Towards a Carrier Grade ONAP Platform FCAPS Architectural Evolution

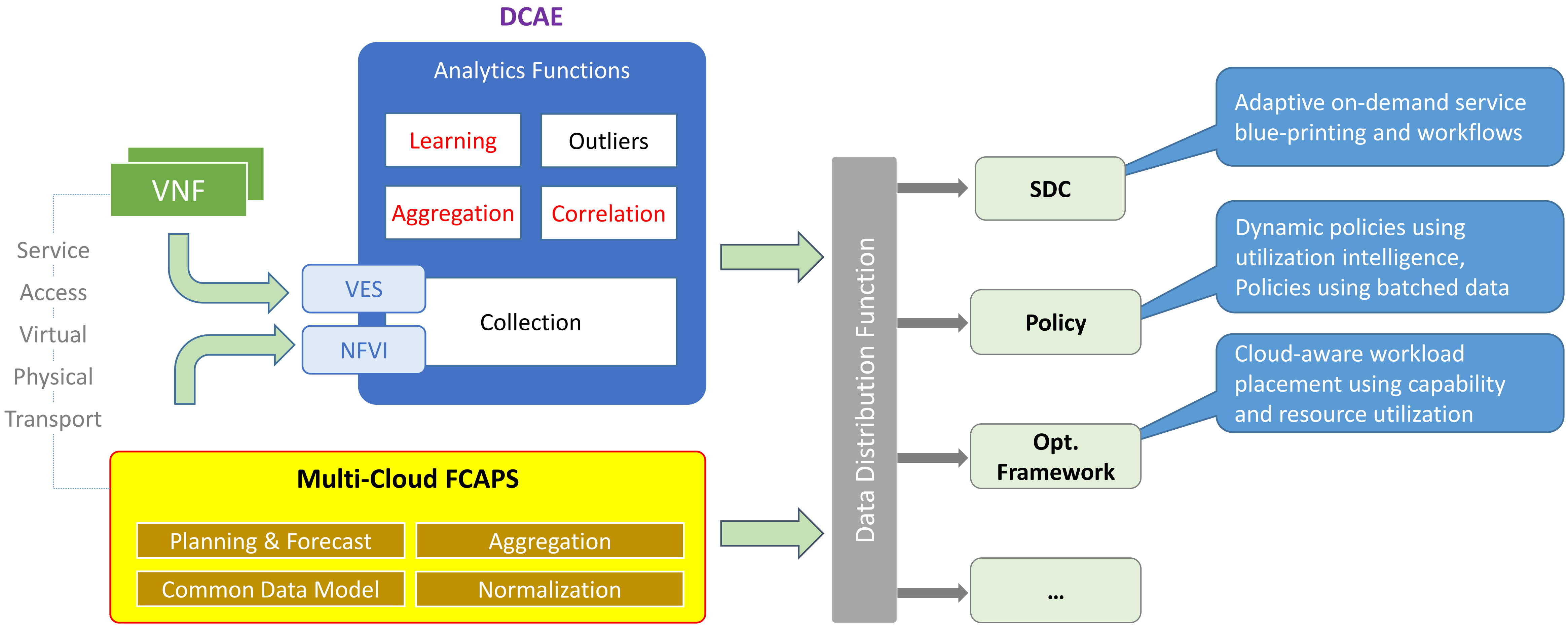
Key Contributors:
Ramki Krishnan, Sumit Verdi, Xinhui Li, Danny Lin, Bin Hu, Gil Hellmann, Bin Yang, Shankar Narayanan, Sastry Isukapalli, Rajesh Gadiyar, Vimal Begwani

Agenda

- Operational Intelligence for Dynamic Orchestration
- MC FCAPS Architecture – Common Data Model & Data Distribution
- MC Impact - VoLTE, vCPE VNF Placement (Homing) Use Case
- MC<->OF Interaction for VNF Placement (Homing)

