotes non-technical issues
- *Service management*
  - repositories and registries for services and contracts
- *Monitoring*
  - conformance to policies, meeting SLAs, preparing for withdrawal
- *Change and configuration management*
  - Code lifecycle, DevOps, SOA, the intersection

Establishing SOA

- *Developer-driven, grass-roots*
  - leads to technological experience; likely to be uncoordinated
- *Business-driven*
  - proof of concept helps adoption; limited benefit from early projects
- *IT-driven*
  - effective for infrastructure; focus on technical aspects
- *Management-driven*
  - top-down coordinated, driven by business priorities; expensive, disruptive, risky

Design Time Governance

Aspects of Governance Registries

- Meta-Modelling
- Taxonomies
- Versioning
- Associations and Properties
- Lifecycle Management
- Dependency Management
- Repository
- Search
- Machine and Human interfaces

Meta-Modelling

- Creating / modifying the model to support new artefacts
  - e.g. Teams, Projects, Organizations, etc
- Also used for extending more technical attributes
  - e.g. adding WADL or Swagger support

Versioning

- Micro-versioning/revision management
  - Keeping track of every minor update to a WSDL
  - Permanent URLs for given versions
- "Business" Versioning
  - Service A is available as
    - 1.2.3 deprecated
    - 2.5.1 current

Associations and Properties

- Properties
  - General name / value pairs attached to resources
- Associations
  - Named Links between resources
  - e.g.
    - A isUsedBy B
    - B isManagedBy C

Lifecycle Management

- Each service in the corporate datacentre MUST:
  - Start as “In Design”
  - Be approved by the Design Review Team
  - Iterate through Development
  - Pass validation tests before entering Staging
  - Be approved by the Security and Performance Teams before entering Production
  - Be deprecated when no longer supported

Dependency Management

- Each Service Sn depends on Schemas {Y1..n}
- Schema Y depends on Schemas {Z1..n}
- Schemas are shared between services
- Owners and users of services need to be made aware of new versions of schemas they depend on (even if they didn't know it!)

Interfaces

- Registries are used by humans, but shouldn't always be!
- e.g. Maven build rather than forcing developers to use a website
  - One company I know hires a "Registry Monkey" who ONLY enters services into a registry
  - Each service takes 83 steps
  - He hasn't yet committed suicide

Registry, DevOps, SCM

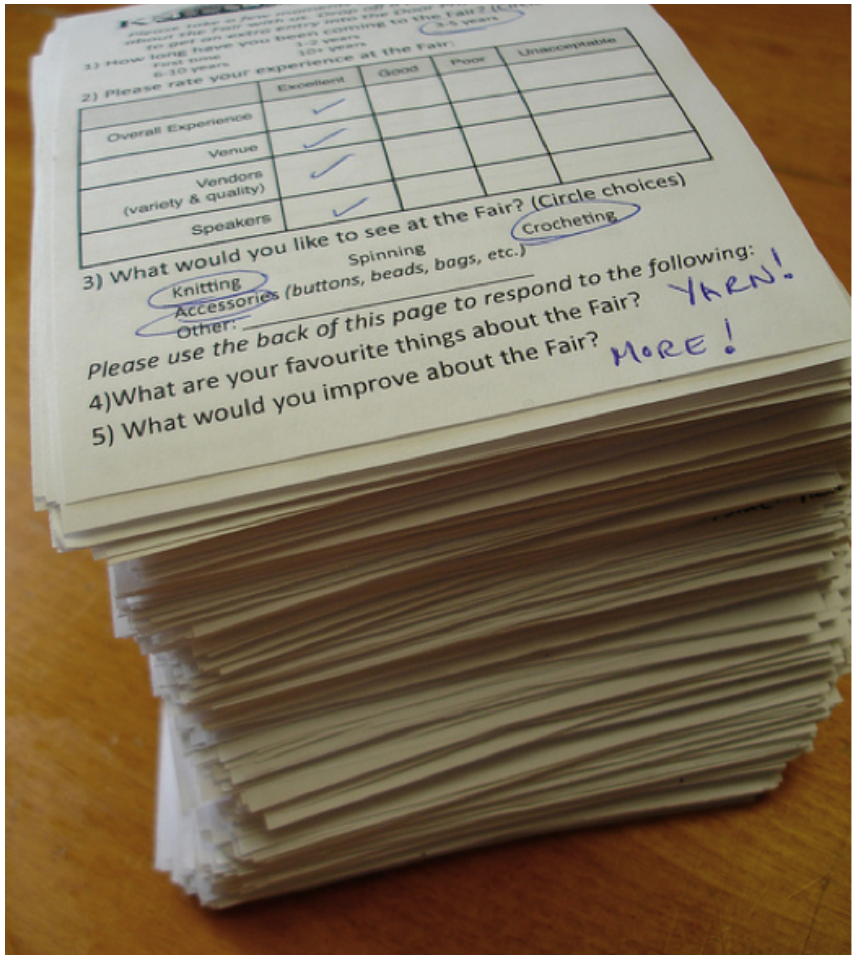
- Ideally need to connect:
  - The Source Code Management (CVS, SVN, Git)
  - The build and test environment
    - Hudson, Jenkins, Bamboo
    - Selenium, JUnit, etc
  - The production management process
    - DevOps, Puppet, Chef
  - The design time registry
  - The runtime registry

