MONITORING OF CLOUD/SERVICE-BASED SYSTEMS

Luciano Baresi
luciano.baresi@polimi.it

LUCIANO BARESI

Professor @ DEIB – Politecnico di Milano

- Researcher at Cefriel
- Visiting researcher
  - University of Oregon (USA)
  - University of Paderborn (Germany)

Research interests

- Software engineering
  - Dynamic software architectures
  - Service-oriented applications
  - Mobile applications
  - Cloud computing

home.deib.polimi.it/baresi

SOA: a couple of definitions (IBM)

"A service-oriented architecture (SOA) is an application framework that takes everyday business applications and breaks them down into individual business functions and processes, called services. An SOA lets you build, deploy and integrate these services independent of applications and the computing platforms on which they run."

A Service-Oriented Architecture is an **enterprise-scale IT architecture for linking resources on demand**. These resources are represented as **business-aligned services** which can participate and be composed in a value-net, enterprise, or line of business to fulfill business needs. The primary structuring element for SOA applications **is a service** as opposed to subsystems, systems, or components.

SOA is a business-driven IT architectural approach that supports integrating your business as linked, repeatable business tasks or services.

See also: "Service-Oriented Architecture and Enterprise Architecture, Part 1: A framework for understanding how SOA and Enterprise Architecture work together" (http://www-128.ibm.com/developerworks/webservices/library/ws-soa-enterprise1/)

SOA: multiple perspectives

A set of services that a business wants to expose to customers and clients

Business

An architectural style which requires a service provider, requestor and a service description.

A set of architectural principles, patterns and criteria, which address characteristics such as *modularity, encapsulation, loose coupling, separation of concerns, reuse, composable and single implementation*.

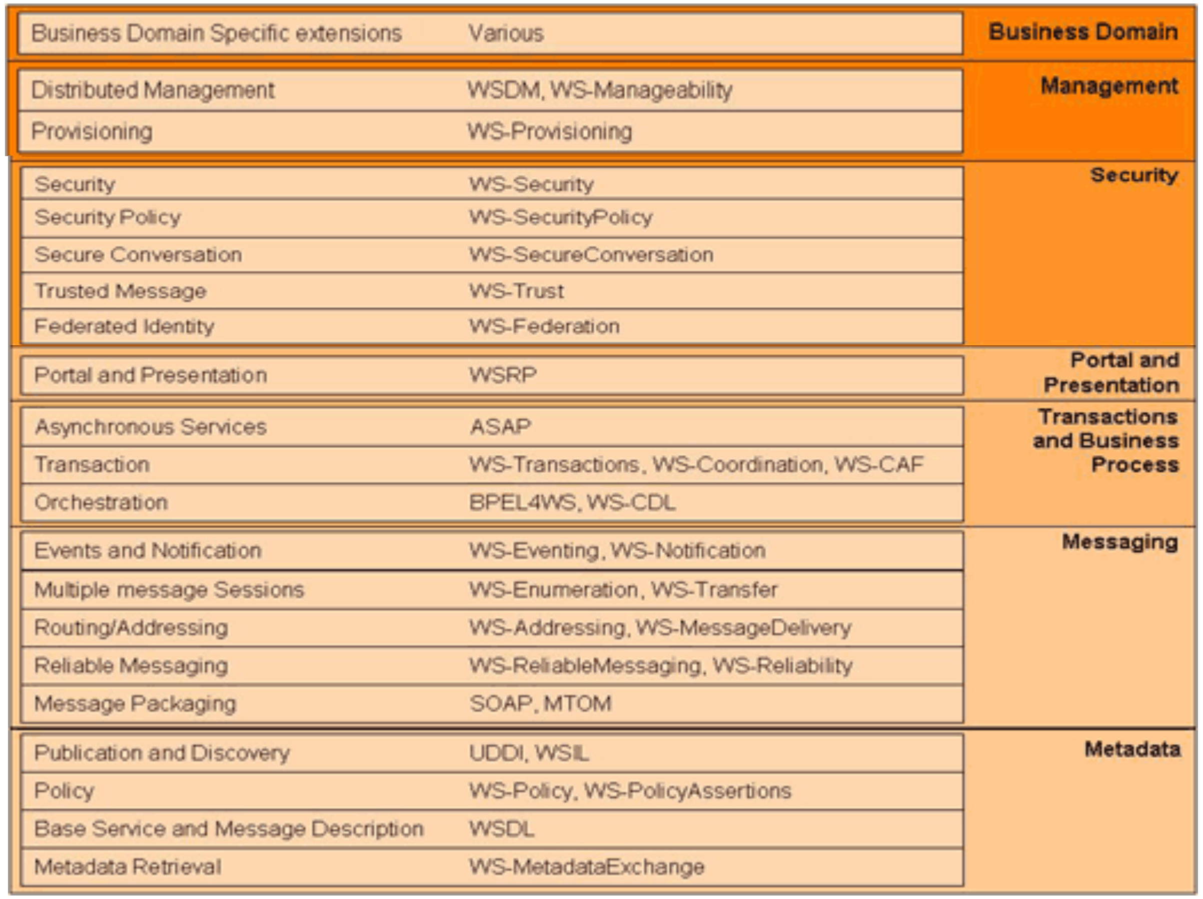
Architecture

A programming model complete with standards, tools, methods and technologies such as web services.

Implementation

SERVICE COMPOSITIONS

A BIT OF HISTORICAL PERSPECTIVE

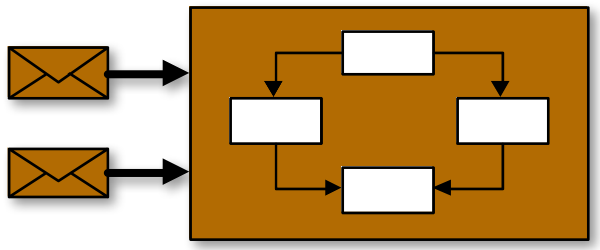


Business Domain Specific extensions	Various	Business Domain
Distributed Management	WSDM, WS-Manageability	Management
Provisioning	WS-Provisioning	
Security	WS-Security	Security
Security Policy	WS-SecurityPolicy	
Secure Conversation	WS-SecureConversation	
Trusted Message	WS-Trust	
Federated Identity	WS-Federation	
Portal and Presentation	WSRP	Portal and Presentation
Asynchronous Services	ASAP	Transactions and Business Process
Transaction	WS-Transactions, WS-Coordination, WS-CAF	
Orchestration	BPEL4WS, WS-CDL	
Events and Notification	WS-Eventing, WS-Notification	Messaging
Multiple message Sessions	WS-Enumeration, WS-Transfer	
Routing/Addressing	WS-Addressing, WS-MessageDelivery	
Reliable Messaging	WS-ReliableMessaging, WS-Reliability	
Message Packaging	SOAP, MTOM	
Publication and Discovery	UDDI, WSIL	Metadata
Policy	WS-Policy, WS-PolicyAssertions	
Base Service and Message Description	WSDL	
Metadata Retrieval	WS-MetadataExchange	