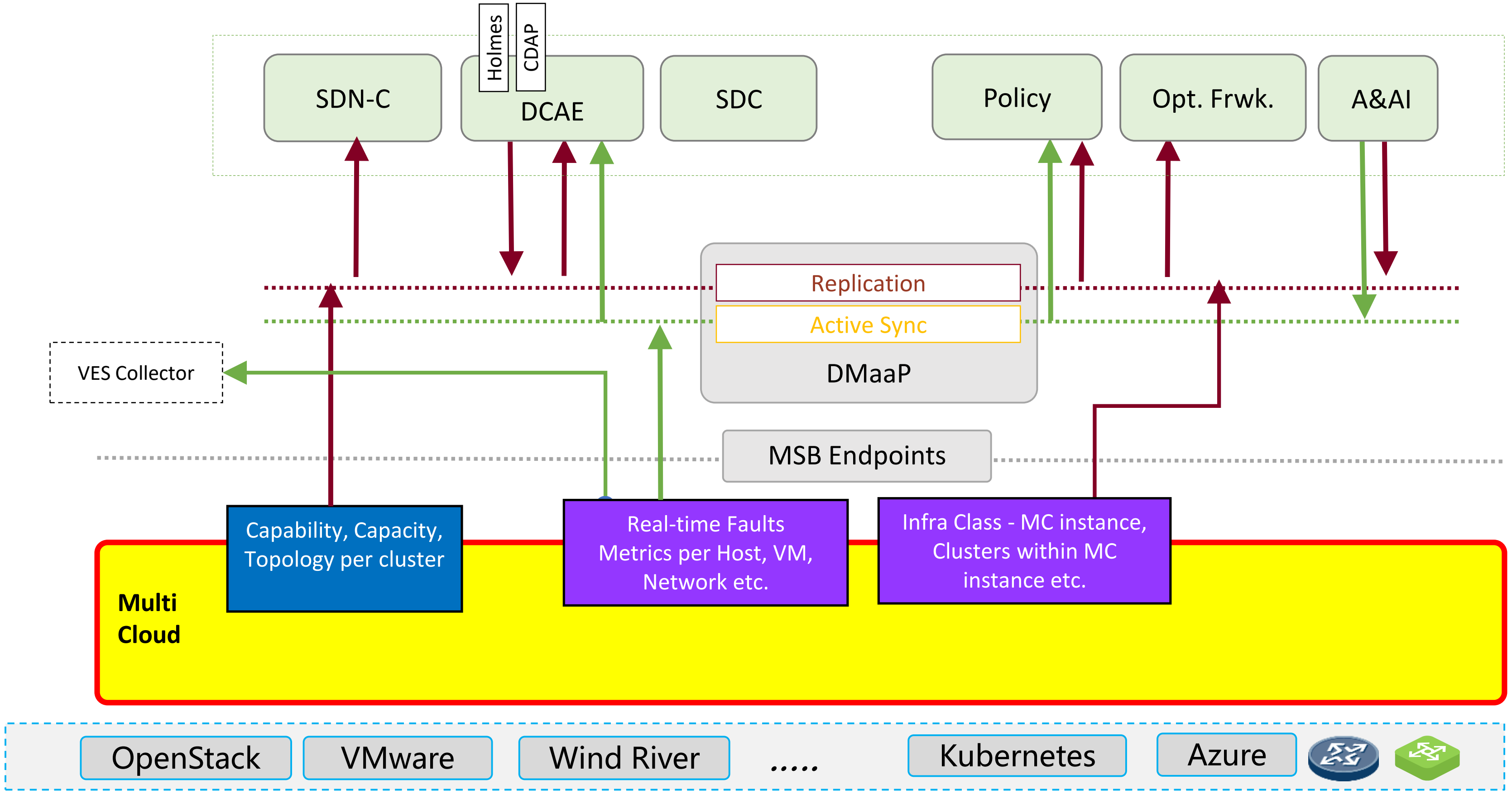
Operational Intelligence for Dynamic Orchestration

