Intent-based Policy Management

John Strassner
with helpful insight from Joel Halpern

SDNrg, IETF 95

Agenda

- **Definitions**
- Motivation
- Traditional Formulation
- Intending to Introduce Intent
- What the SDOs are Doing (and not Doing)
- Ongoing Research
- Summary

Definitions

- **Policy**
  - "Policies are rules governing the choices in behavior of a system" – Sloman, 1994 [5]
  - "Policy is a set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects." - Strassner, 2003 [4]
- **Why We Care**
  - Devices will not, in general, be autonomic – but with appropriate management and orchestration, the overall system can appear to be autonomic
- **Types of Policies**
  - By domain or application
    - Deontic logic (e.g., obligation, authorization): ECA vs. logic-based reasoning
    - Security (mostly ECA)
    - Network Management (different disciplines)

What is Intent?

- Imperative vs Declarative
  - Imperative: CA vs ECA
  - Declarative:
    - Logic Programming
    - Functional Programming
    - Constraint Programming

Agenda

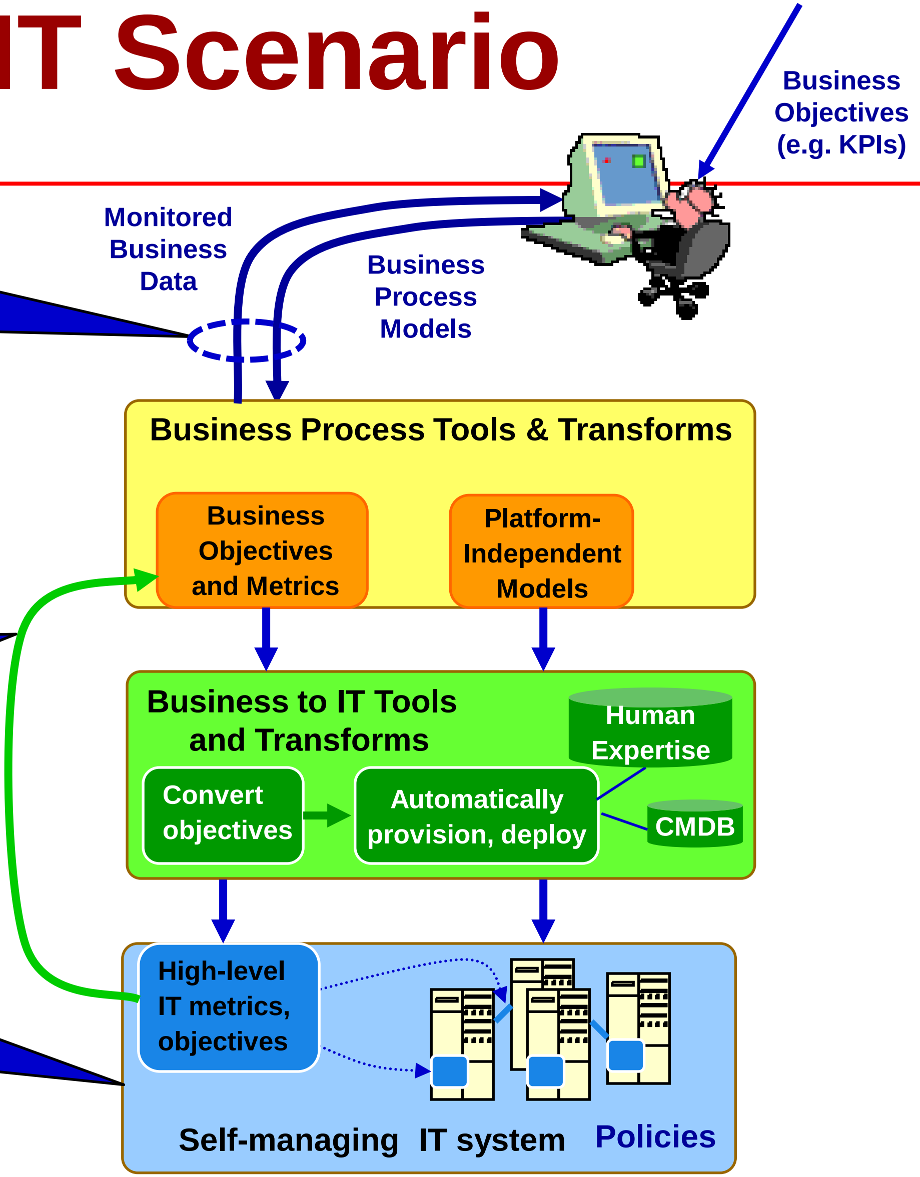
- Definitions
- **Motivation**
- Traditional Formulation
- Intending to Introduce Intent
- What the SDOs are Doing (and not Doing)
- Ongoing Research
- Summary

Business-Driven IT Scenario

There will be continuous feedback between the Business and the rest of the System to calibrate business-to-IT transformations

Translation of models, metrics and objectives from business terms to IT terms will become increasingly automated

Human specification of low-level, platform-specific policies gives way to high-level discipline-specific objectives with tradeoffs



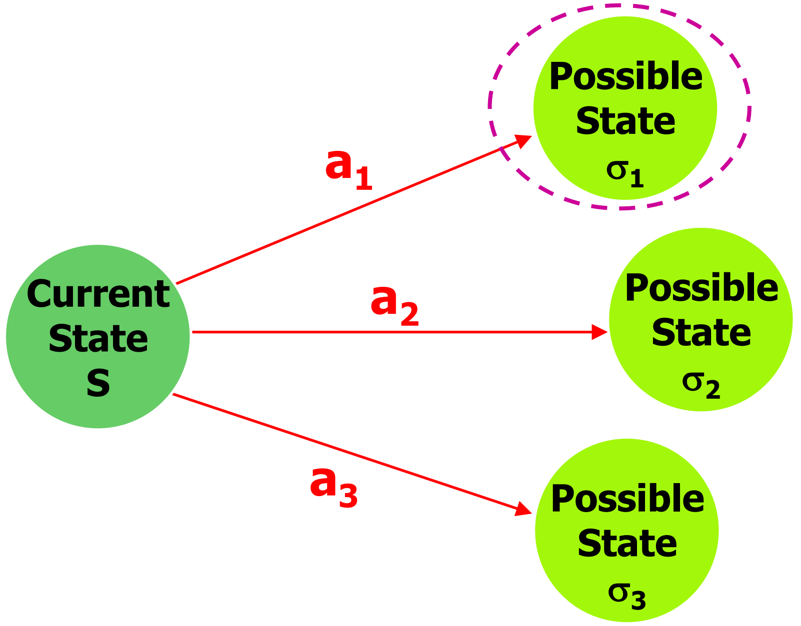
Agenda

- Definitions
- Motivation
- **Traditional Formulation**
- Intending to Introduce Intent
- What the SDOs are Doing (and not Doing)
- Ongoing Research
- Summary

Imperative (ECA) Policy Rules

- **ECA Policy**
  - Specifies action *a* that should be taken in *current* state *S when* event E is received

    ON (*Event*) IF(*Condition*) THEN (*Action*)
  - *Event* triggers evaluation of the condition
  - *Condition* specifies state or set of states
  - *Action* defines what is required to transition to this state
  - Knowledge:
    - Current state *S*
    - Action to take *a*
  - Policy author (human or computer) knows exactly what should be done



Rationality is *compiled into the policy*

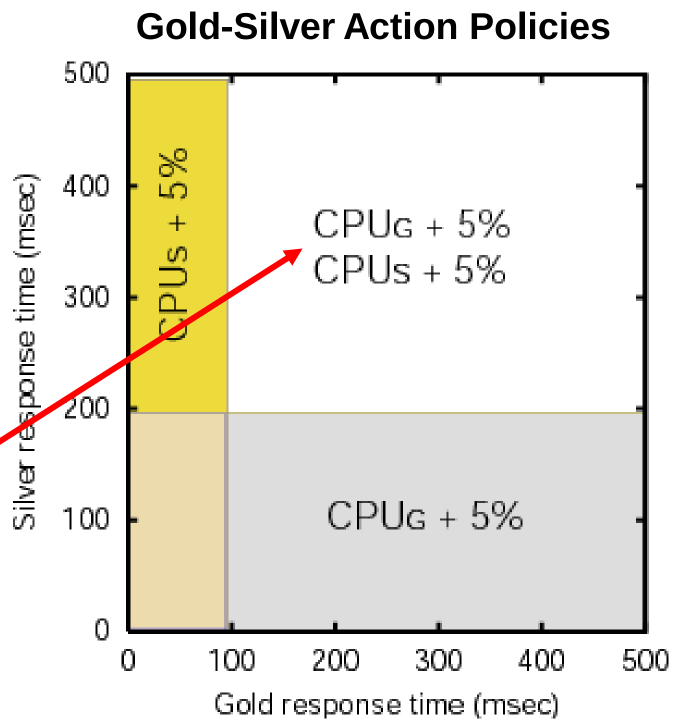
Ref [1]

Imperative Policy Conflicts

Gold: IF (RT_G > 100 msec)
THEN (Increase CPU_G by 5%)

Silver: IF (RT_S > 200 msec)
THEN (Increase CPU_S by 5%)

Overlapping Action Policies (conflict depends on CPU utilization) *



G: IF (RT_G > 100 msec) THEN (Increase CPU_G by 5%) : Priority = 10
S: IF (RT_S > 200 msec) THEN (Increase CPU_S by 5%) : Priority = 5

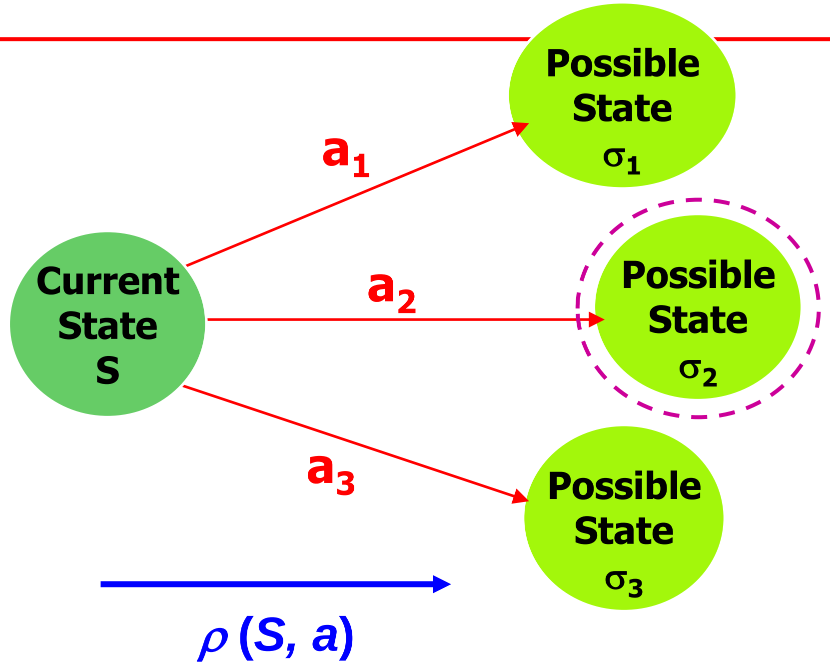
Ref [1, 11]