COMPOSITION MODELS

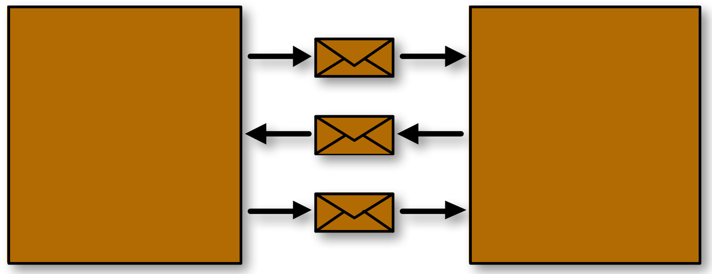
Orchestration

- Intra-process
- Process controlled by one party

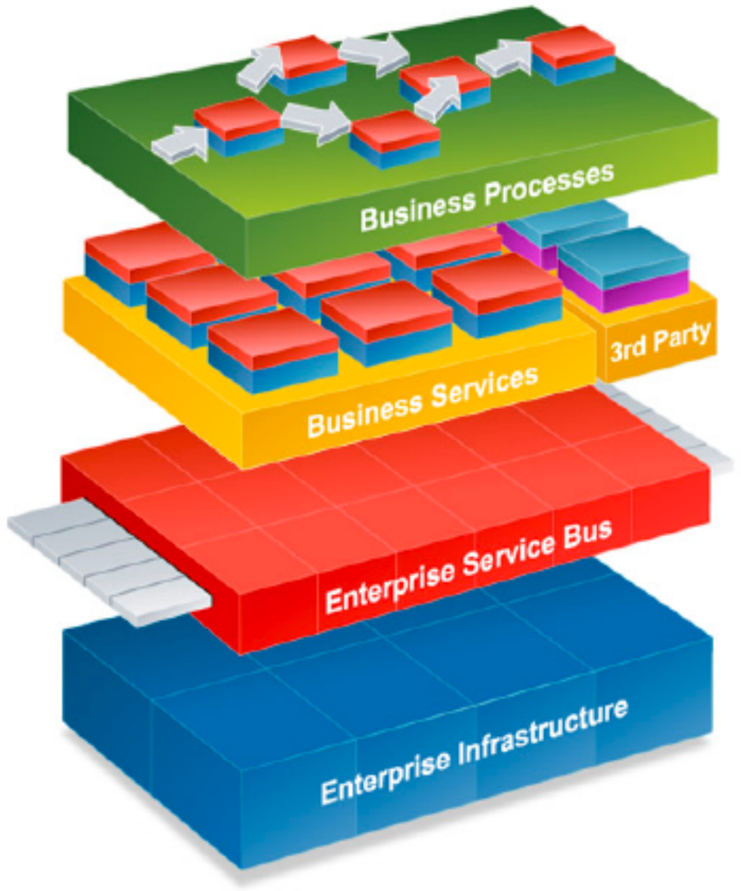


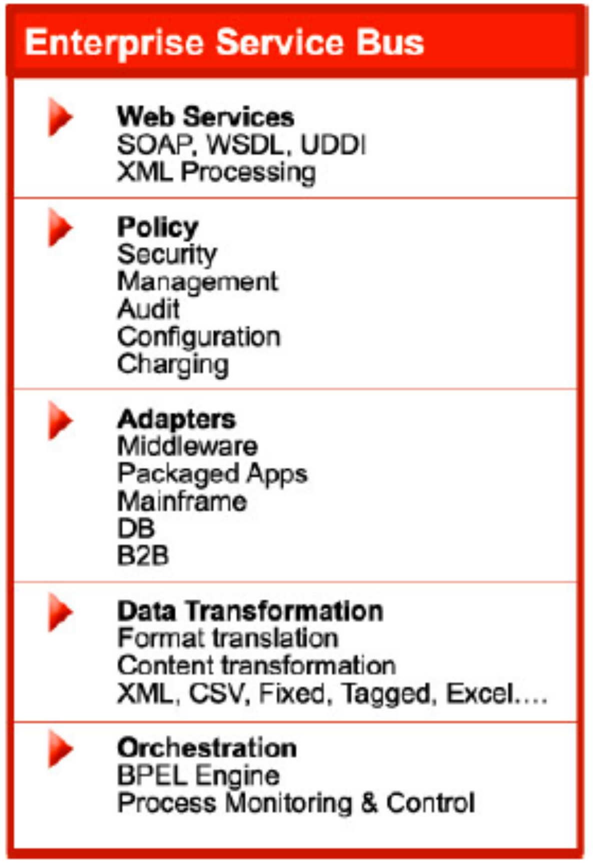
Choreography

- Inter-processes
- Sequence of observable messages
- Conversation among equals



SOA AND THE ESB STACK





Enterprise Service Bus

- **Web Services**
  SOAP, WSDL, UDDI
  XML Processing
- **Policy**
  Security
  Management
  Audit
  Configuration
  Charging
- **Adapters**
  Middleware
  Packaged Apps
  Mainframe
  DB
  B2B
- **Data Transformation**
  Format translation
  Content transformation
  XML, CSV, Fixed, Tagged, Excel….
- **Orchestration**
  BPEL Engine
  Process Monitoring & Control

DYNAMO

Design by contract

Separation of concerns

- Business logic defined separately from supervision
- Supervision is a cross-cutting concern

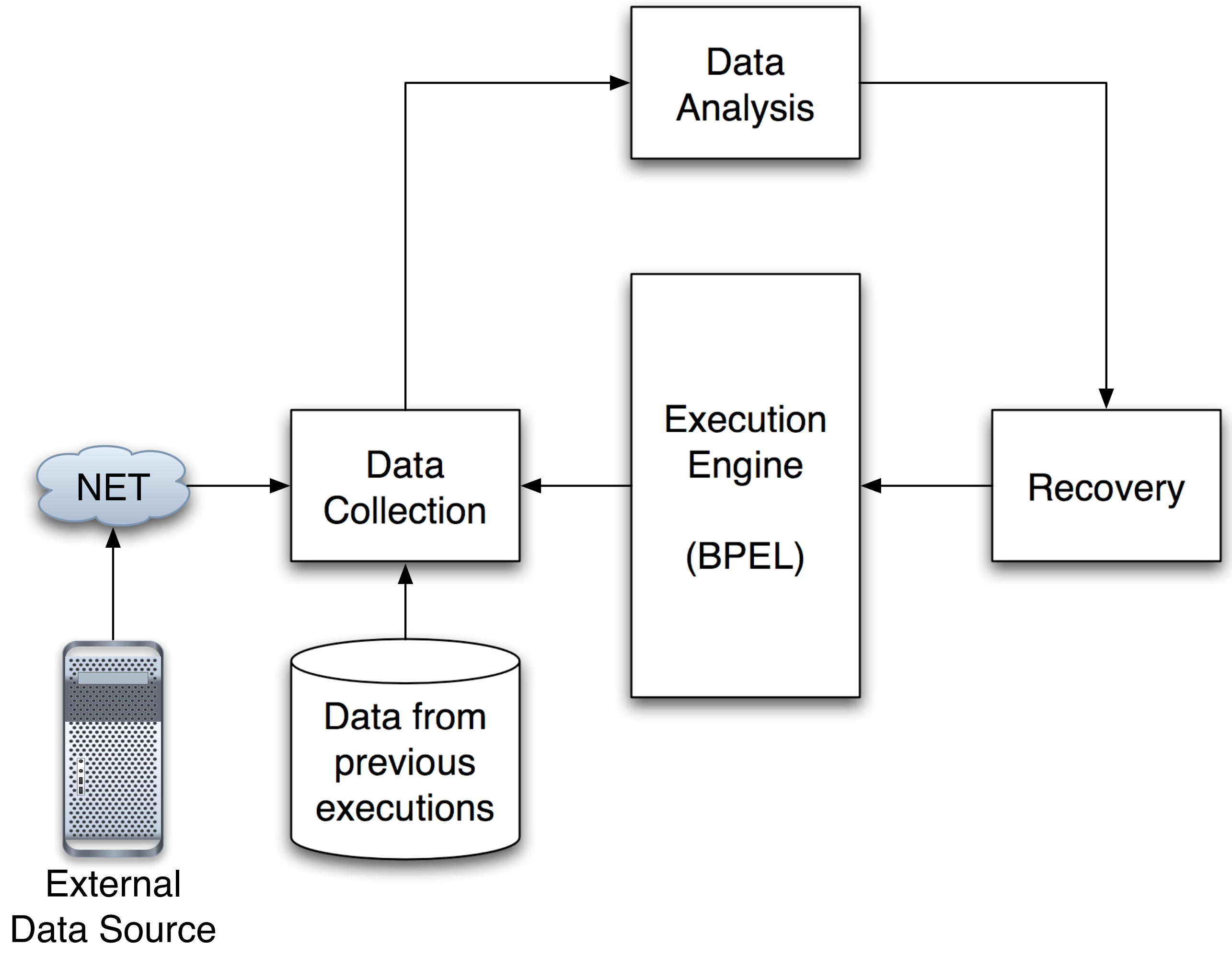
Monitoring (WSCoL)

- Assertion-based
  - The functionality and QoS needed by the process
  - Pre- and post-conditions on the interactions with partner services