Application and Infrastructure correlated context is key…



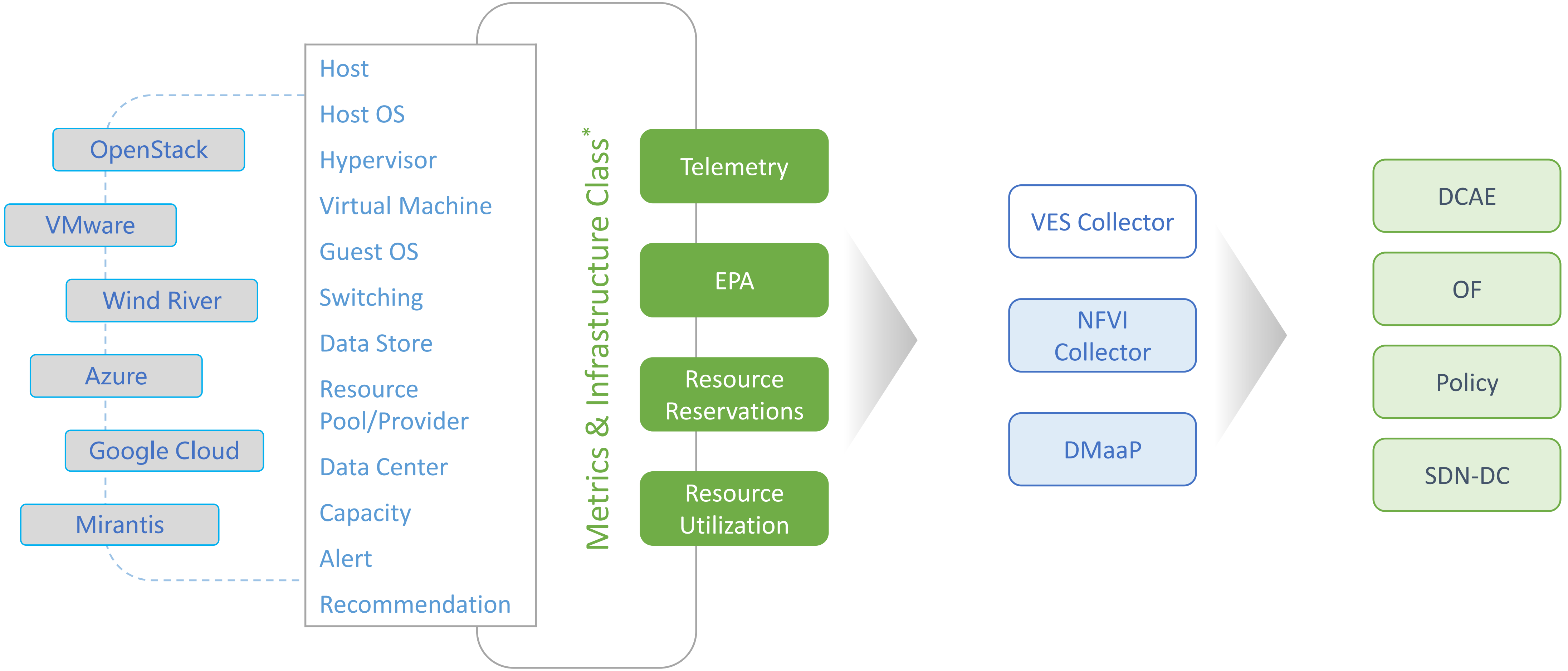
MC FCAPS Architecture

- **Standardized Common Data Model across Cloud Providers -** Various cloud providers have disparate data structures, representations, middle-wares and more for infrastructure telemetry collection and management
- **Composite NFVI Intelligence** at atomic, aggregate and infrastructure capability granularities to enable cloud-aware decisioning by OF, DCAE, SO
- **Enable Continuous Service Deployment** with run-time resource reservations and utilization telemetry
- **FCAPS Data Distribution** to enable simplified and scalable many-many communications using DMaaP pub-sub



FCAPS Common Data Model, Distribution & Integration

*Joint collaboration between VMware, Intel, AT&T, China Mobile, WindRiver



Infrastructure Class Example

Cloud Provider | Multi-Tenancy | Multi-Device Access | U/P Disaggregation | Business Model | Cost