*** Priorities work for simple ECA cases, but cannot solve all conflicts***

Declarative (Goal) Policy Rules*

- **Declarative (a.k.a., Goal) Policy**
  - Specifies desired *resulting* state ρ or criteria for set of states
    - ➢ Any member of desired states acceptable
  - System must compute action
    $a: S \rightarrow \rho$
  - Objective: Desired state ρ
  - Knowledge
    - ➢ Current state S
    - ➢ *System model*: $\rho\,(S, a)$

Rational behavior is *generated* by optimizer/planner



Compare to action policies:

- **What we *want*, rather than what to do**
  - **Higher-level**
  - **More flexible**
- **Requires sophisticated models, optimization/planning algorithms**

Ref [1]

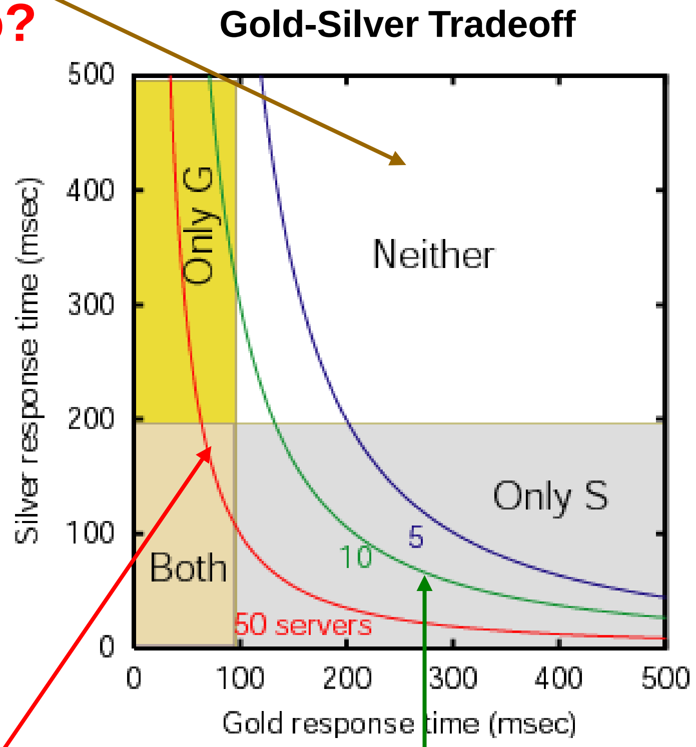
** Inspiration for, but not the same as, "Intent"*

Goal Policy Conflicts

G: RT_G < 100 msec

S: RT_S < 200 msec

It's all bad!
What to do?



It's all good!
What is best?

Conflict:
Gold/Silver Tradeoff
What to do?

Ref [1]

Resolving Conflicts in Goal Policies

Priorities

G: RT_G < 100 msec, Priority 10
S: RT_S < 200 msec, Priority 5
B: RT_B < 250 msec, Priority 3

Typical priority semantics:

1. Satisfy top priority goal (if feasible)
2. Satisfy second priority goal (if feasible)

   ...

N. Satisfy Nth priority goal (if feasible)

Do we always want to satisfy Gold at the expense of all other Services?

- Better to partially satisfy all classes?
- Better to satisfy both Silver and Bronze at expense of Gold?

Simple goals and priorities provide a limited language

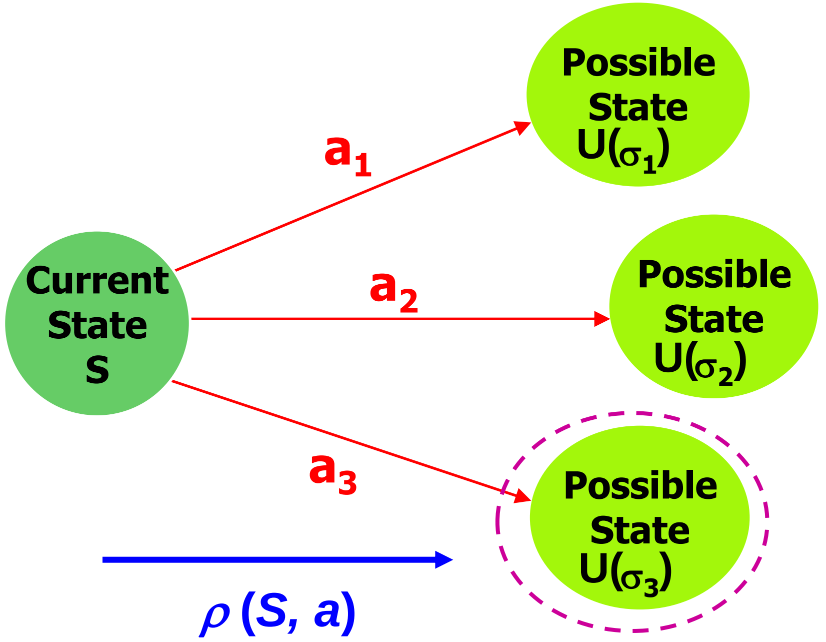
– Could enumerate compound goals with associated priorities
– A better way is to use utility functions!

Ref [1]

Utility Function Policies

- **Utility Function Policy**
  - Function assigns a single real value to each *resulting* state
  - Tradeoffs directly encoded, thus no conflicts
  - System must compute optimal action
  - Objective: Maximize $U(\rho)$
  - Knowledge
    - ➢ Current state S
    - ➢ *System m odel*: $\rho\,(S, a)$

Rational behavior is *generated* by optimizer/planner

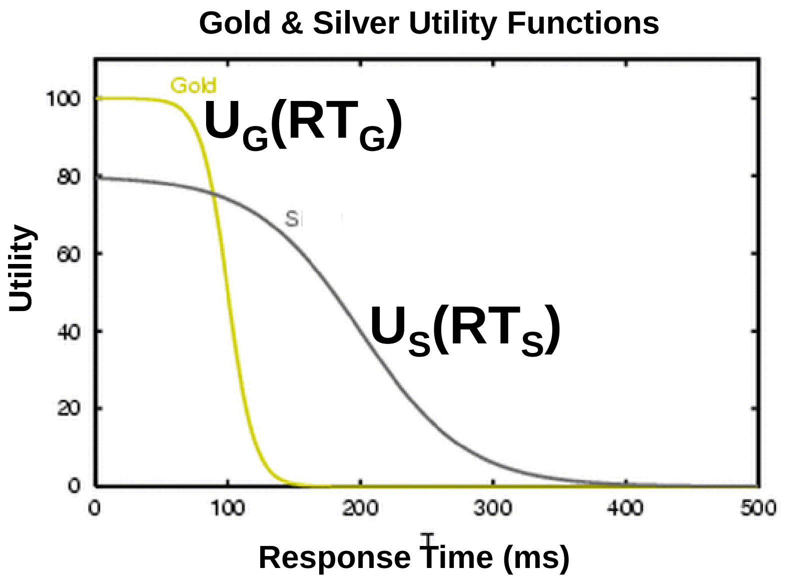


Compare to other policy types:

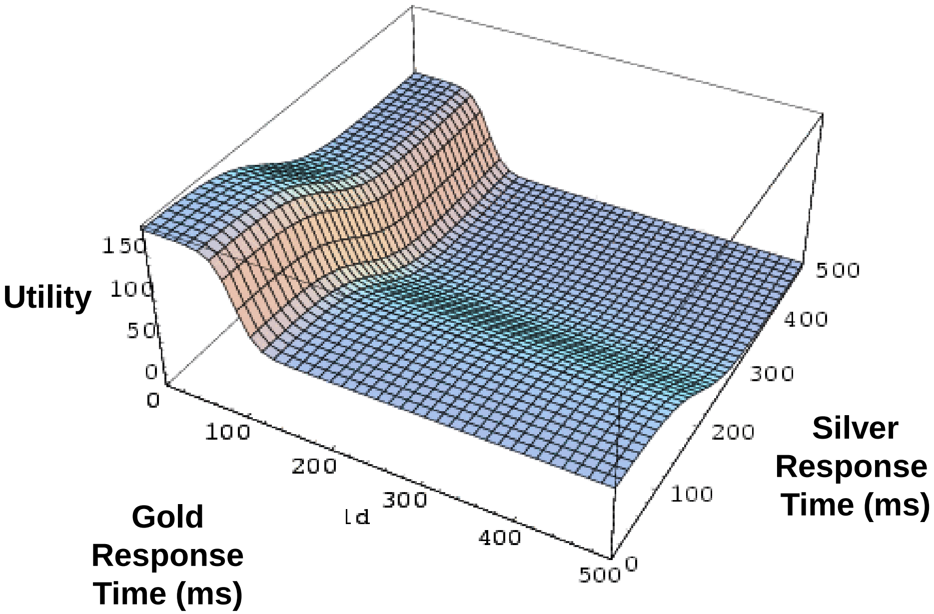
- **High-level & flexible (like Goal)**
- **Range of state values (rather than binary Goal classification)**
- **Strict generalization of Goal**
- **No conflicts (like Action and Goal)**
- **Utility elicitation can be hard!**

Ref [1]