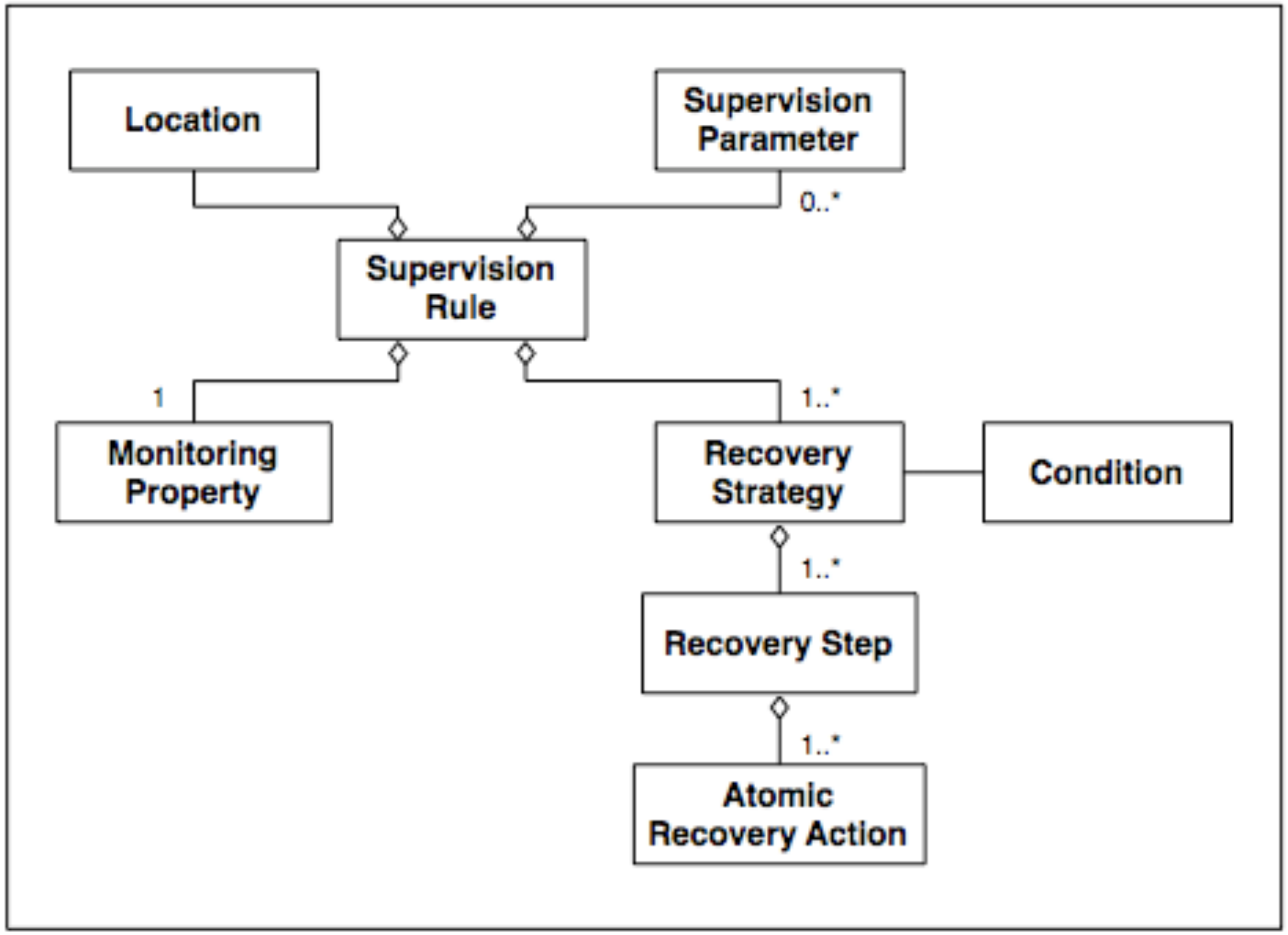
Recovery (WSReL)

- ECA rules

OVERALL IDEA



SUPERVISION RULES



WSCOL

Declarative specification of behavioral properties

Data Collection

- Internal, external, and historical variables

Data Analysis

- Boolean operators (and, or, not, implies, if and only if)
- Relational operators (<,>,==, <=, >=)
- Mathematical operators (+, -, *, /, %)
- Universal and existential quantifiers
- Data computation - max, min, avg, sum, product
- Type specific functions - length, starts-with, etc.

WSREL

Event

- Anomalies signaled by monitoring

Conditions

- WSCoL expressions

Strategies

- Each is a sequence of atomic actions

Rules have instance validity

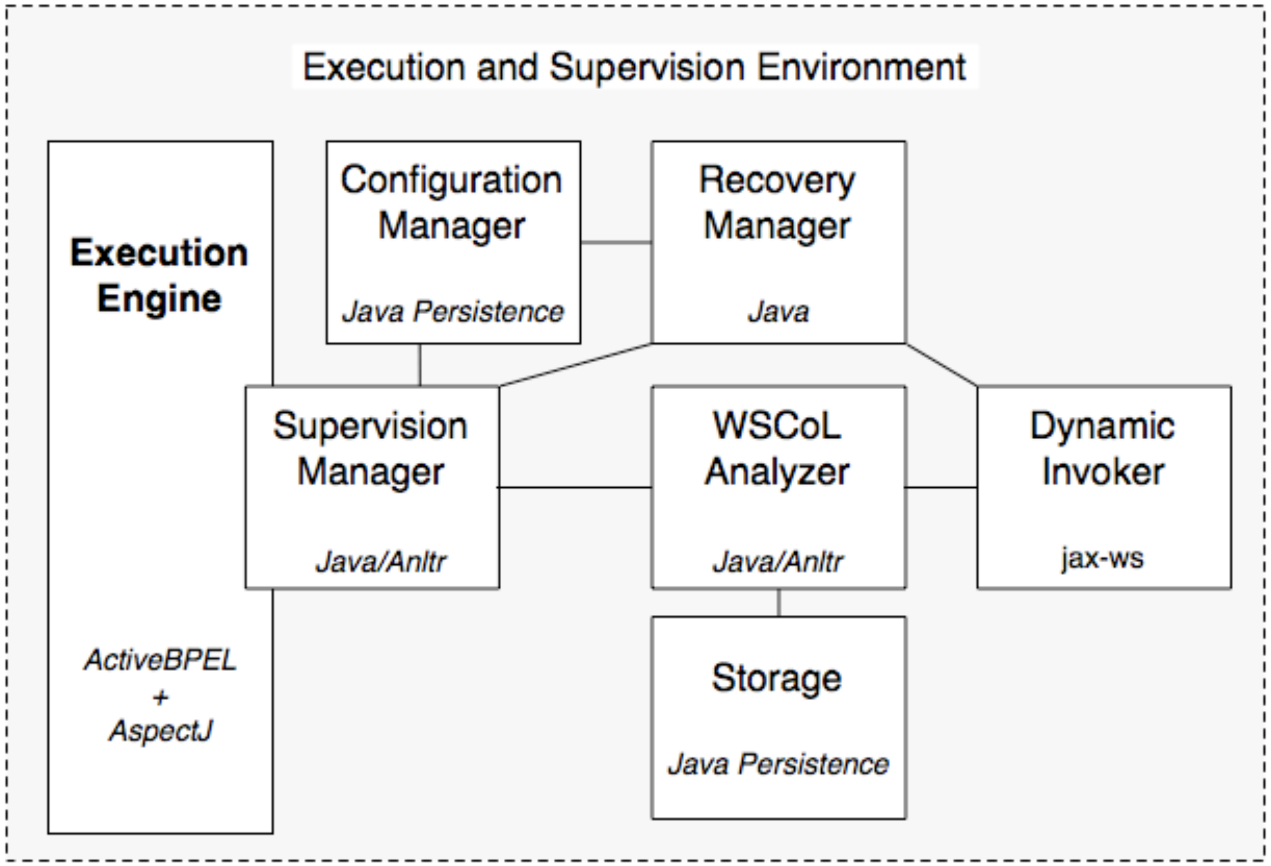
Built-in solutions

- Retry
- Change supervision rule
- Change partner link
- Call handlers
- Warn and stop
- Rollback
- Restore