Cloud Agnostic Data Model

Capability

- Hardware encryption
- Hardware transcoding
- NUMA nodes available
- Storage class
- CPU, Memory, NIC class
- WAN interconnects and protocols

Resource Utilization

- Number of hosts currently active
- Number of running VMs
- Total number of vCPU's powered
- VM density per host
- Available bandwidth per host
- NUMA utilization

Resource Reservation

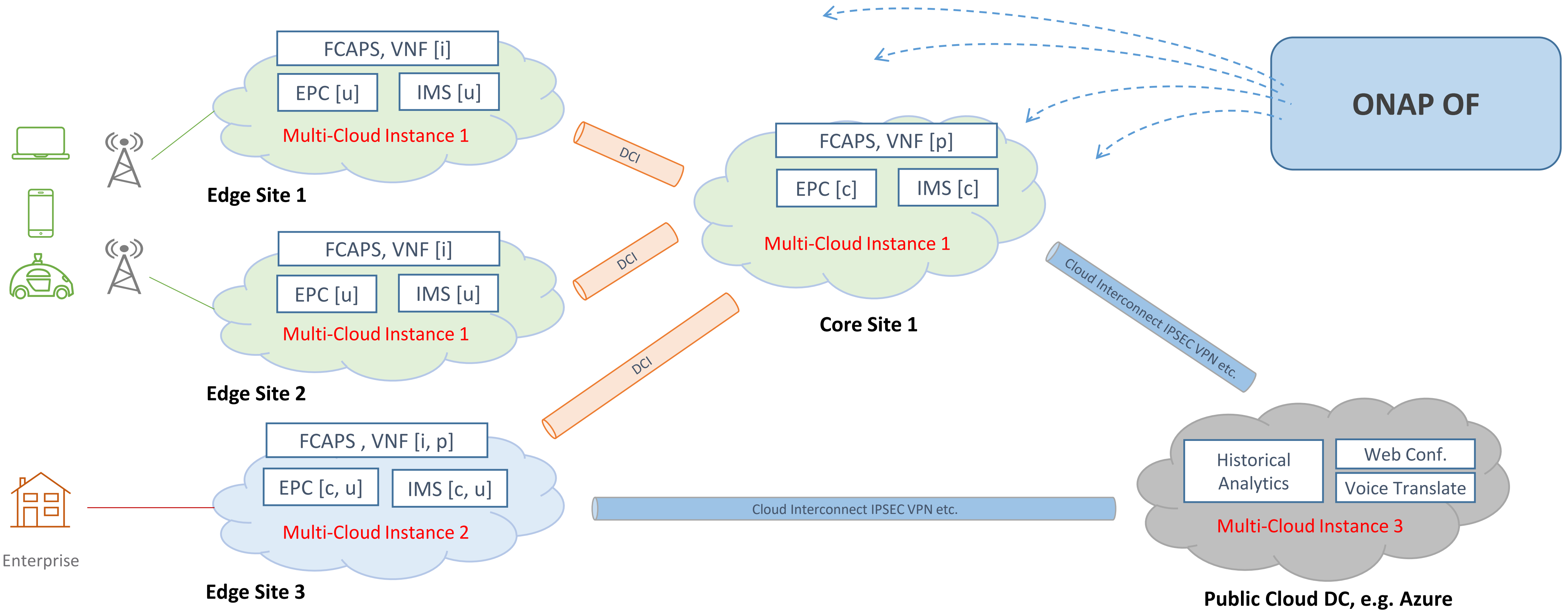
- Number of tenants
- MC instances per data center
- Maximum number of VM's
- Total number of clusters
- Total number of hosts
- Total number of CPU, MEM
- Total number of data stores

Cluster Topology

- Network Fabric type
  - CLOS etc., 2 level, 3 level
- Interconnect BW
  - Max, Min