Utility Function Policies



$U(RT_G, RT_S) = U_G(RT_G) + U_S(RT_S)$

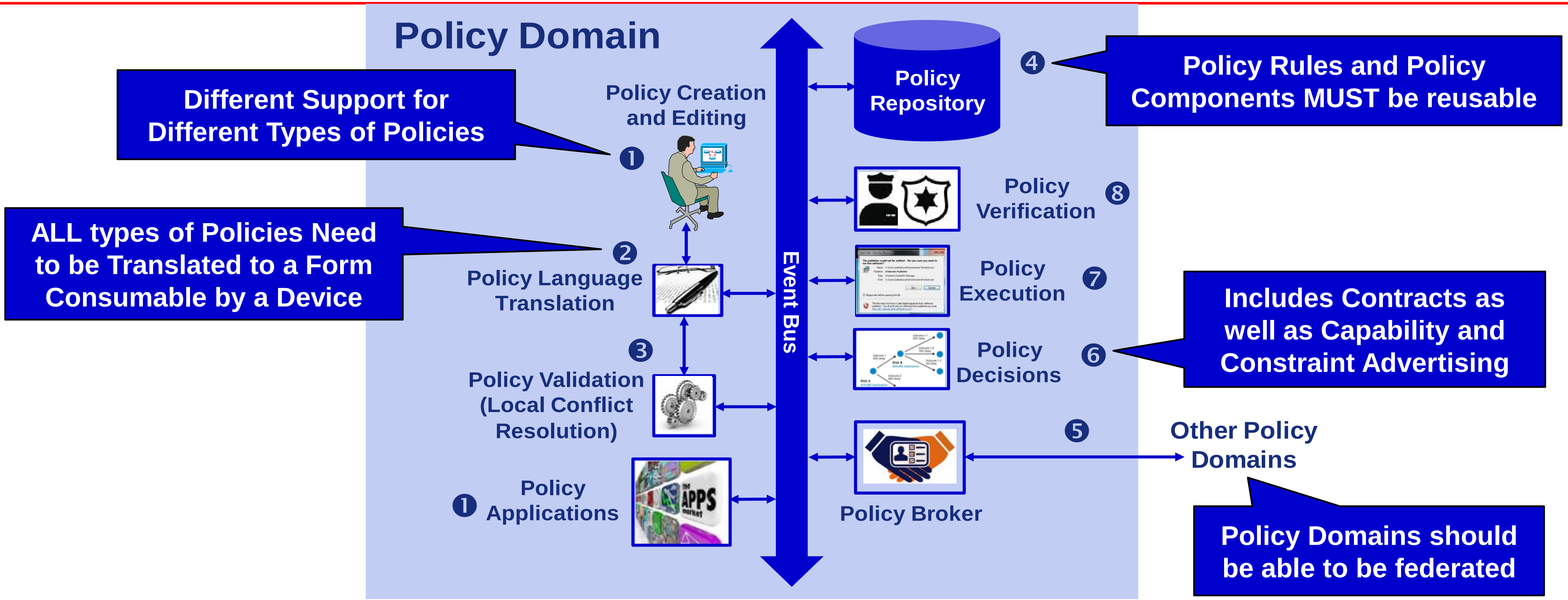


- States have real value, rather than binary good/bad classification
- Map all states of interest in to single unique value
- Tradeoffs directly encoded, so there are NO conflicts!*

Ref [1]

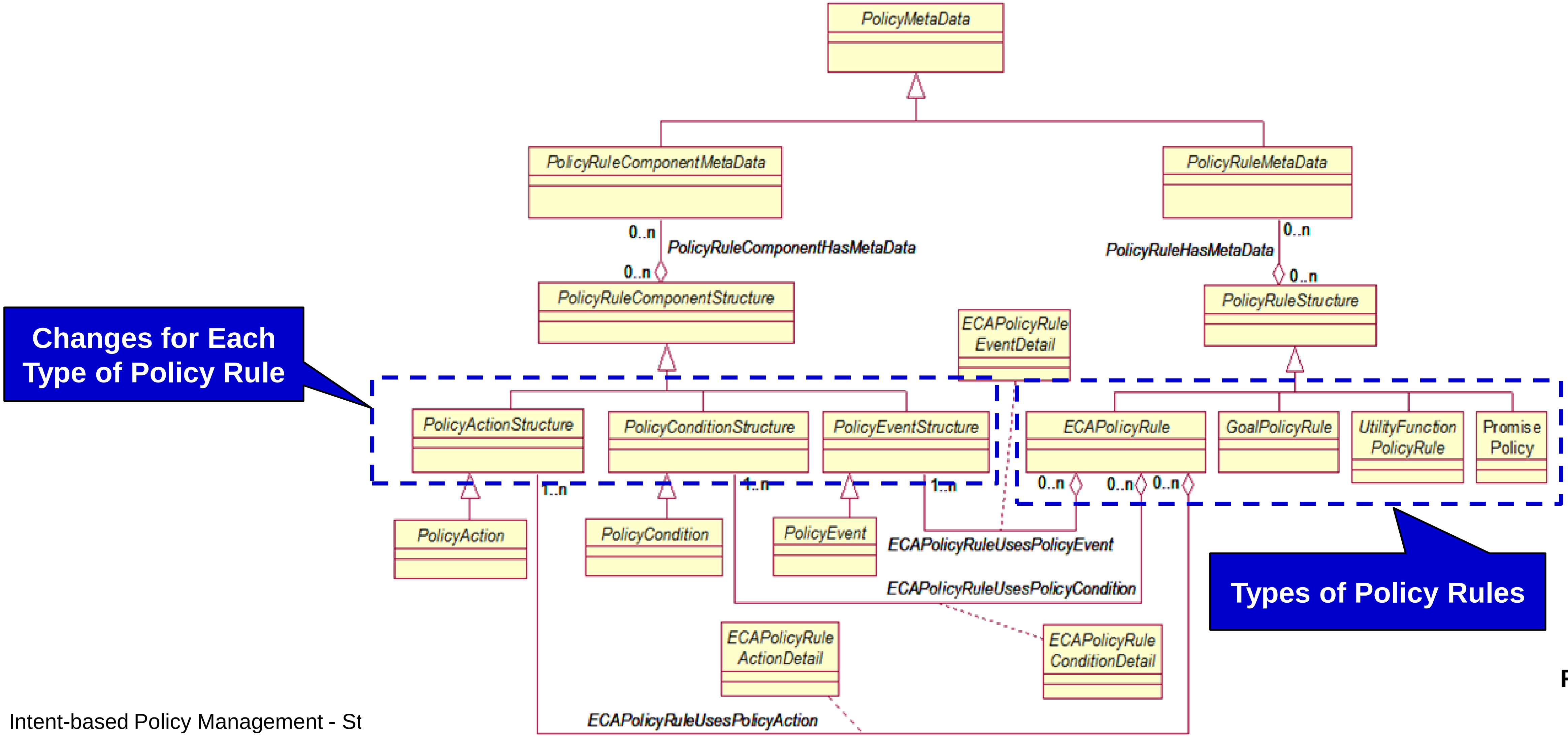
** Assuming that the utility functions were designed in concert*

An Exemplary Policy Architecture



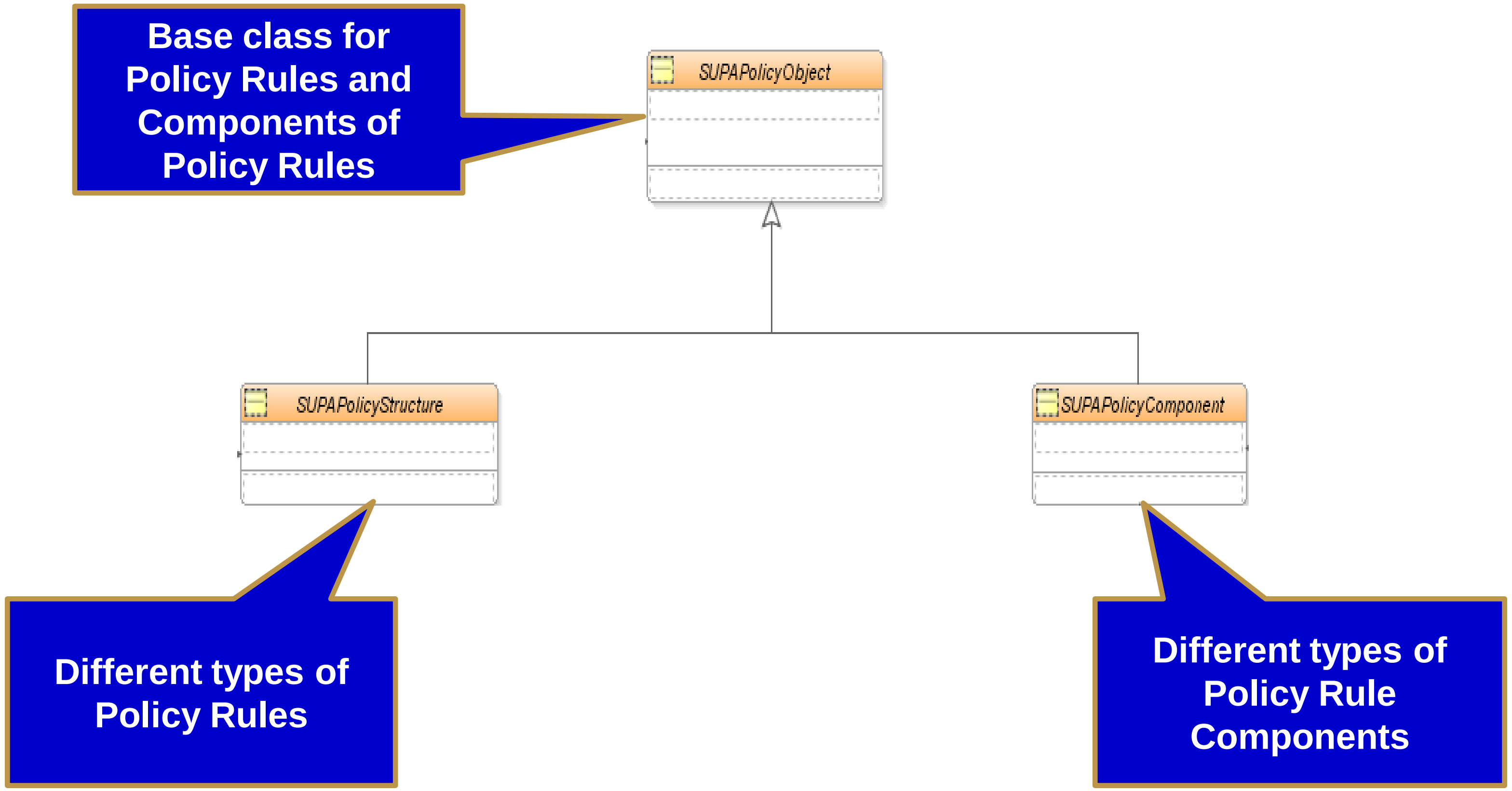
Ref [14]

The Policy Portion of DEN-ng



Ref [14]