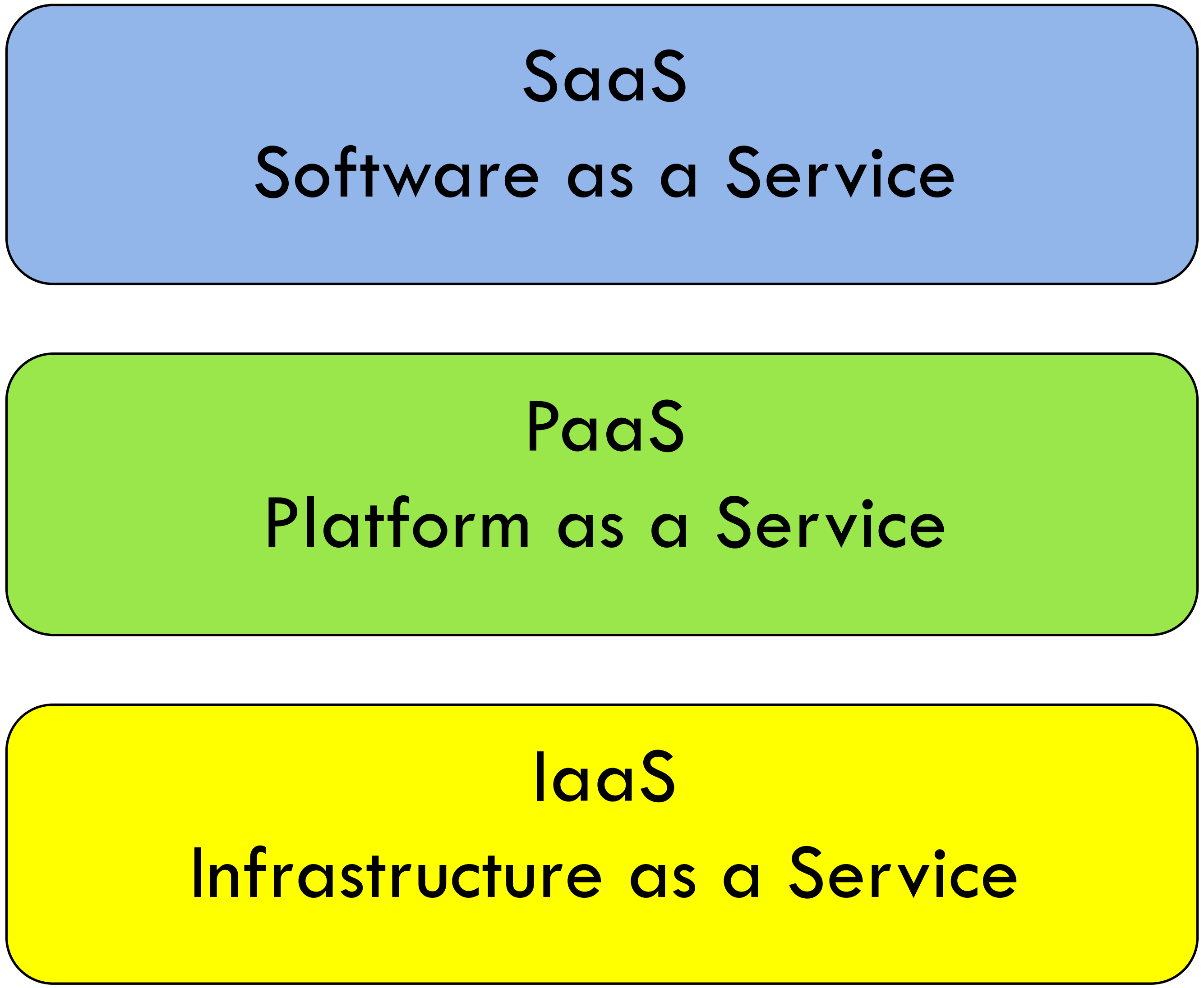
Third-party solutions

- Rebind
- Reorganize
- Renegotiate
- …

AOP-BASED SOLUTION



MULTI-LAYERED SYSTEMS

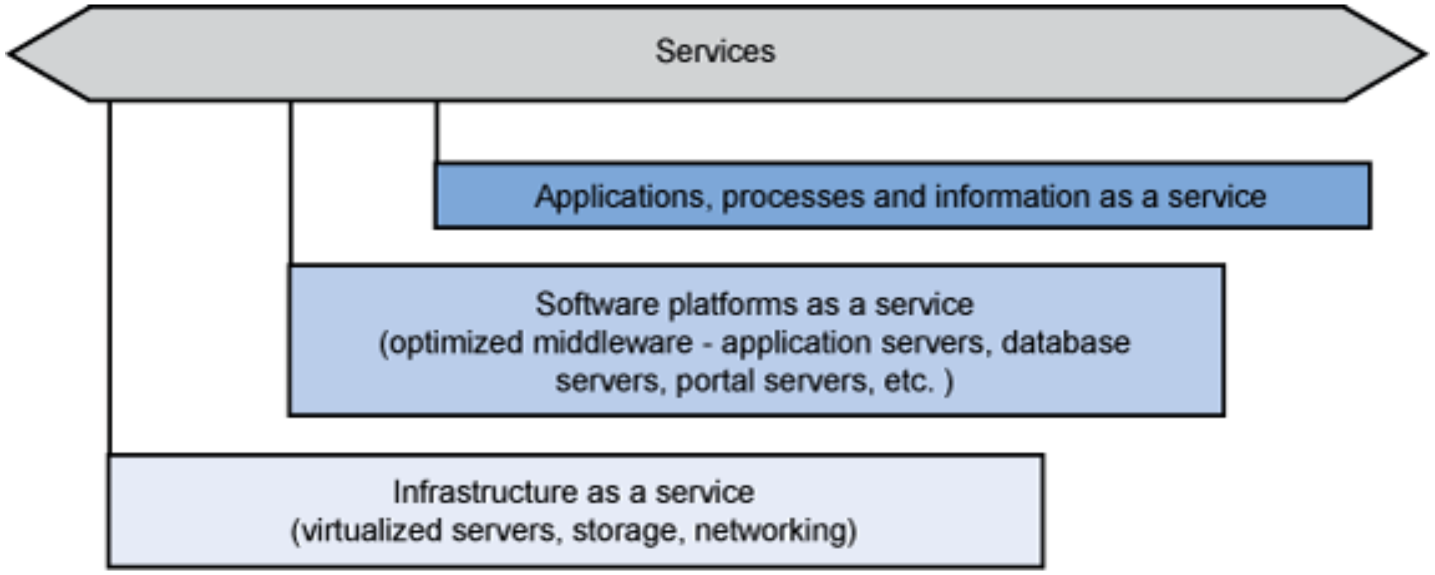

SaaS
Software as a Service
PaaS
Platform as a Service
IaaS
Infrastructure as a Service

CLOUDS AND SOA

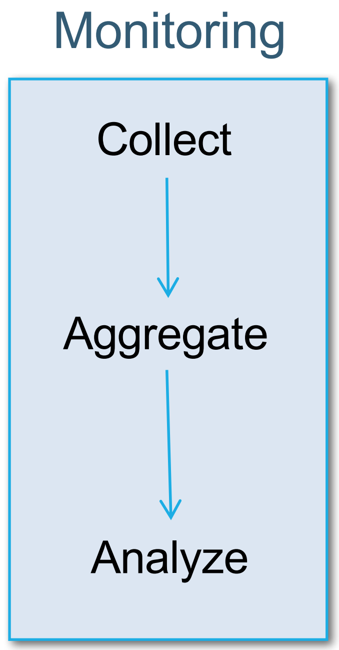
SOA enabled cloud computing to what is today