VoLTE: Distributed DC VNF Placement (Homing) Use Case

Workflow: Continuous Deployment - Day 1 & Beyond



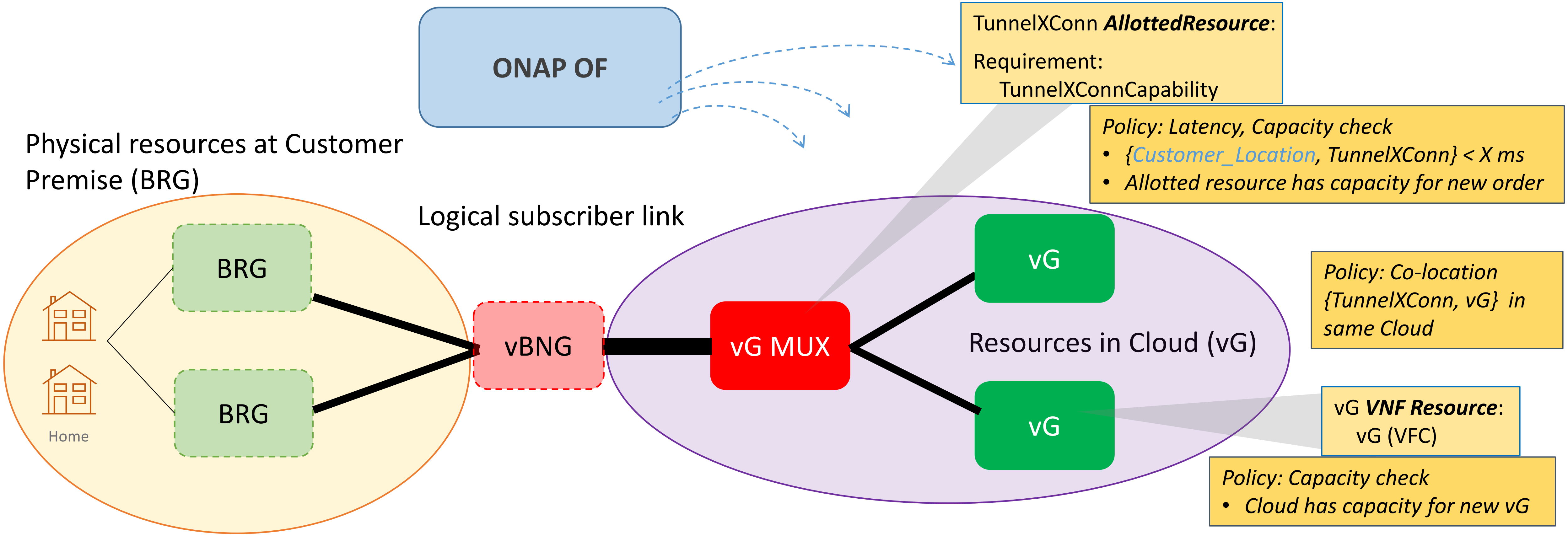
Current ONAP Challenge: Static MC instance selection for workload placement leading to higher cost due to under-utilization or poor application QoE due to over-subscription of infrastructure

Value Proposition: OF to deliver the best VNF placement solution in terms of Cost, Security and Application QoE by dynamically determining the appropriate multi-cloud instances leveraging aggregate infra data from MC