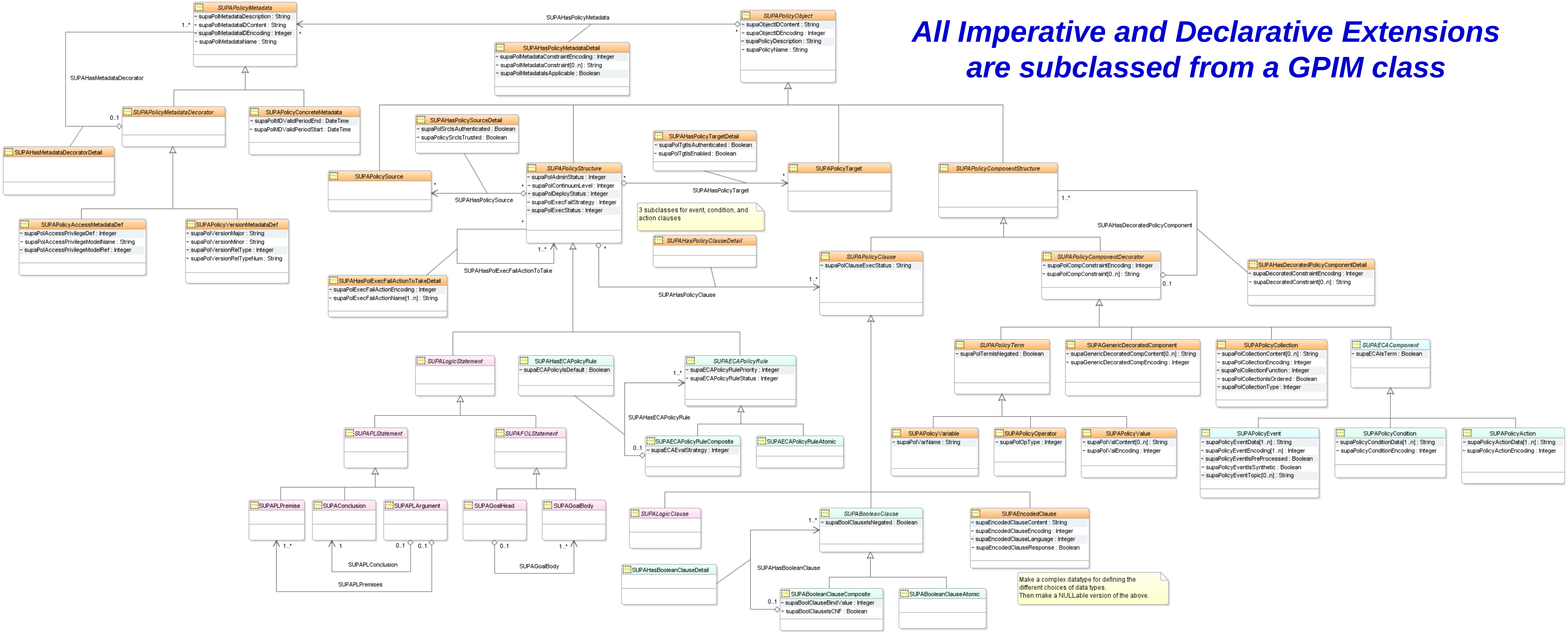
The SUPA GPIM



Ref [3]

SUPA Generic Policy Rules

All Imperative and Declarative Extensions are subclassed from a GPIM class



Ref [3]

Note: please see a demo of the SUPA Policy Engine at BnB on Thursday!

Agenda

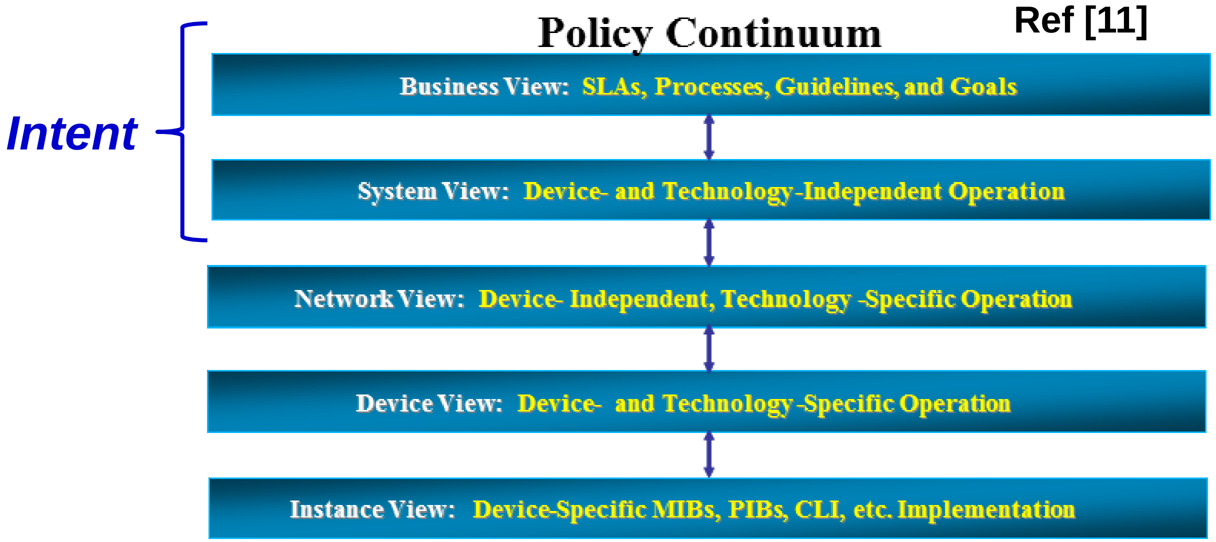
- Definitions
- Motivation
- Traditional Formulation
- **Intending to Introduce Intent**
- What the SDOs are Doing (and not Doing)
- Ongoing Research
- Summary

Motivation for Intent

- **Policy Management is HARD**
  - People want simpler solutions
- **Many Different Constituencies Want Intent**
  - End Users who aren't technical want to define policies to control behavior
  - Application Developers want to build Network Services, but existing network interfaces don't help them do this
  - Operators want more abstract and powerful ways to define Network Services
  - Intent offers the ability to define consumer abstractions that invoke Network Services

Intent Discussions in the ANIMA WG (1) *

- **Who Writes Intent**
  – Originated by humans, not by devices
- **What Does Intent Look Like**
  – My opinion: a restricted natural language
- **Who or What Consumes Intent**
  – One form of a policy; must be translated to a form that is consumable by a device
- **How Is Intent Used**
  – The probability of a device being able to consume multiple intents that use the same natural language is very low, and negative for using multiple natural languages



** These are MY opinions; they have been posted on the ANIMA WG, but have not achieved consensus*

Intent Discussions in the ANIMA WG (2) *

- **Is Intent Large in Size?**
  - NO! However, it could affect a large number of devices, and/or when translated to lower-level forms, could generate a lot of policies
  - If intent becomes large, it is likely that it is not actually intent
- **How Many Intents Will Be Present?**
  - IFF it is easy to use, a LOT
  - Hiding complexity from the user will increase implementation complexity.
- **Should We Combine Intent into a Single File?**
  - WHY is this needed? Plus, see slide 24

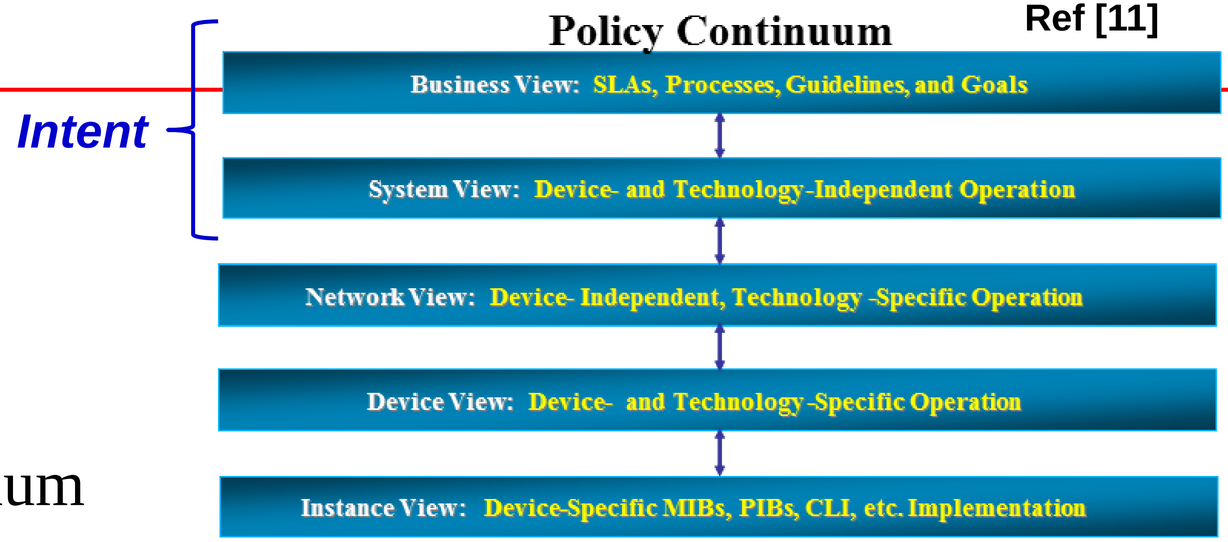
** These are MY opinions; they have been posted on the ANIMA WG, but have not achieved consensus*

Intent Discussions in the ANIMA WG (3) *

- **Do We Need to Specify the Target(s) of Intent?**
  – The target(s) should be able to be inferred from the intent without having to specify low-level details (e.g., ports and IP addresses).
- **Can Intent be Updated by Devices?**
  – Intent MUST be transformed to a form that devices can consume. However, since Intent is (by my definition) a restricted natural language, it takes too many resources to construct and validate to be put in routers and switches
- **What About Context?**
  – Every SDO I know of has NOT considered context. This is very dangerous – how does the system adapt to change, and understand if intent is no longer valid?