Runtime Governance

- Finding services at runtime
- Monitoring services at runtime
- Managing SLAs
- Correlation
- Acting on situations

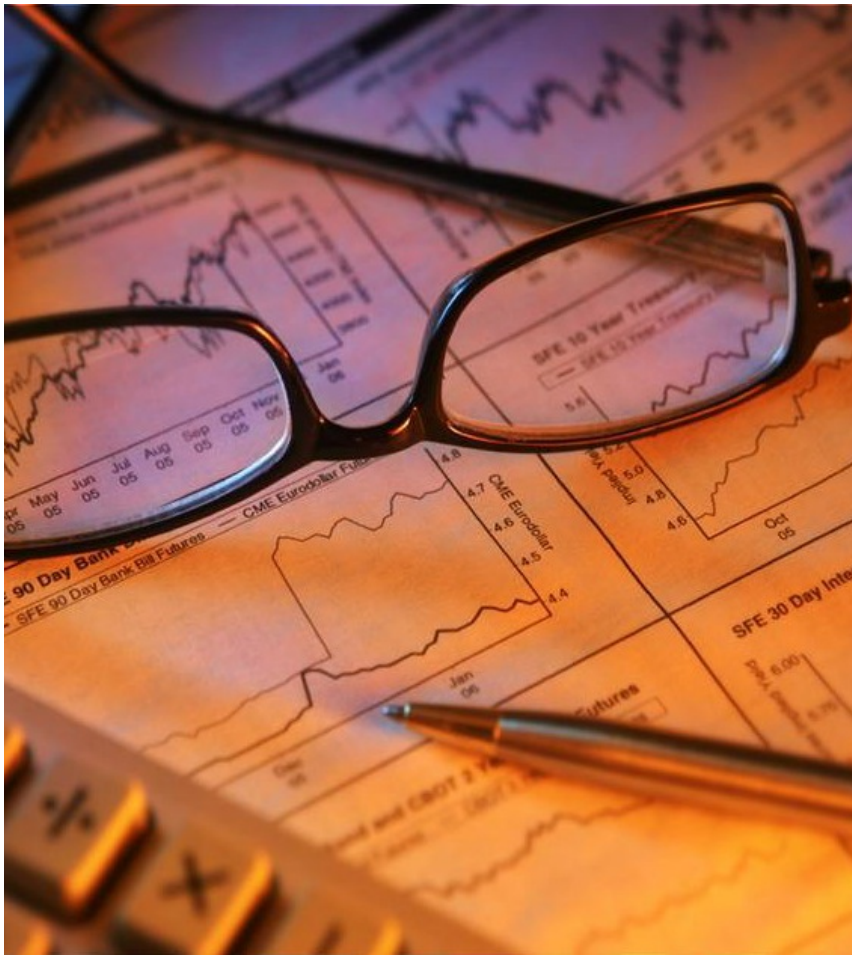
Aggregation

- Gathering data
- How to collect data efficiently
- How to store data effectively
- What data to capture