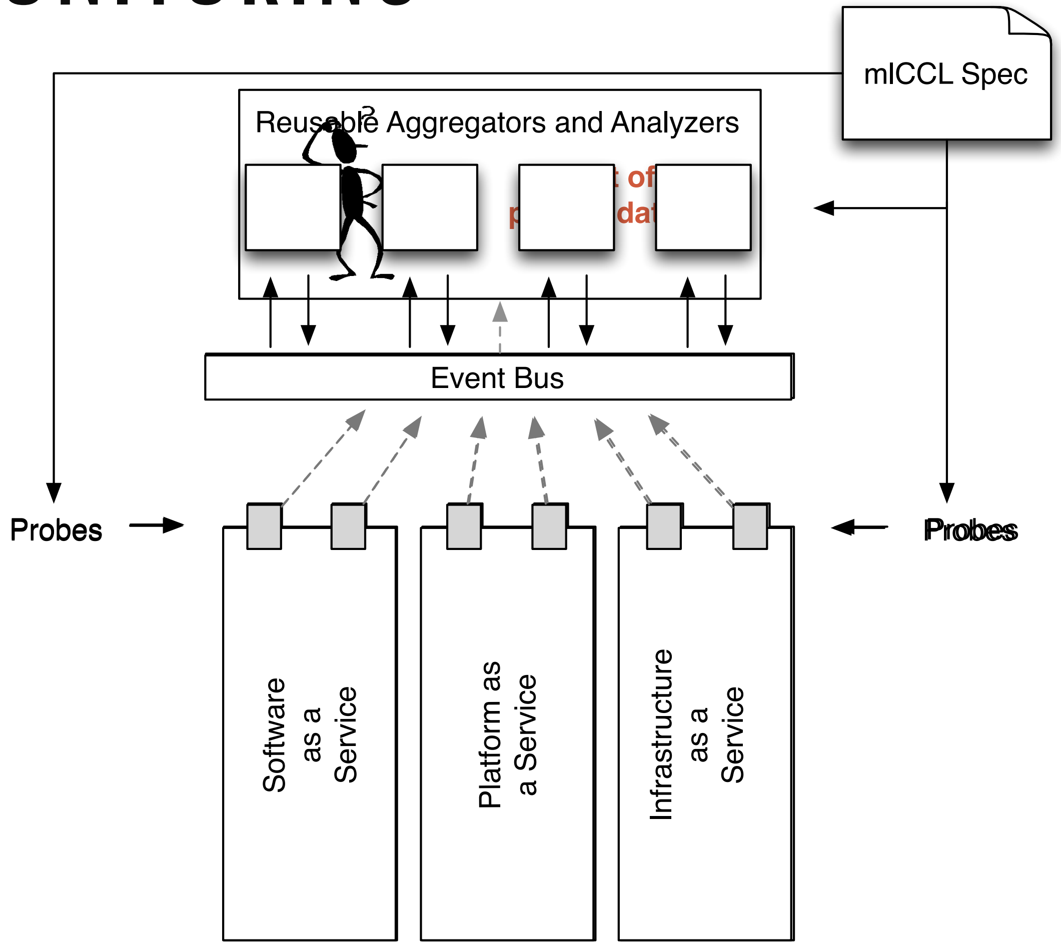
Physical infrastructure like SOA must be discoverable, manageable and governable

REST Protocol widely used (Representational State Transfer)


Services
Applications, processes and information as a service
Software platforms as a service (optimized middleware - application servers, database servers, portal servers, etc.)
Infrastructure as a service (virtualized servers, storage, networking)

MULTI-LEVEL MONITORING





Aggregators and Analyzers collaborate to produce the knowledge we need!

TWO MAIN CONTRIBUTIONS

Multi-layer Collection and Constraint Language (mlCCL)

- Declarative language for defining
  - The runtime data we want to collect from the various layers
  - How to aggregate the data to build higher-level knowledge
  - How to analyze the data to identify undesired behavior

ECoWare FrameWork

- Event Correlation middleWare
- Supports mlCCL specifications
- Provides advanced data aggregation and analysis

DATA COLLECTION

Data described in terms of Service Data Objects (SDOs)

- Language-agnostic
- Set of named properties
  - Single- or multi-valued (array)
  - Primitive (number, string, boolean) or complex (SDO)

Data Collection is about configuring probes

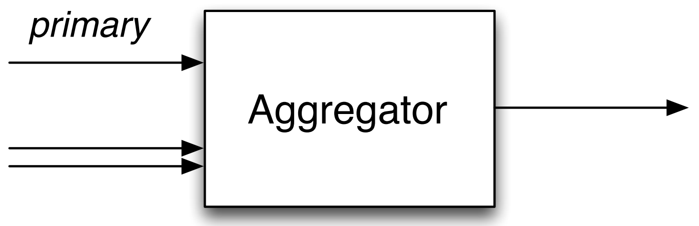
Two kinds of Data Collection:

- Message Collection
- Indicator Collection

DATA AGGREGATION

Aggregate multiple SDOs into one

- SDOs can be collected at different times…
- When to aggregate? What to aggregate?
- Primary vs. Secondary Events



- To aggregate a "window" of events, attach window(interval) to a secondary event

DATA ANALYSIS

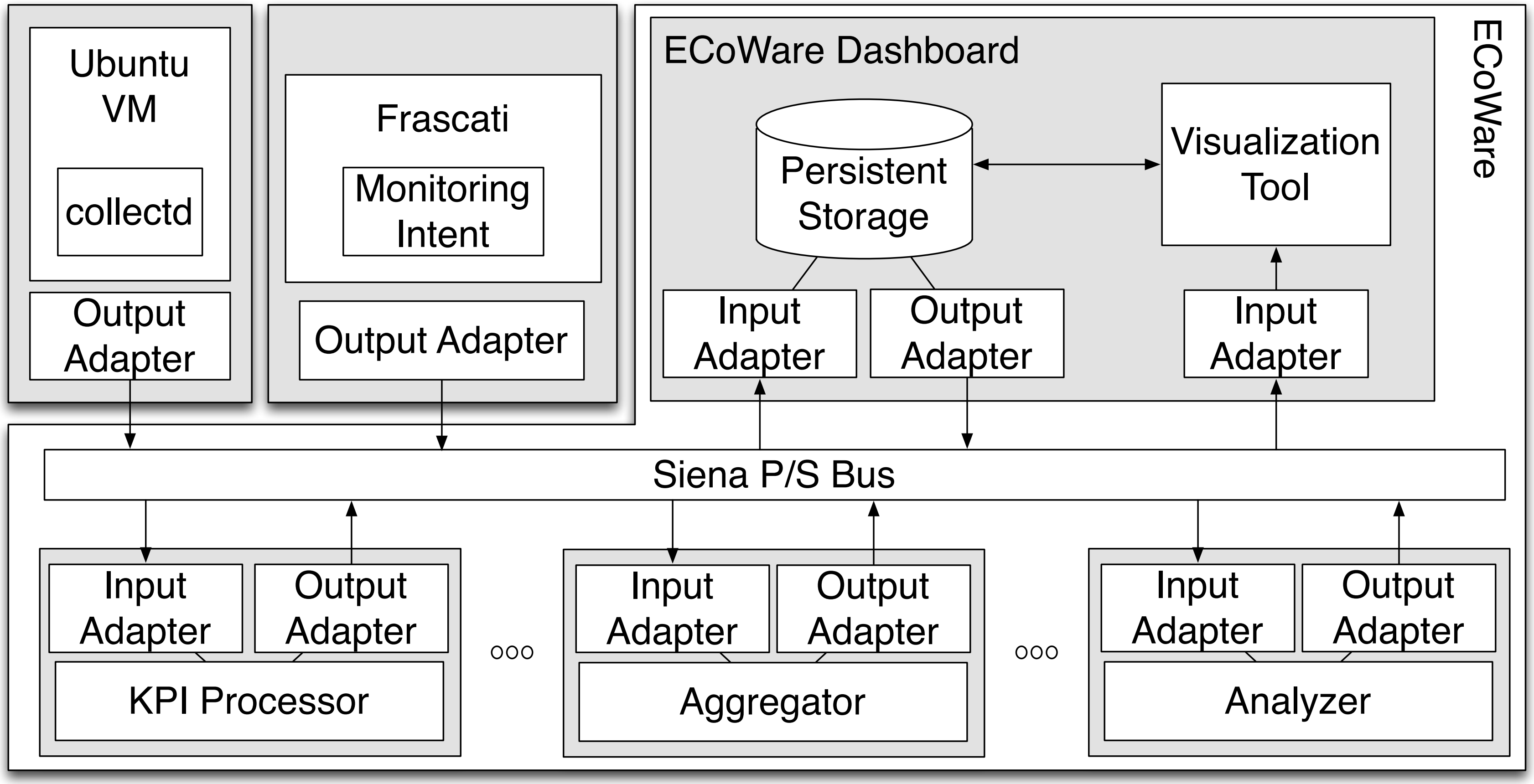
Predicate over the contents of our SDOs

- append get(propertyName) to alias
- Further manipulate the data
  - Numbers: absolute value, square root
  - Strings: substring, length, replace
  - Arrays: legnth, i-th value, subset of values that satisfy a property
  - Array of Numbers: sum, avg, min, max

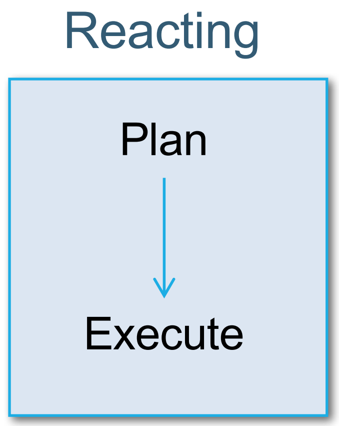
When should I perform the data analysis?