** These are MY opinions; they have been posted on the ANIMA WG, but have not achieved consensus*

Intent Discussions in the ANIMA WG (4) *

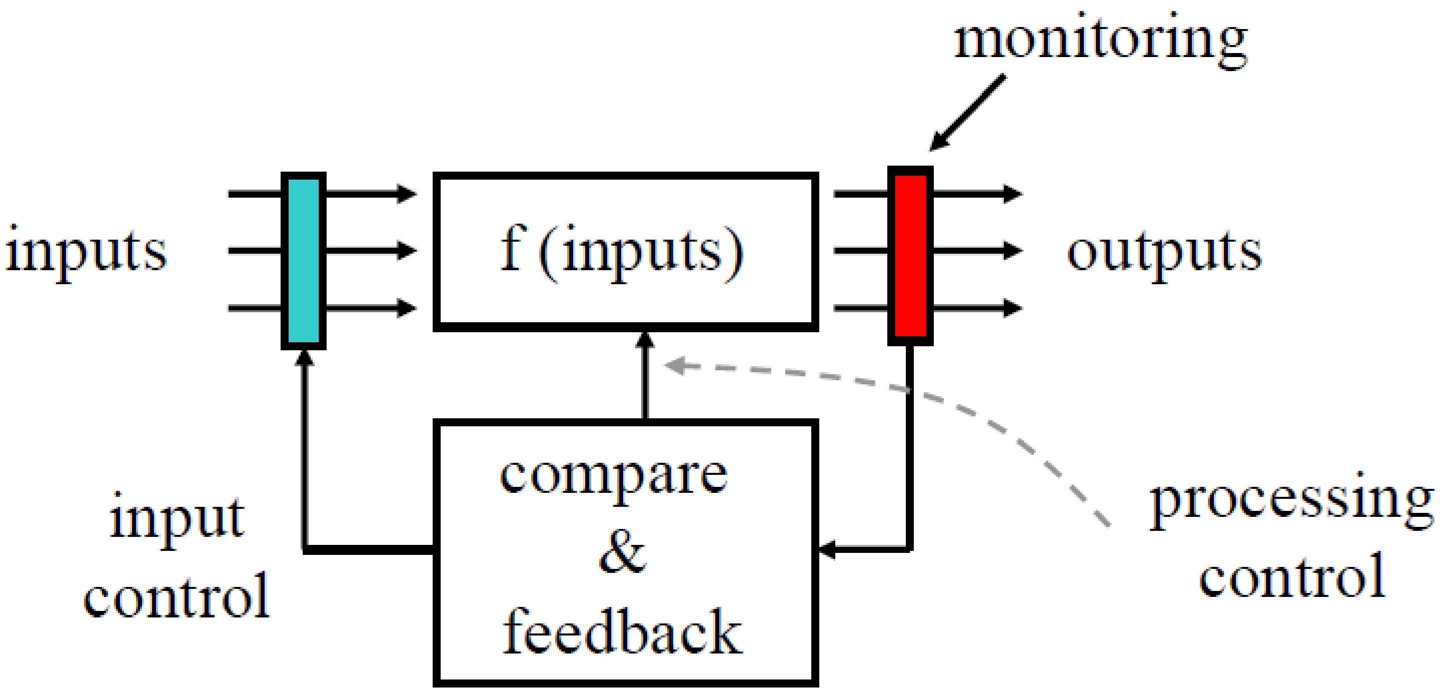


- **How Do We Identify Intent?**
  - I recommend {domain, role, context}
- **Are There Types of Intent?**
  - Intent is one layer in the Policy Continuum
  - The number and nature of each continuum is determined by the actors that use it
- **Who/What Validates, Coordinates, and Distributes Intent?**
  - A dedicated management entity (e.g., a set of agents) validates and distributes intent (typically using a pub-sub bus; ANIMA is discussing flooding instead)
  - Devices MUST NOT coordinate and distribute intent – they do not have a complete view of the system

** These are MY opinions; they have been posted on the ANIMA WG, but have not achieved consensus*

An Important Note

Policy may not be an atomic blob!



Ref [12]

Agenda

- Definitions
- Motivation
- Traditional Formulation
- Intending to Introduce Intent
- **What the SDOs are Doing (and not Doing)**
- Ongoing Research
- Summary

Intent Inside the IETF

- **SUPA Could Use Data Produced by These WGs as Data for Policies**
  - I2RS, ALTO
- **SUPA Could Help**
  - L3SM map L3 VPN service requests to L3 VPN configurations on network devices
  - TEAS define which TE data should be used per customer, and how flows should be treated abstractly
  - BESS (BGP Enabled Services) generate BGP configurations by using BESS data
  - NVO3 define how the behavior of logically centralized network virtualization management entities
- **Since Declarative Policy is Currently Not in Scope for SUPA**
  - SDNrg could be a good place to work on and research how to implement declarative policies

Intent Outside the IETF

- **NFV has defined VNFs**
  - These are lower-level functions, as they are not consumer-oriented; policy needs more definition
- **ONF is working on Intent**
  - A long series of discussions about what Intent is, but no concrete work; policy needs more definition
- **MEF and TMF are thinking about Intent**
  - So far, there aren't any active WGs that are formalizing Intent
  - MEF is bottom-up, but has a good orchestration definition; TMF is top-down, but has a good policy model and definition
- **Open Source**
  - OpenStack Congress is a declarative model; ODL GBP is a relational model
  - Neither is defining an ***abstract*** form of Intent suitable for most application developers and end-users

Agenda

- Definitions
- Motivation
- Traditional Formulation
- Intending to Introduce Intent
- What the SDOs are Doing (and not Doing)
- **Ongoing Research**
- Summary

The Importance of Semantics

"An object by itself is intensely uninteresting"

- Grady Booch, Object Oriented Design with Applications, 1991

Data	Examples	What You Get
Types of Data	Machine data, documents, multimedia, email, blogs, pictures, LOD, …	**Syntax** *Context and semantics are hidden*
Named Entities	Objects in a model, or concepts in an ontology	**Context** *Semantics are hidden*
Relationships	Typically *hidden in the data*	**Semantics** *Now the data are understood!*

Increasing Meaning and Computational Complexity

- **Semantics**
  - The key to understanding data, and being able to make decisions
  - **Context** *orients* the data, **semantics** *helps interpret* the data   Ref [2]
  - Intent ***needs semantics*** in order to be properly understood!

DEN-ng Context Definition*

“

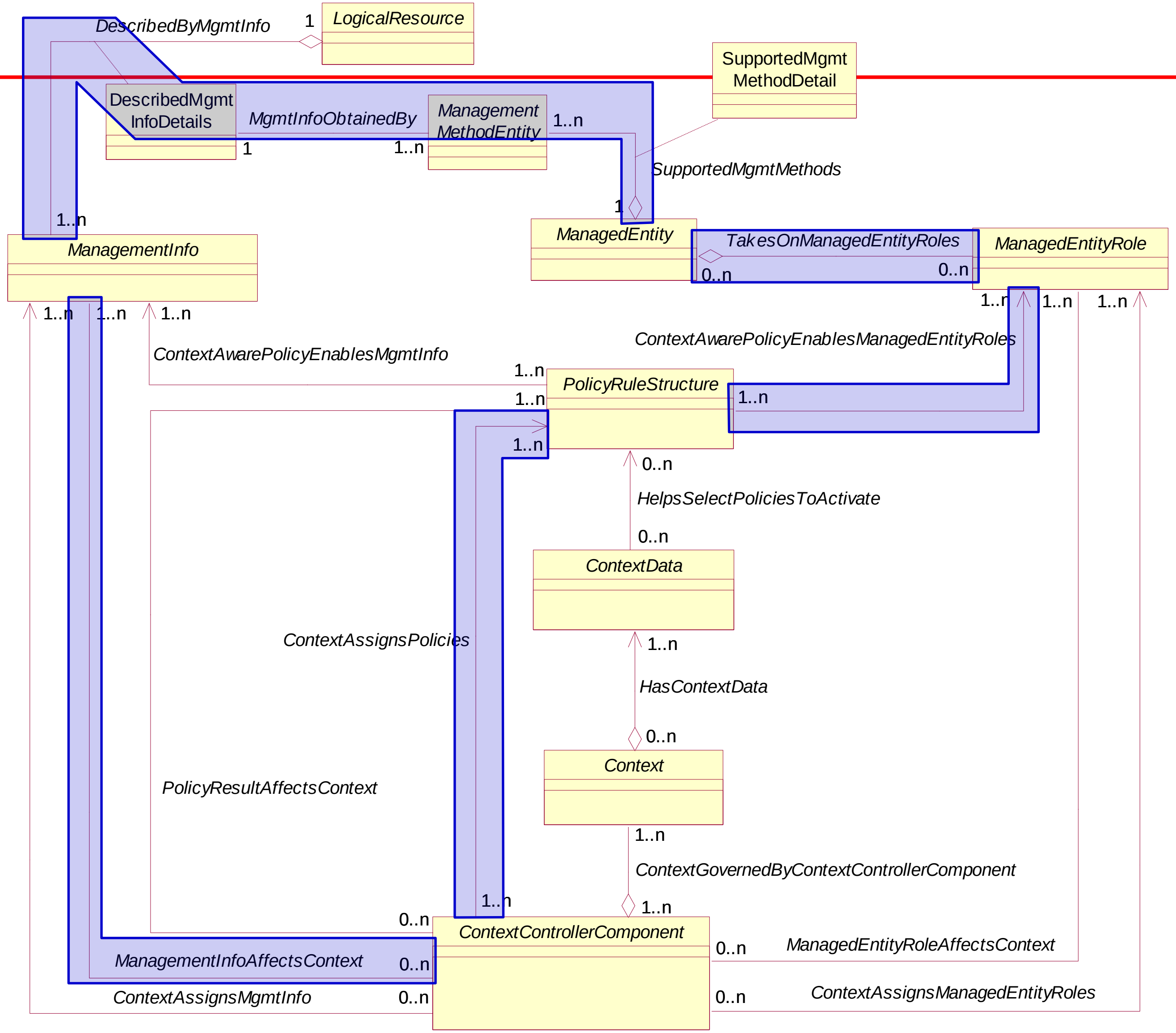
The Context of an Entity is a collection of measured and inferred knowledge that describe the *state* and *environment* in which an Entity exists or has existed

”