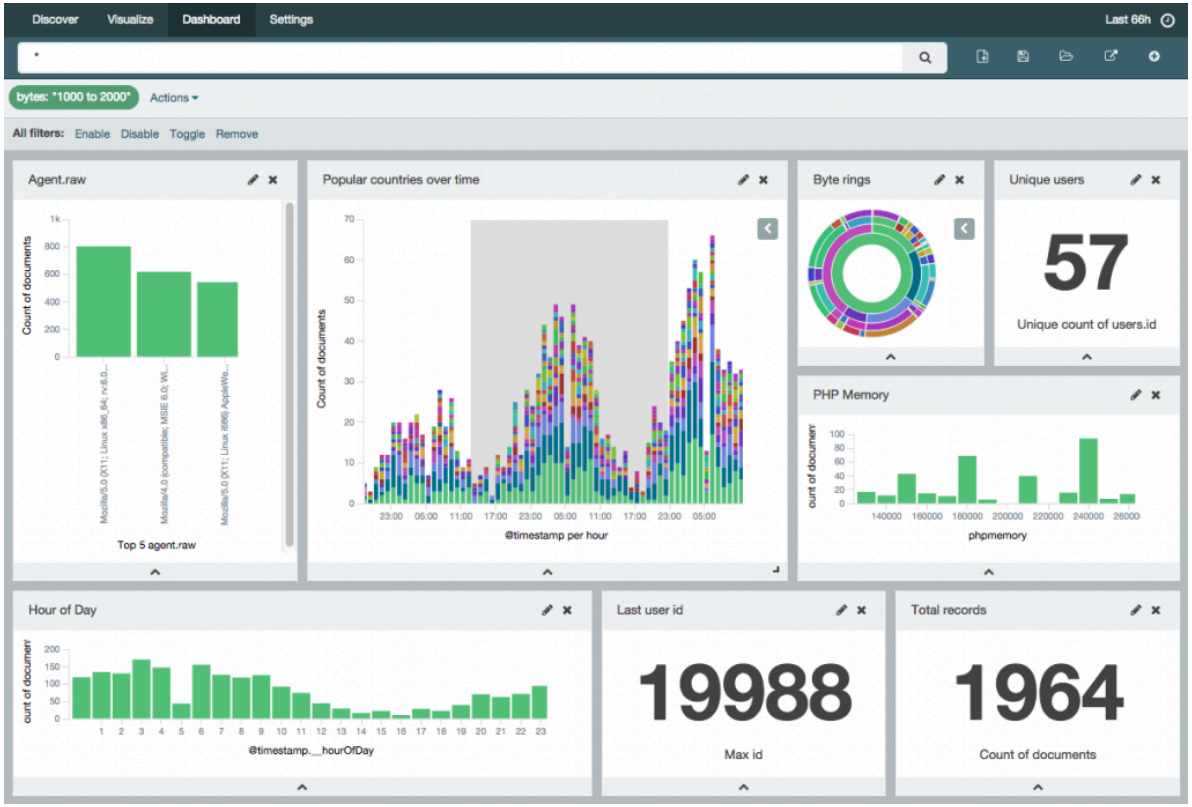
Analysis

- Data operations
- Defining KPIs and analytics
- Operating on large amounts of historical or current data
- Creating intelligence



Presentation

- Visualization
- Dashboards
- Reports


Discover
Visualize
Dashboard
Settings
Last 66h
bytes: "1000 to 2000"
Actions
All filters: Enable Disable Toggle Remove
Agent.raw
Count of documents
Top 5 agent.raw
Popular countries over time
Count of documents
@timestamp per hour
Byte rings
Unique users
57
Unique count of users.id
PHP Memory
phpmemory
Hour of Day
@timestamp._hourOfDay
Last user id
19988
Max id
Total records
1964
Count of documents