[u] – User Plane
[c] – Control Plane
[i] – Data Ingestion
[p] – Data processing

Residential vCPE: Distributed DC Placement (Homing) Use Case

Workflow: Continuous Deployment - Day 1 & Beyond



BRG – Broadband Residential Gateway

BNG – Broadband Network Gateway

vG – Virtual Gateway

vG Mux – Virtual Gateway Multiplexer

Infrastructure resource

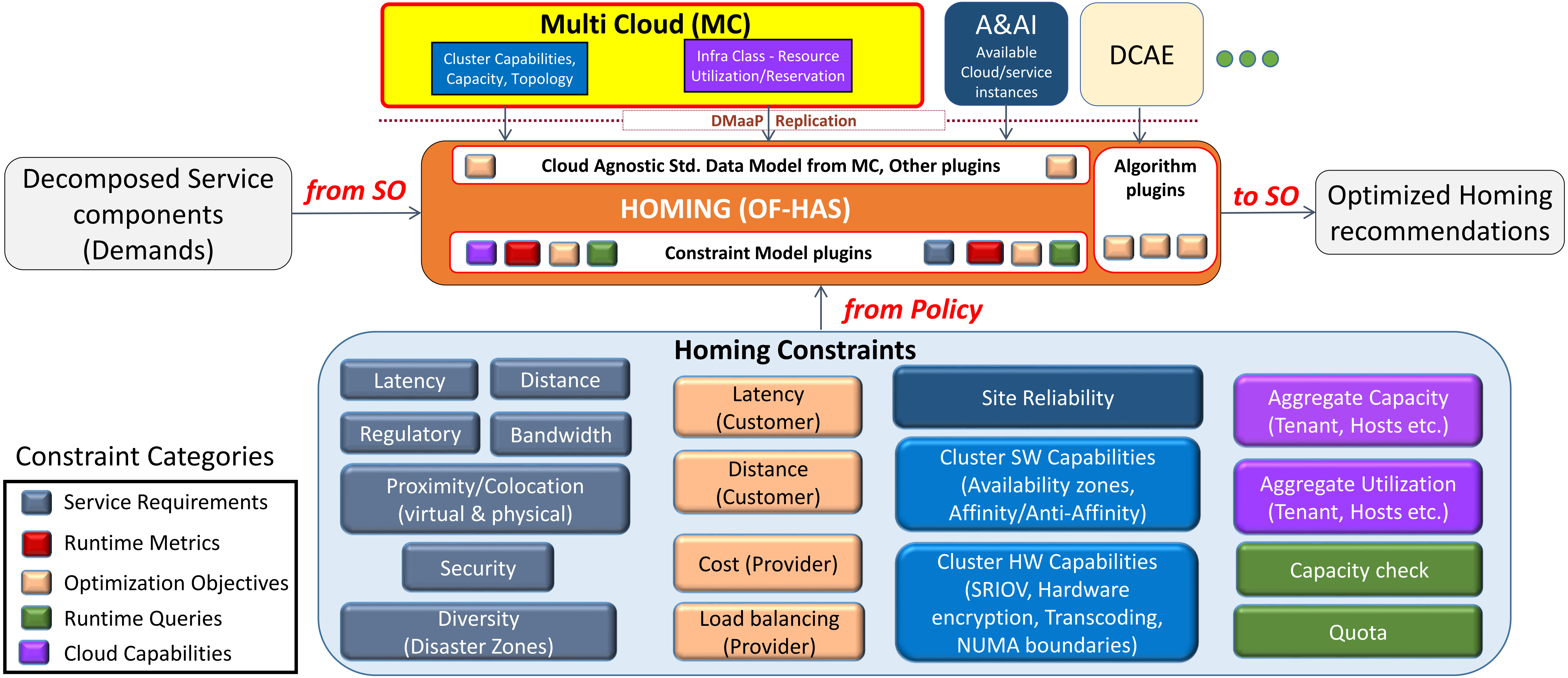
Per-customer resource

ONAP OF targeted for R2 to address R1 use cases (VoLTE, CPE) and upcoming use cases (5G etc.)