** See next slide as to how Context could be used in Policy Systems*

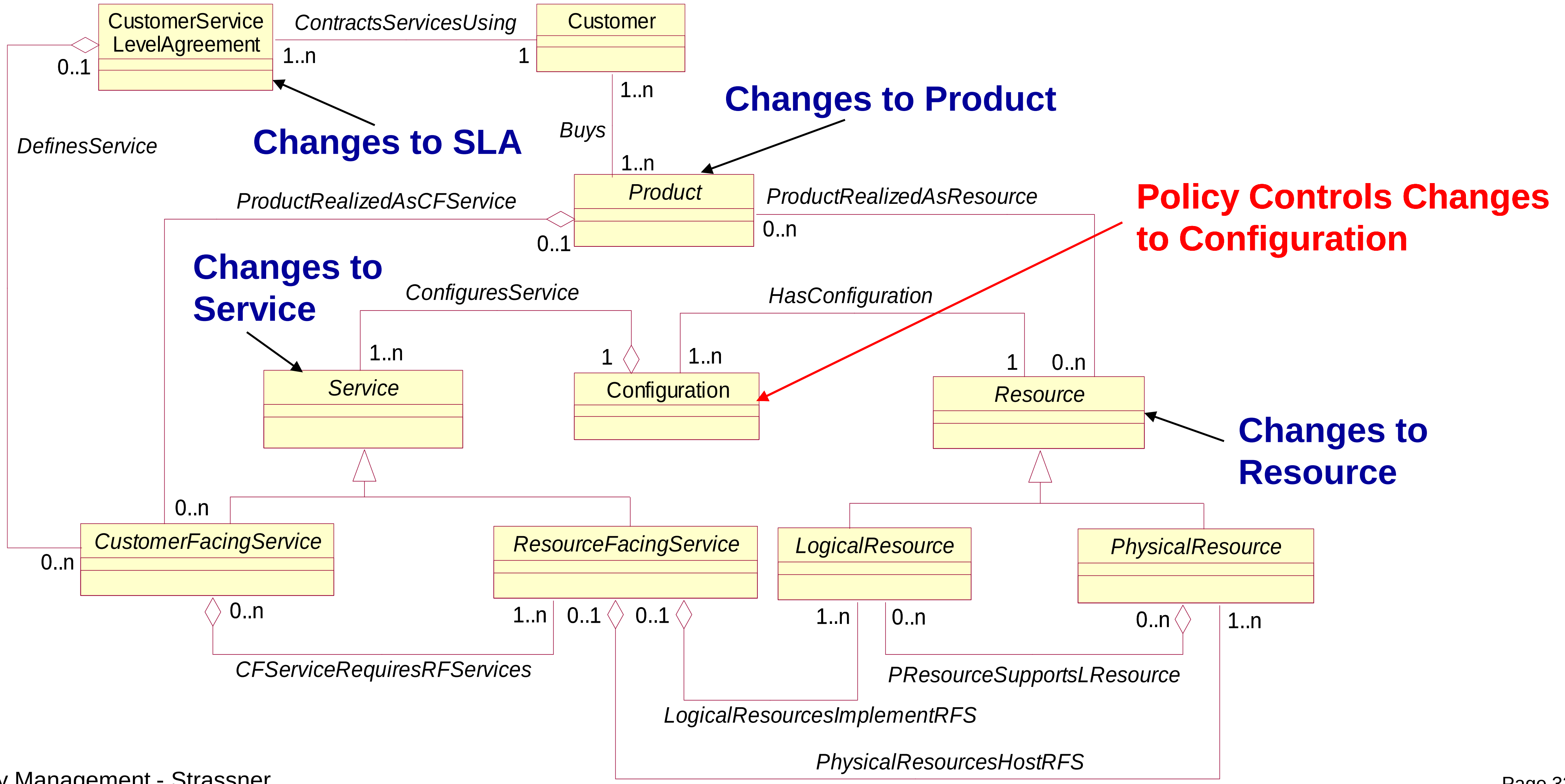
Ref [2]

Context Provides Situation Awareness

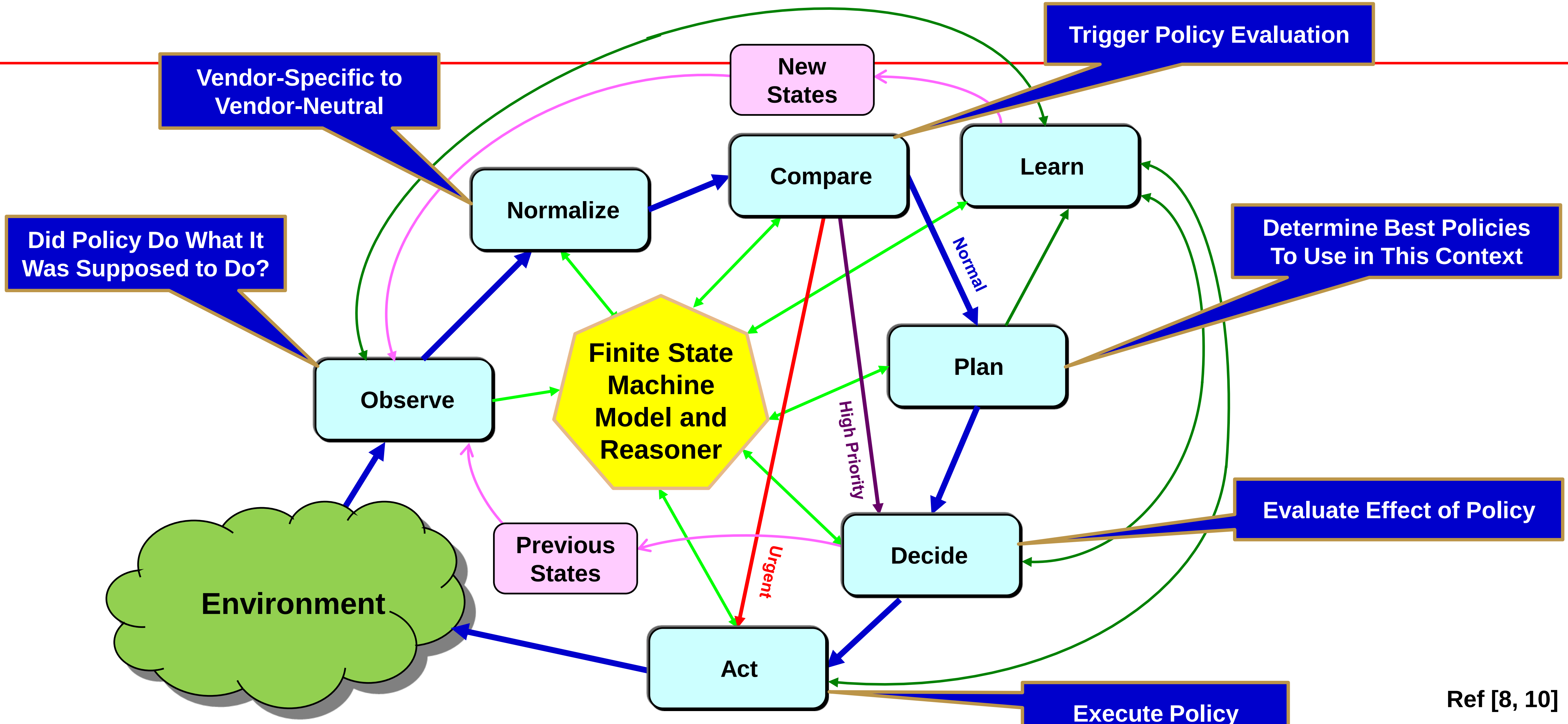


Ref [2]