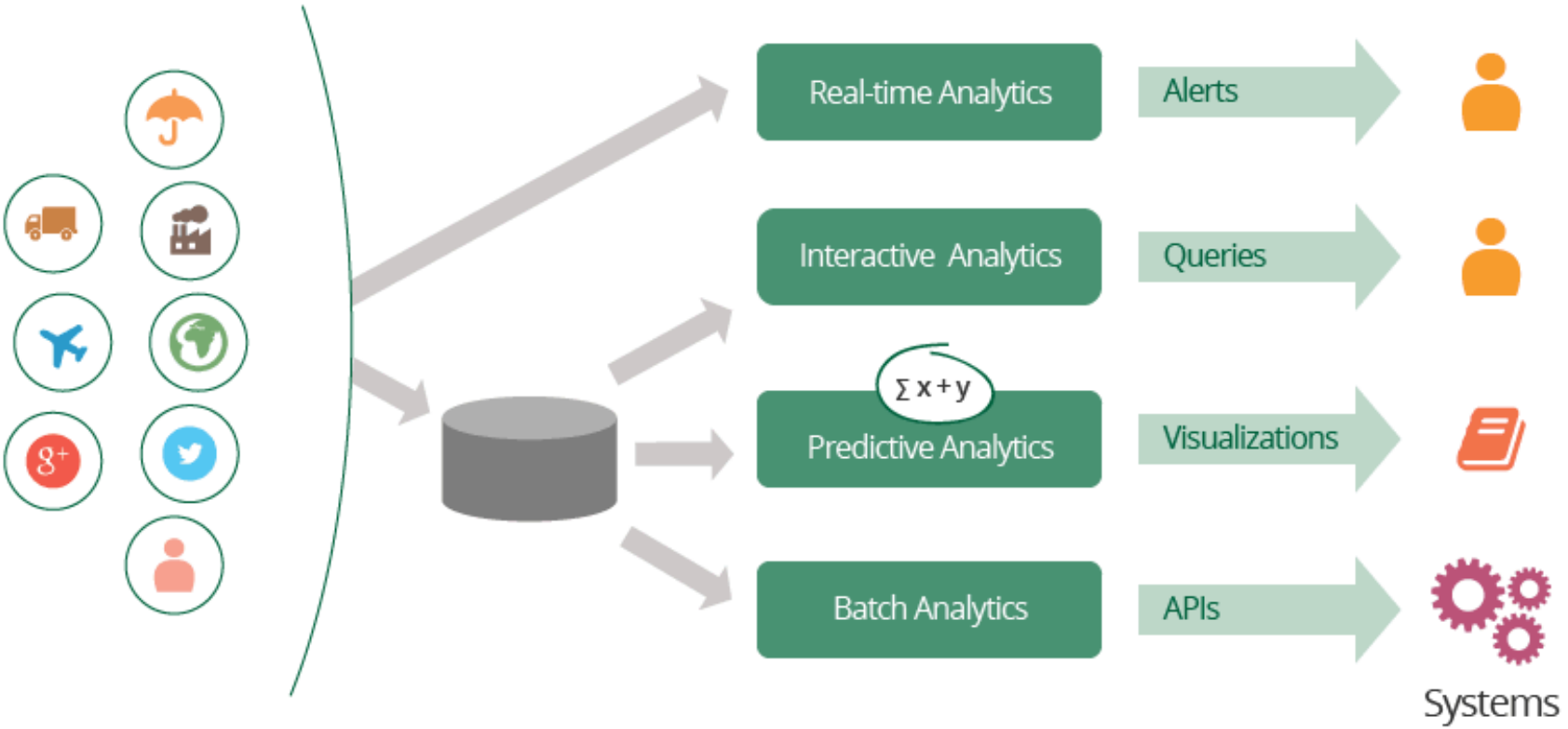

LOG


logstash


ELASTIC
SEARCH

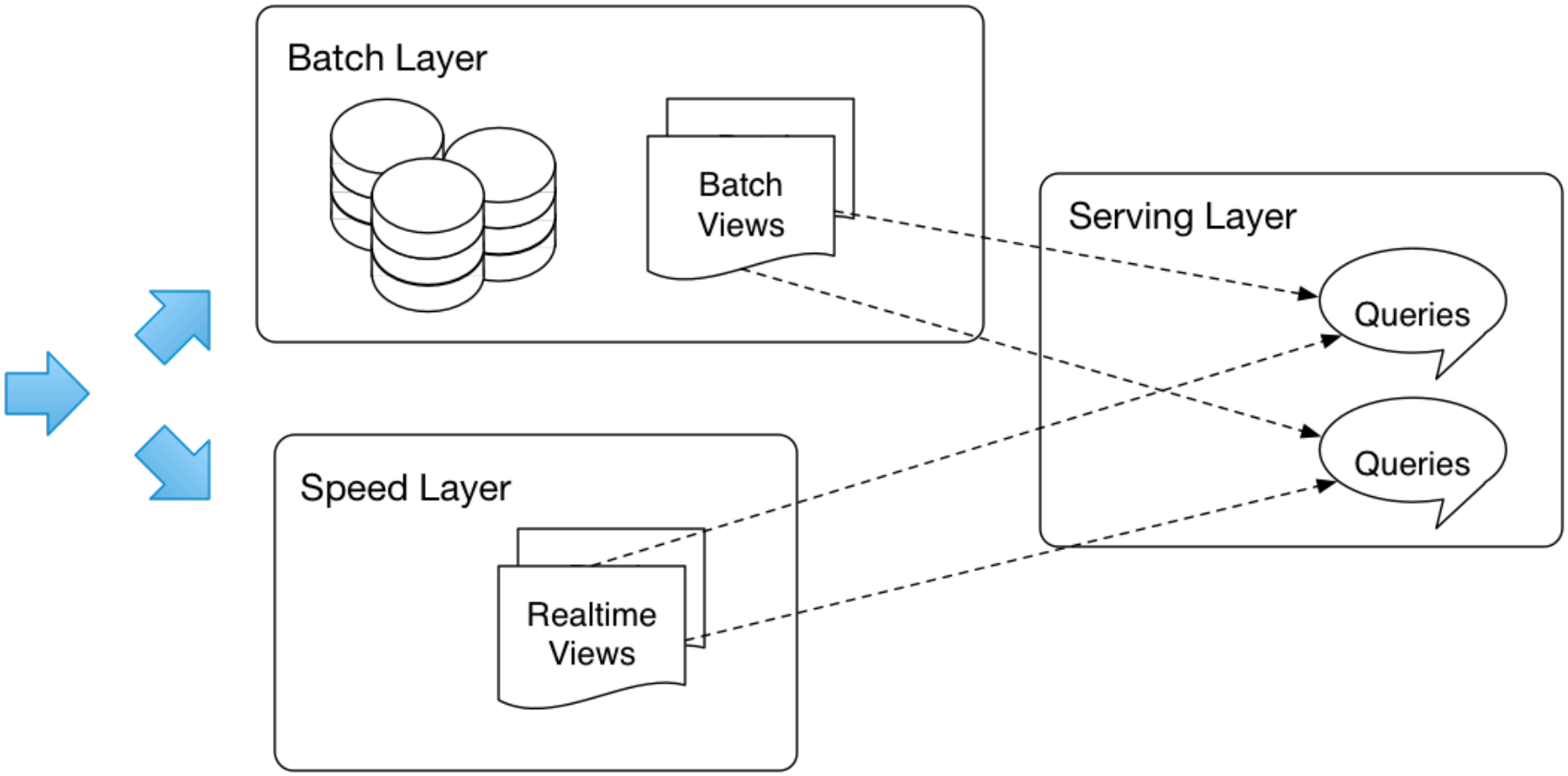

Kibana

WSO2 Data Analytics Server



Collect Data → Analyze & Make Decisions → Communicate

Lambda Architecture



Closing the loop

- SLAs are time based rules about performance data
  - Is service X responding in under 50ms for more than 99% of calls within the last 5 mins?
  - Does the sales team respond to leads within 4 hours?
  - Has the average CPU utilization over the last day gone more than 50% higher than the weekly average

Governance today

- Many organizations have moved away from Governance registries to API Management
- Even internally
- Why?
  - Better encapsulation
  - Handles many of the governance requirements
    - Design time versioning, documentation, understanding your users
    - Runtime monitoring, analytics

Questions?