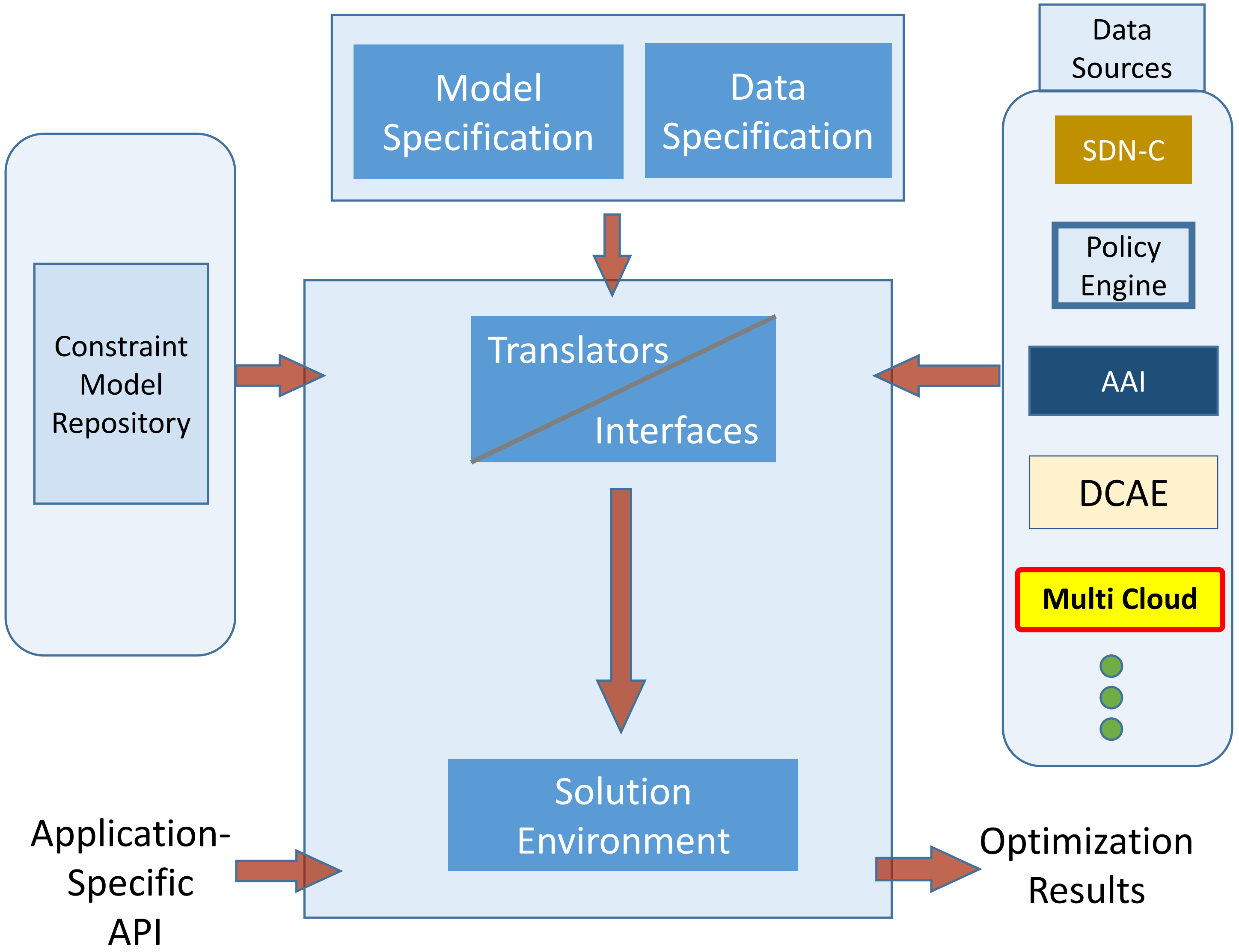
MC <-> OF Interaction

Key Contributors:
Shankar Narayanan, Sastry Isukapalli, Ramki Krishnan

Policy Driven Homing



OF Model-Driven Approach



Goal: provide a platform that facilitates model-driven optimization

- Model-driven (Declarative)
- Library of building blocks
  (as SDC models, policy models, etc.)
- Reuse models from OF-contrib library

- Recipes for model composition
- Adapt at operation time (no new code)
  (new constraints, objectives, runtime flags)

- Op-ex benefits – reducing software dev costs
- Rapid analyses – what-if scenarios via config

- Platform can seamlessly provide new functionality/advances to application

ONAP-OF Based on Standardized Constraint Modeling

ONAP-OF Contributions

- **Data Interfaces**
- **Model/Constraint Translators**
- **Building Blocks**
- **Recipes and Knowledge Base**
- **Operational Environment**

Available Extensions

- Contributed Models
- Global Constraint Catalog
- Stochastic Minizinc – Uncertainty Considerations
- MiningZinc – Constraint-Based Mining
- MiniBrass – Soft Constraints