Importance of Modeling in Policy Management

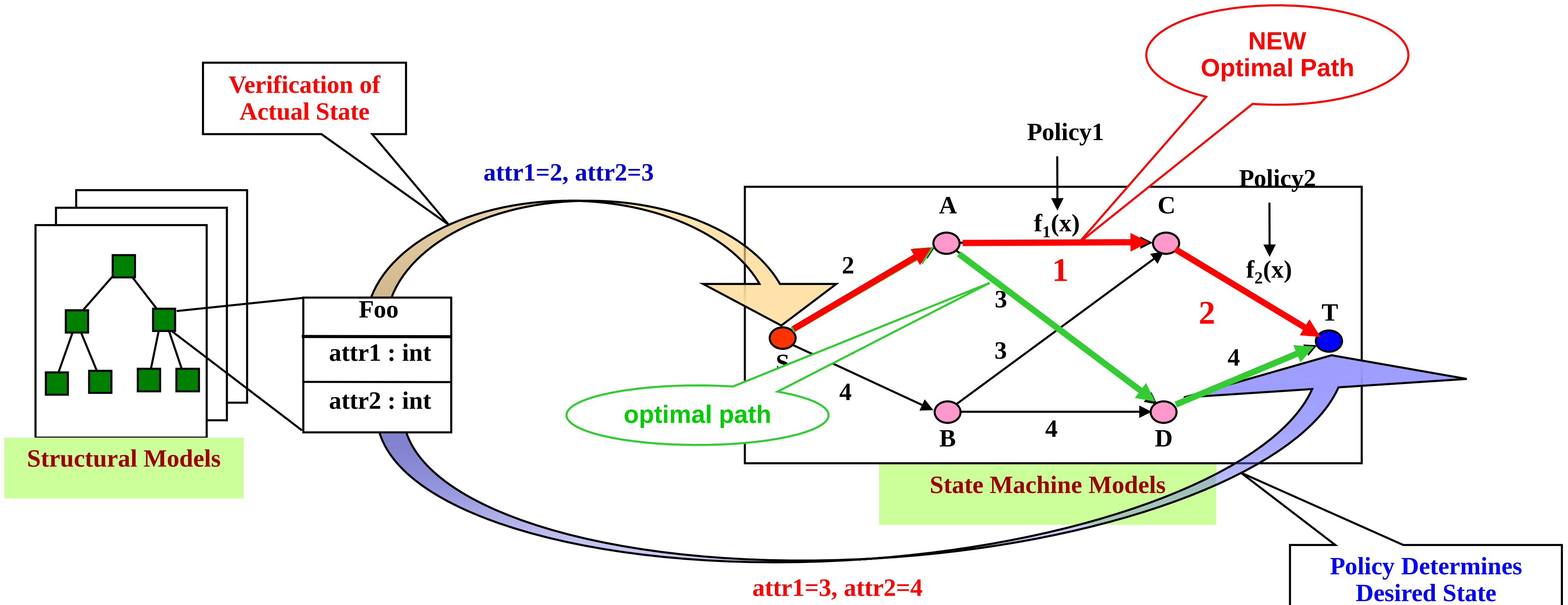


FOCALE Cognition Cycle



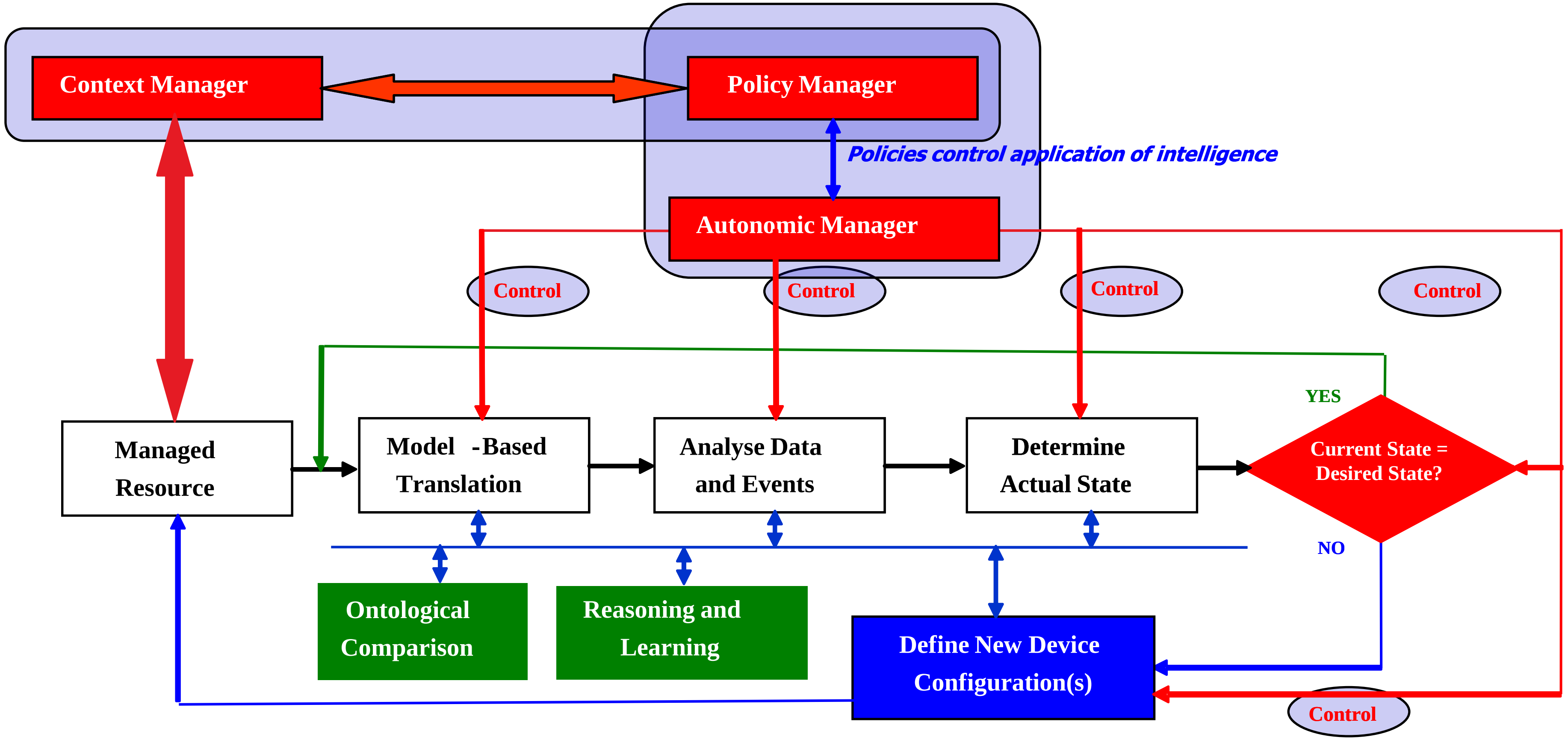
Ref [8, 10]

Policy-driven Behavioral Orchestration



Ref [4]

FOCALE Autonomic Architecture



Autonomic Computing, Policy, and AI

Autonomic Computing

***Self-managing*: configuration, optimization, healing, protection**

•Don't want all behavior hard-coded
•High-level description of how to self-manage

Unified Framework

•Automated decision making
•Rational self-management

Policy

formal behavioral guide

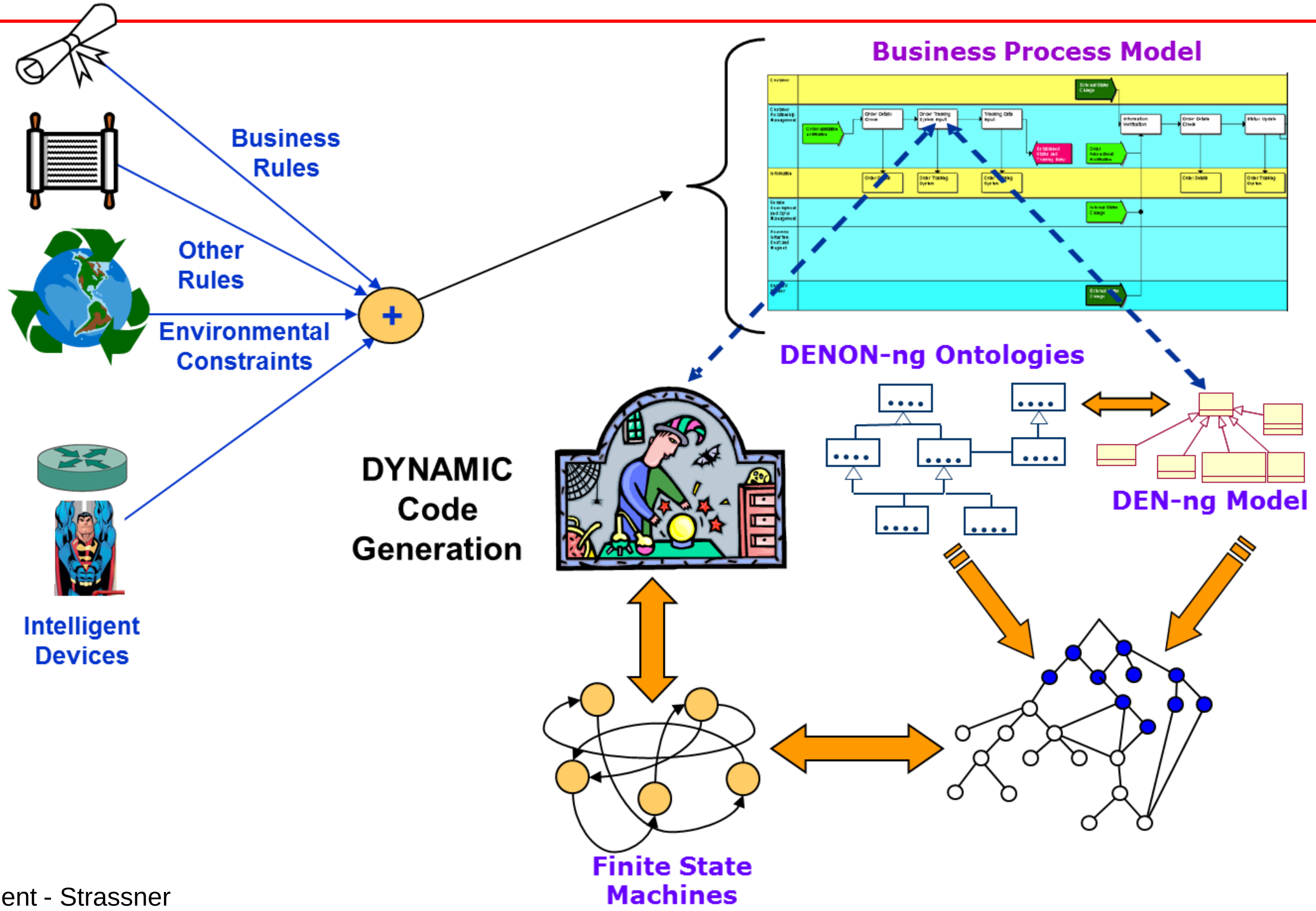
- **Rationality as guide in designing policies**
  - **Imperative**
  - **Goal**
  - **Utility Function**
  - **Declarative**

Artificial Intelligence

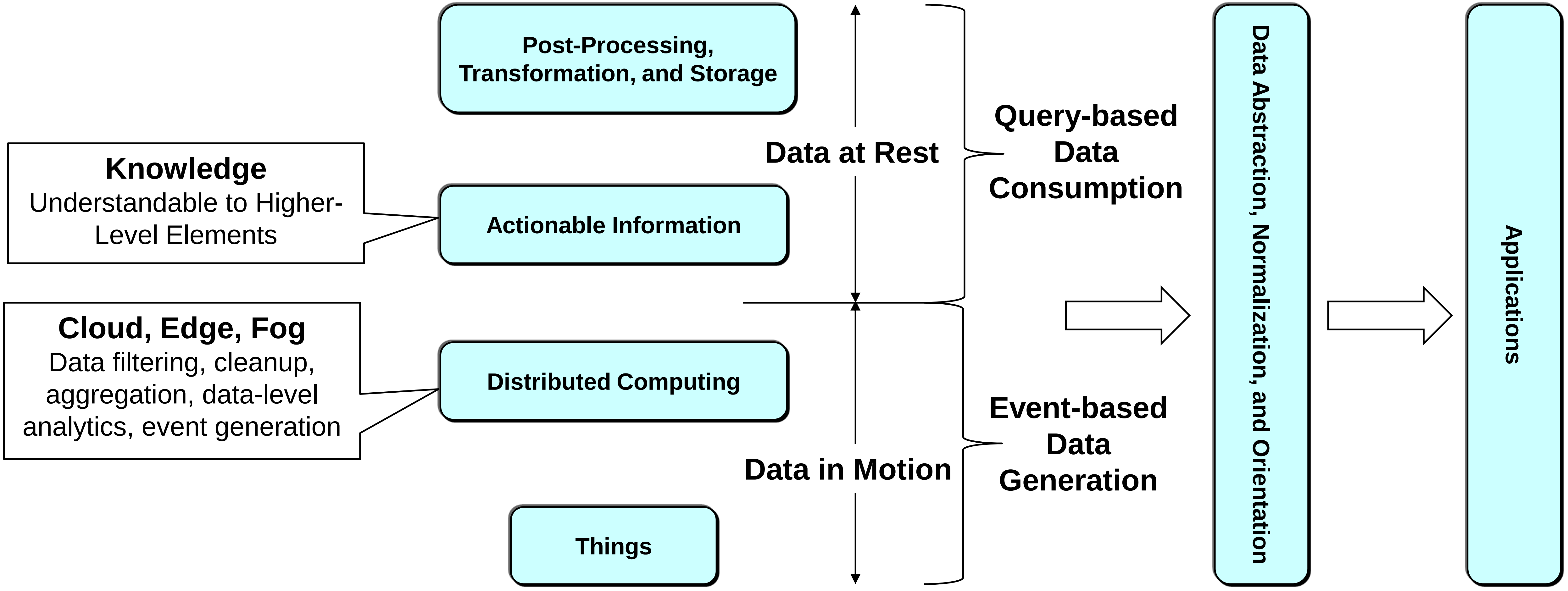
design of rational agents

- **Perceives and acts upon environment**
- **Makes the "right" (best/optimal) decisions**
  - ***with respect to* objective**
  - ***based on* knowledge**