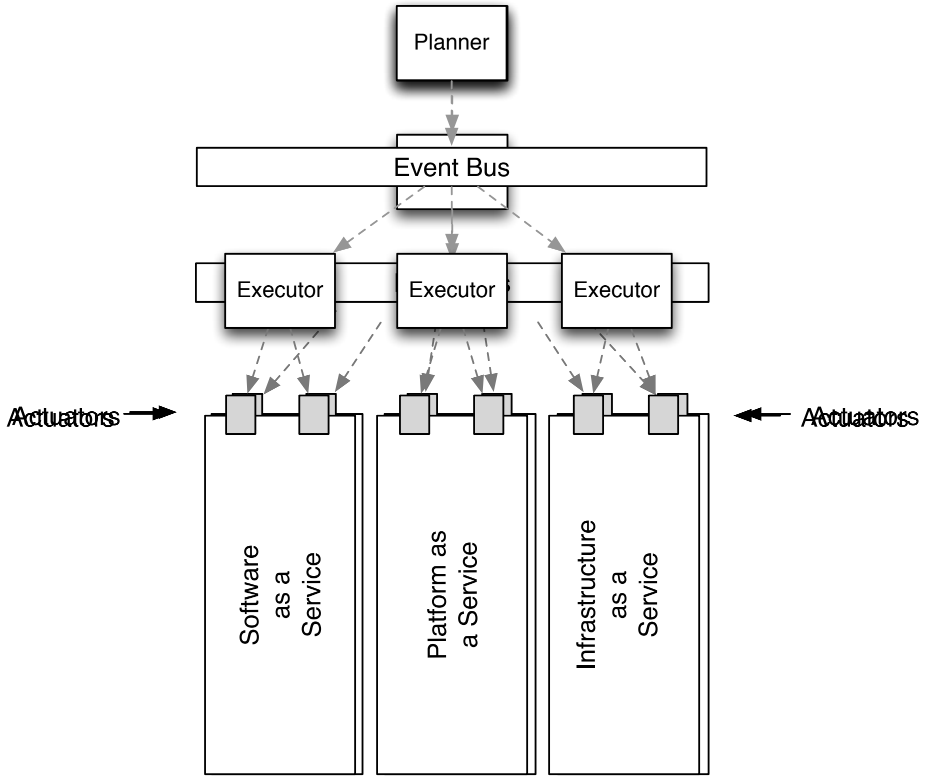
- We use primary again!

ECOWARE


Ubuntu VM
collectd
Output Adapter
Frascati
Monitoring Intent
Output Adapter
ECoWare Dashboard
Persistent Storage
Visualization Tool
Input Adapter
Output Adapter
Input Adapter
ECoWare
Siena P/S Bus
Input Adapter
Output Adapter
KPI Processor
ooo
Input Adapter
Output Adapter
Aggregator
ooo
Input Adapter
Output Adapter
Analyzer

MULTI-LEVEL ADAPTATION





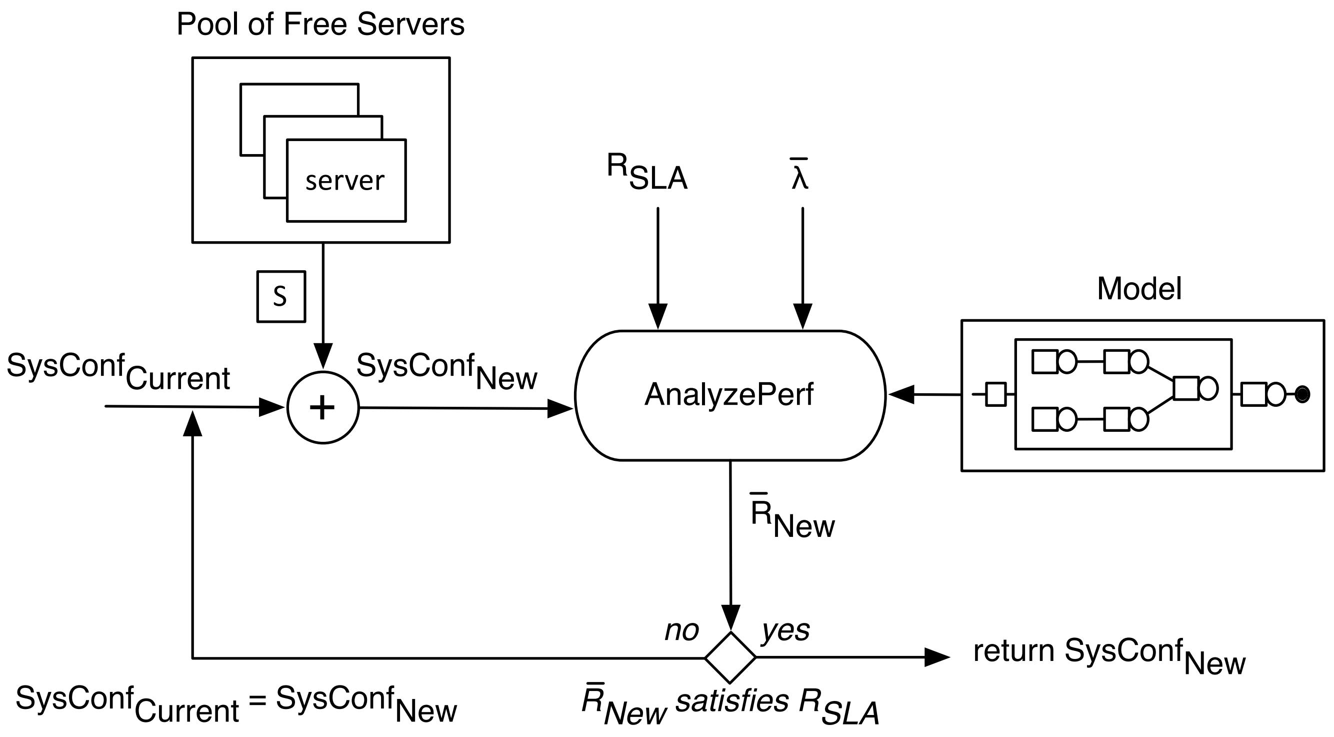
PLANNING

Planning is responsible for defining an adaptation strategy that can fix the problem

There are many important aspects we can consider. Some examples are:

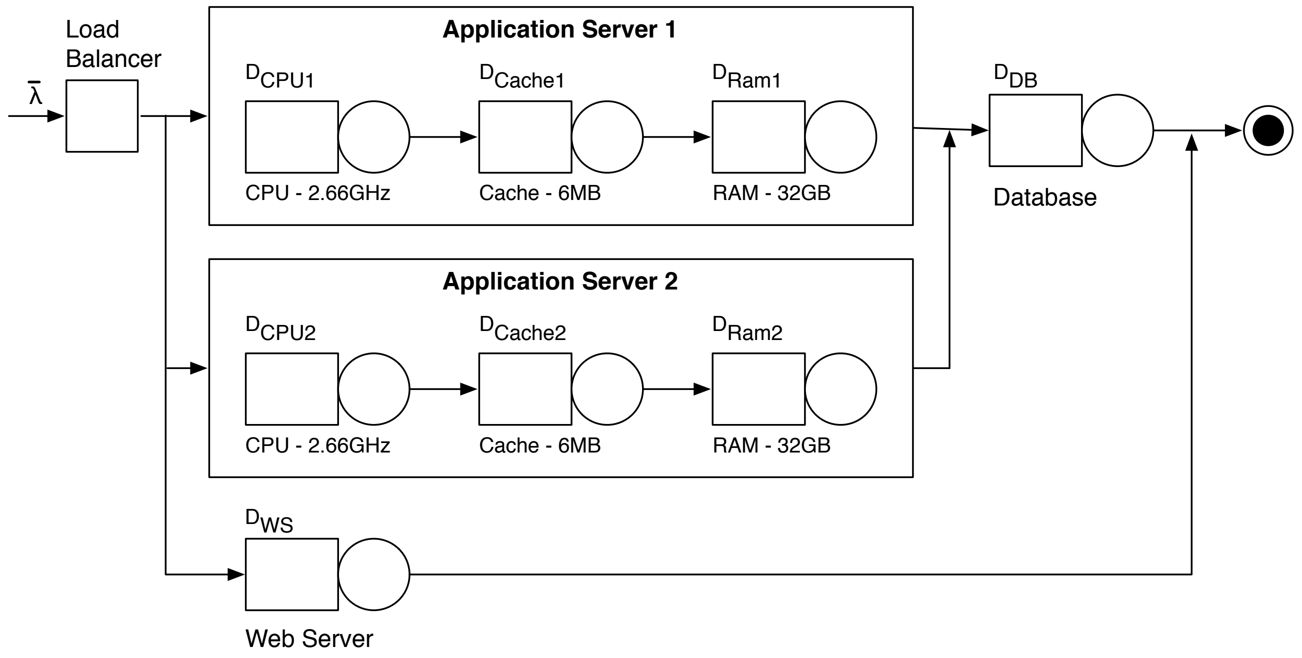
- the kind of resources we can add or remove
- the cost of adding or removing resources
- previous adaptations – to avoid continuous adaptation (cool-down)
- nature of client interactions – what call are they making?
- history of client interactions – what will supposedly happen next?