Business to System Interactions



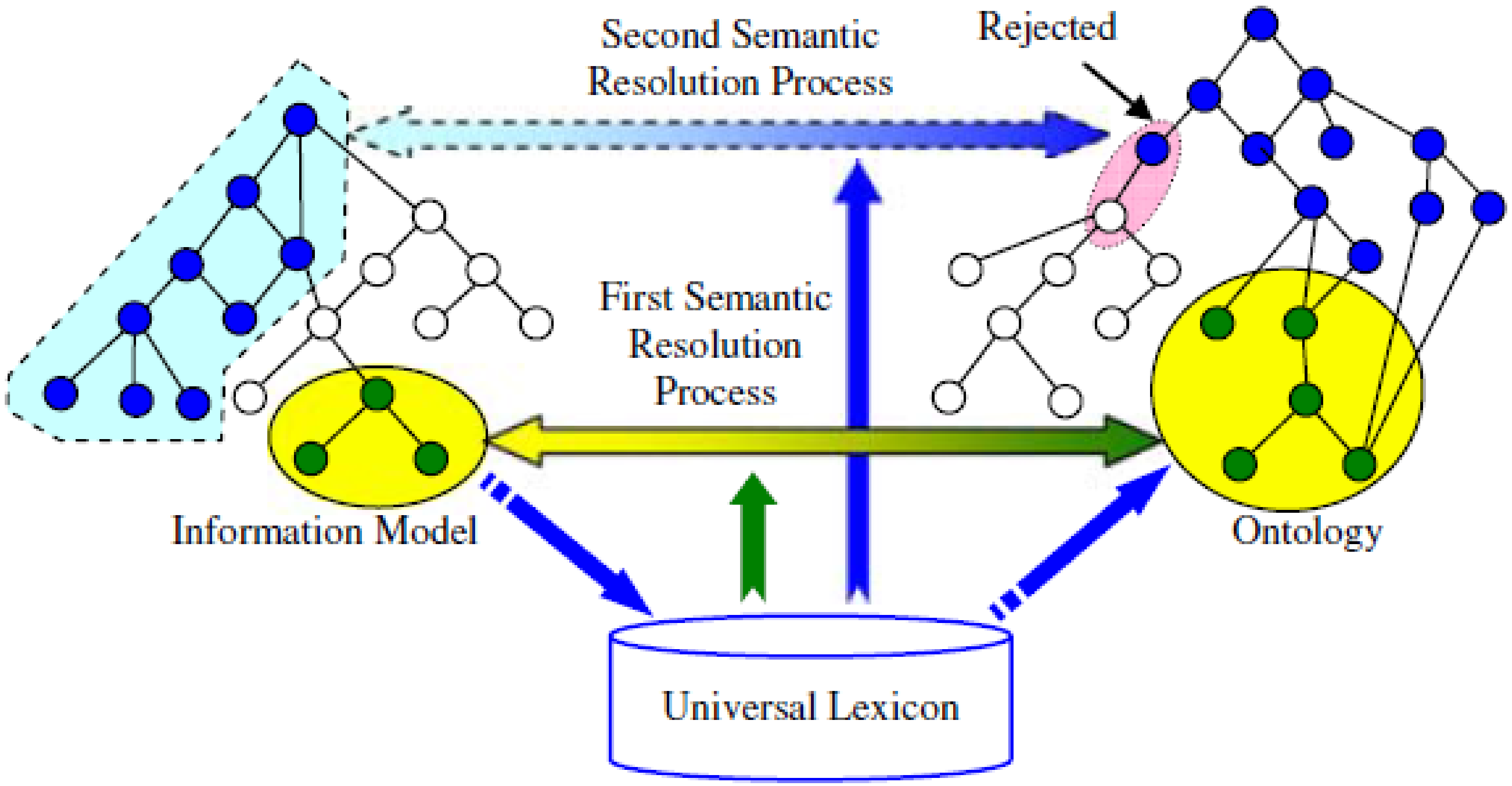
High-Level Semantic Architecture



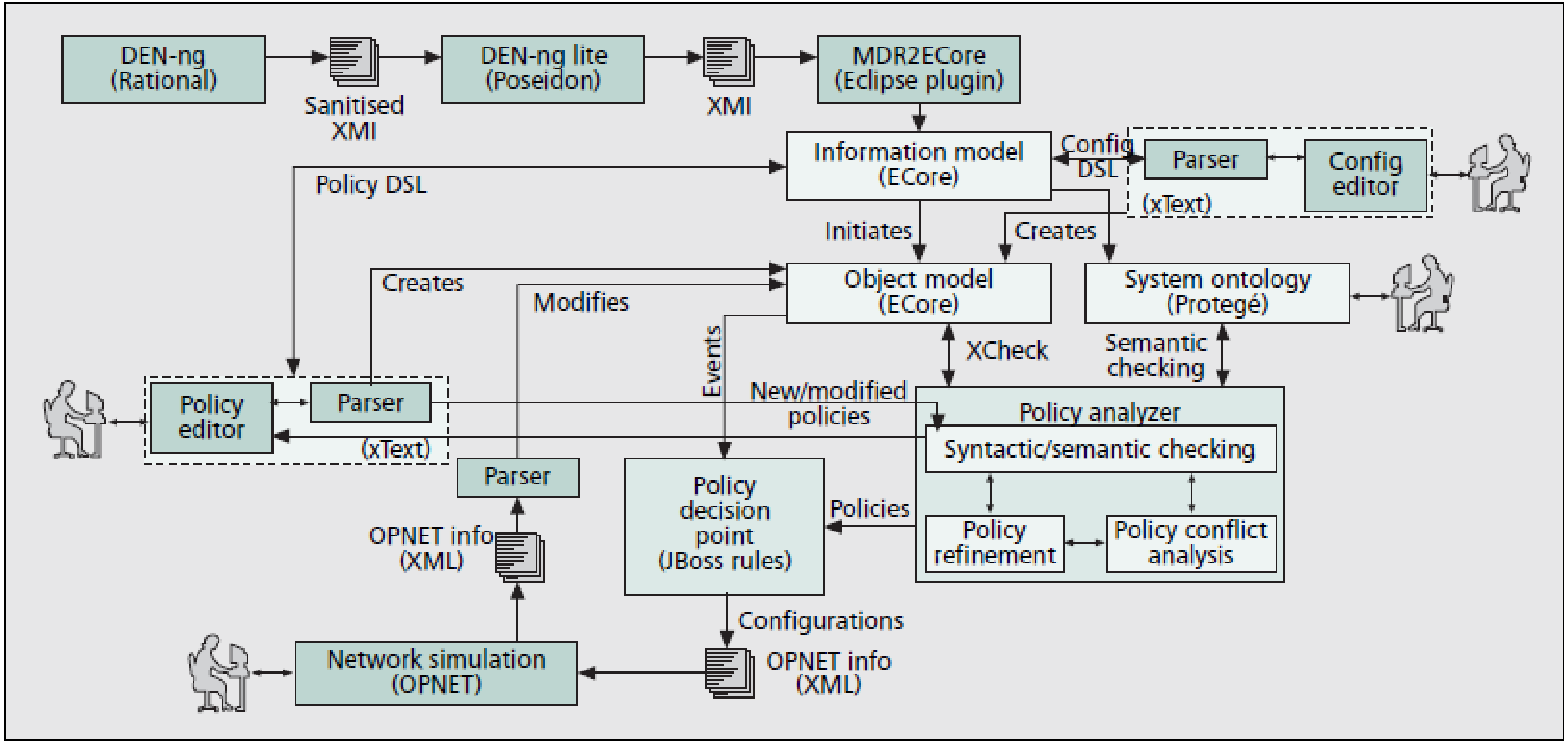
Understanding Network Data

- **What About Data Whose Schema-level Understanding Is Missing**
  – e.g, raw tables, graphs, xml, logs, new machine data that has not been modeled
- **Such Data Needs Semantics for Interpretation**
  – Semantics can be used to “match” unknown data
    ➢ Available from the Web, from domain-specific knowledge bases, and industrial ontologies
  – Different semantic measures provide different levels of confidence
  – If data doesn’t match…
  – …use large background knowledge bases (e.g., Freebase) and relax the level of semantic matching used
  – …but will inevitably have to manually engineer some knowledge bases

Exemplary Semantic Resolution Process



Exemplar Implementation



Ref [7]

Agenda

- Definitions
- Motivation
- Traditional Formulation
- Intending to Introduce Intent
- What the SDOs are Doing (and not Doing)
- Ongoing Research
- **Summary**

Summary

- **Intent Is Currently Poorly Defined**
  - *Hoping* we agree that it is sufficiently abstract as to encourage end-users and application developers who don't know networking to use it to develop policies for network service management
    - ➢ See a demo of a SUPA Policy Engine at BnB on Thursday
- **Intent is ONE TYPE of Policy; it MUST Peacefully Co-Exist with Other Policies**
  - A Policy Continuum enables all constituencies to define policies that can work together
- **Policy Management Architectures are Typically Under-Specified**
  - Policies are key to closing the loop between Business, IT, and the Infrastructure
    - ➢ This requires a comprehensive information model and multiple data models
  - Policy SHOULD be about defining behavior, not changing a line in a config file
  - Lack of true context and semantic reasoning
  - Lack of federation of different policy domains

References

[1] J. Strassner, J. Kephart, “Autonomic Systems and Networks: Theory and Practice”, NOMS 2006 Tutorial

[2] J. Strassner, S. van der Meer, D. O’Sullivan, S. Dobson, “The Use of Context-Aware Policies and Ontologies to Facilitate Business-Aware Network Management”, JNSM (17), pp 255-284, 2009

[3] J. Strassner, J. Halpern, J. Coleman, “Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)”, draft-strassner-supa-generic-policy-info-model-04

[4] J. Strassner, “Policy Based Network Management”, Morgan Kaufman, ISBN 978-1558608597, Sep 2003

[5] M. Sloman, “Policy Driven Management for Distributed Systems”, JNSM, v2, No 4, 1994

[6] K. Barrett, S. Davy, J. Strassner, B. Jennings, S. van der Meer, “Model Based Generation of Integrated Suites of Languages and Tools for Policy Specification, Analysis and Deployment”, IEEE Global Information Infrastructure Symposium, 2007

[7] B. Jennings, S. van der Meer, S. Balasubramaniam, D. Botvich, J. Strassner, M. Ó Foghlú, W. Donnelly, J. Strassner, “Towards Autonomic Management of Communication Networks”, IEEE Communications Magazine, Vol 45., No 10, pp 112-121, Oct 2007

[8] J. Strassner, N. Agoulmine, E. Lehtihet, “FOCALE – A Novel Autonomic Networking Architecture”, International Transactions on Systems, Science, and Applications (ITSSA) Journal, Vol. 3, No 1, pp 64-79, May, 2007

[9] T. Parr, “Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages”, Pragmatic Bookshelf, 2010

[10] J. Strassner, N. Agoulmine, E. Lehtihet, “FOCALE – A Novel Autonomic Networking Architecture”, International Transactions on Systems, Science, and Applications (ITSSA) Journal, Vol. 3, No 1, pp 64-79, May, 2007

[11] S. Davy, B. Jennings, J. Strassner, “The Policy Continuum – Policy Authoring and Conflict Analysis”, Computer Communications Journal, Elsevier, Volume 31, Issue 13, pages 2981-2995, August 2008

[12] J. Strassner, J. Halpern, M. Behringer, “The Use of Control Loops in Autonomic Networking”, draft-strassner-anima-control-loops-01, Nov 2015

[13] J. Strassner, J. Halpern, Q. Wu, “Semantics and the Internet of Things”, draft-strassner-t2trg-semantics-and-iot-00, March 2016

[14] J. Strassner, ed., “ZOOM Policy Model and Architecture Snapshot”, TR235, Release 14.5.1, February 2015

Questions?

"Create like a god. Command like a king. Work like a slave"

- Constantin Brancusi