RESOURCE PLANNER

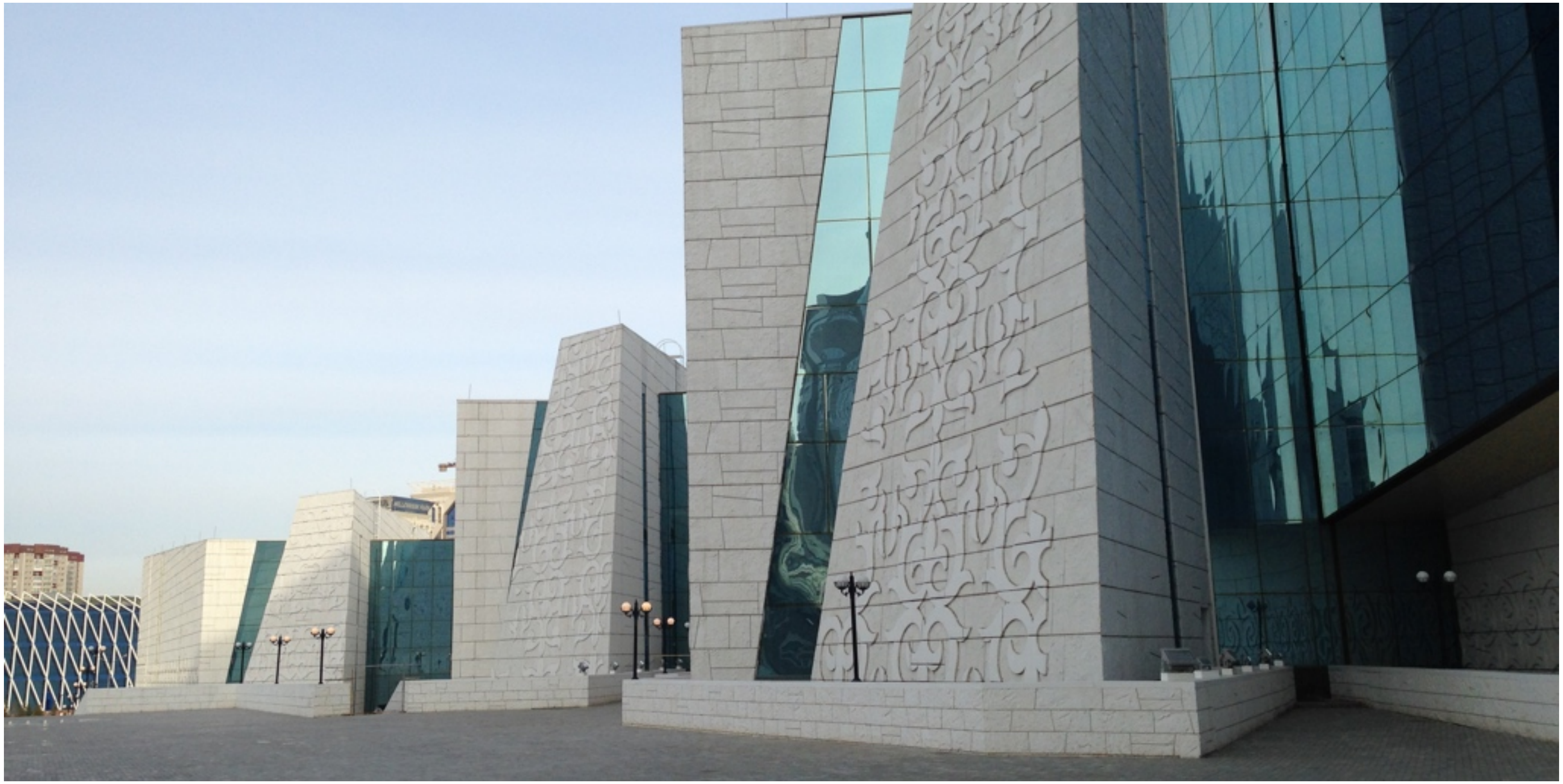


PERFORMANCE MODEL

The model considers the app's multiple tiers and multiple levels

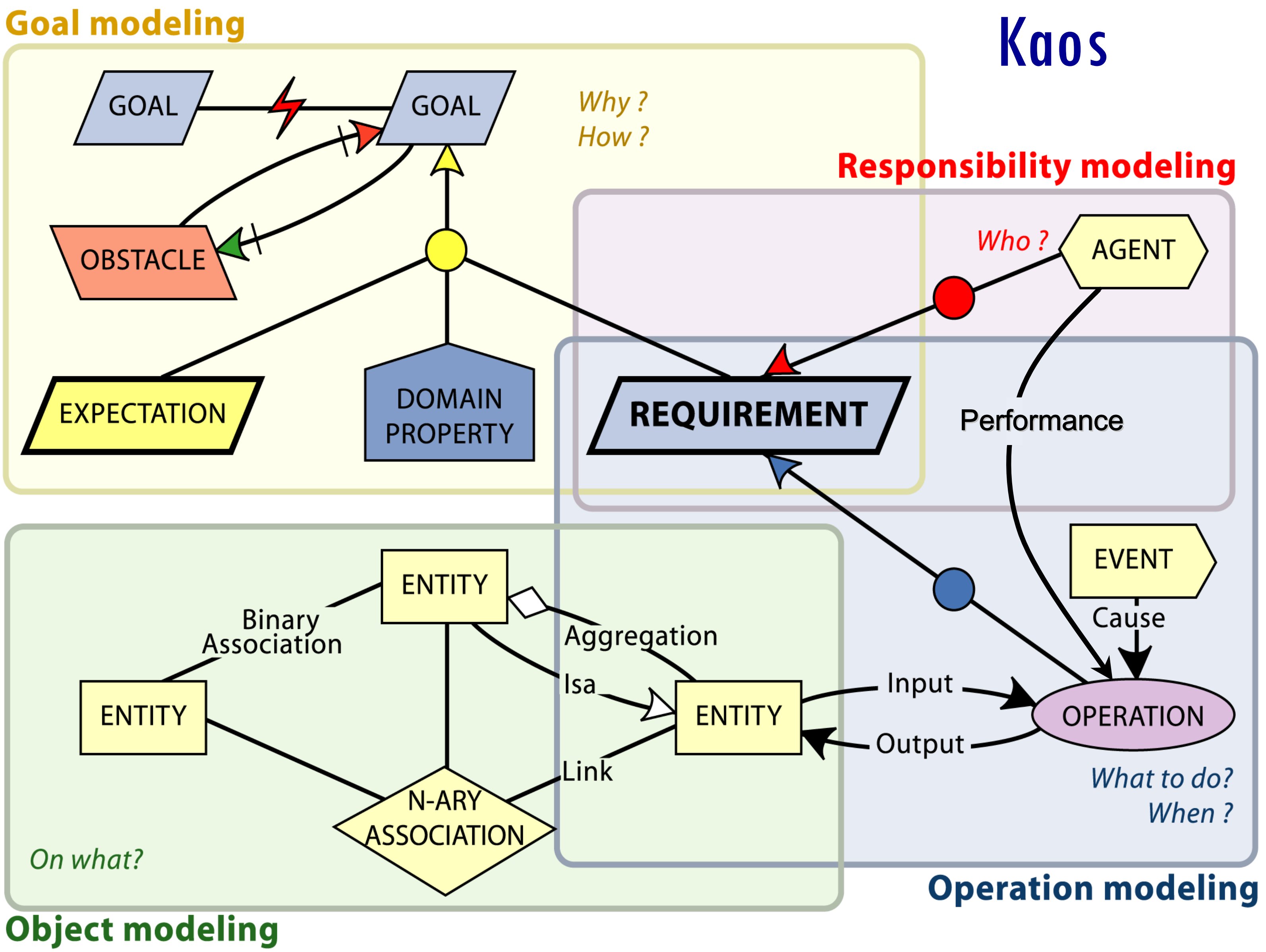


We used JMT as our model solver



REQUIREMENTS

Kaos


Goal modeling
GOAL
GOAL
Why ?
How ?
OBSTACLE
EXPECTATION
DOMAIN PROPERTY
REQUIREMENT
Responsibility modeling
Who ?
AGENT
Performance
EVENT
Cause
Input
Output
OPERATION
What to do?
When ?
Operation modeling
ENTITY
Binary Association
ENTITY
Aggregation
Isa
ENTITY
Link
N-ARY ASSOCIATION
On what?
Object modeling

FLAGS

Fuzzy Live Adaptive Goals for Self-adaptive systems

- Functionality
- Qualities of service
- Adaptation capabilities

- Distinction between crisp and fuzzy goals

CRISP GOALS

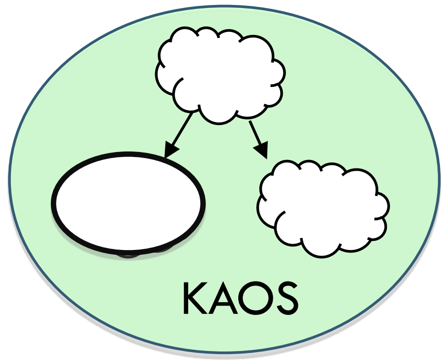
Formalization

Refinement

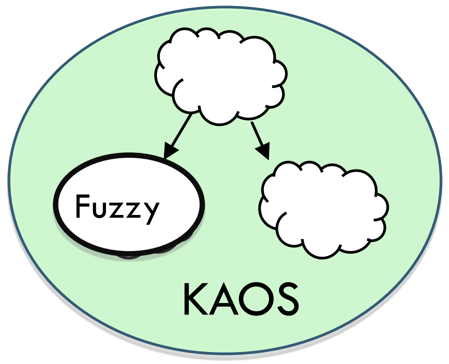
Operationalization

Nothing about

- Adaptation
- Uncertainty and small deviations
- Unforeseen adaptation



FUZZY GOALS



Satisfaction level between [0, 1]

Soft-goals

Tolerate small/transient deviations

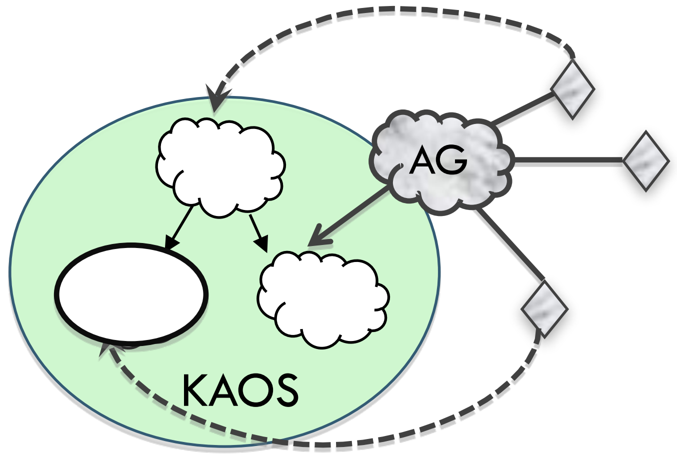
Adaptation based on satisfaction levels

ADAPTATION GOALS

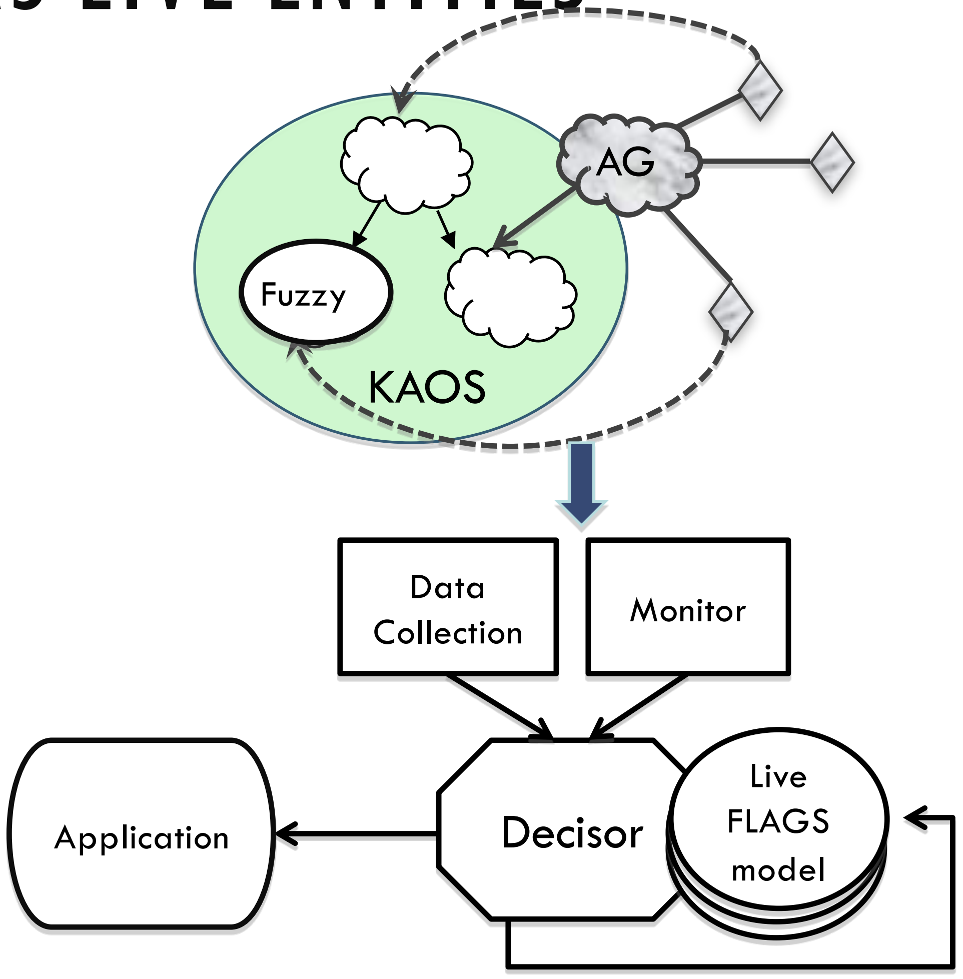
WHY (objective, related goals)

WHEN (trigger and conditions)

HOW (adaptation actions)



GOALS AS LIVE ENTITIES



PUTTING THINGS TOGETHER

Functional goals

Non-functional goals

Adaptation goals

QUESTIONS?