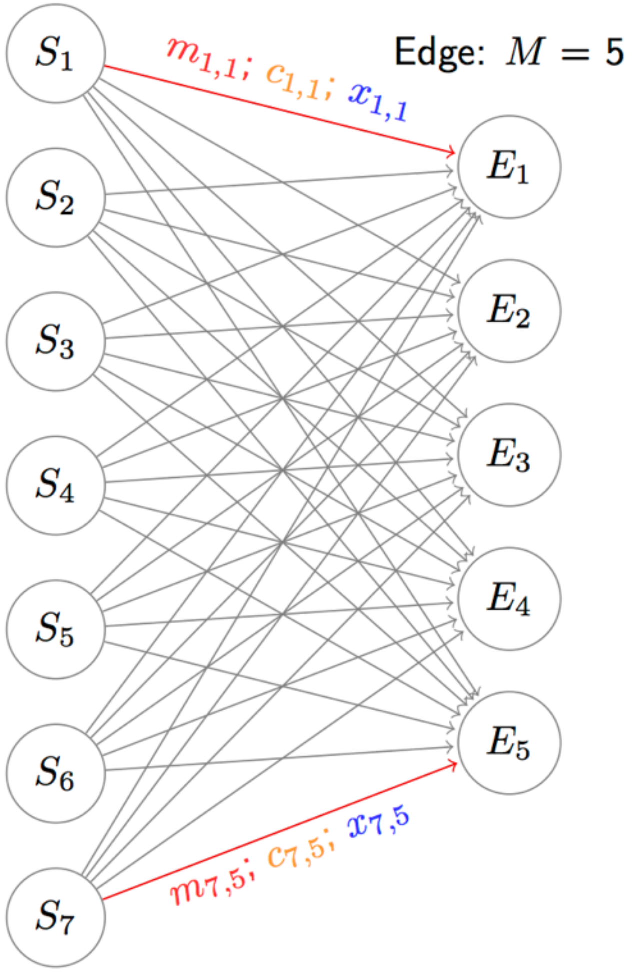
Available Technology

- LibMzn (Embeddable Library)
- Minizinc Standard Library
- MiniZinc Model → Flat Zinc ← Data (dzn format)
- Flat Zinc → solvers: ECLiPSe, CHOCO, Chuffed, IBM CPLEX, COIN|OR

BACKUP

Example Problem: Budget-Constrained Max Flow

Router: $N = 7$



Terms

x_{ij}	Utilization of link $S_i \to E_j$
c_{ij}	Cost/unit for using link $S_i \to E_j$
m_{ij}	Maximum bandwith of link $S_i \to E_j$
p_i	Maximum amount of traffic from node S_i
q_j	Maximum amount of traffic to node E_j
B	Budgeted funds

Objective

Maximize: $\sum_{j=1}^{M} \sum_{j=1}^{M} x_{ij}$

Constraints

$0 \leq x_{ij} \leq m_{ij} \quad i = 1, \ldots, N \quad j = 1, \ldots, M$ (flow limits)

$\sum_{i=1}^{N} x_{ij} \leq q_j \quad \forall j \in 1, \ldots, M$ (node capacity)

$\sum_{j=1}^{M} x_{ij} \leq p_i \quad \forall i \in 1, \ldots, N$ (node capacity)

$\sum_{j=1}^{M} \sum_{j=1}^{M} c_{ij} x_{ij} \leq B$ (budget)

Example Problem: Budget-Constrained Max Flow

Model

```
int: N;  % input nodes
int: M;  % output nodes
int: maxbw; % max bandwidth (for convenience)
float: budget;

set of int: inNodes = 1..N;
set of int: outNodes = 1..M;

array[inNodes] of int: inCap;  % capacities for input nodes
array[outNodes] of int: outCap;  % capacity for output nodes

array[inNodes, outNodes] of int: bw;  % max bandwidth of link
array[inNodes, outNodes] of float: cost;  % unit cost for the link
array[inNodes, outNodes] of var 0..maxbw: x;  % amount through this link

constraint forall (i in inNodes) (sum (j in outNodes) (x[i,j]) <= inCap[i]);
constraint forall (j in outNodes) (sum (i in inNodes) (x[i,j]) <= outCap[j]);
constraint forall (i in inNodes, j in outNodes) (x[i,j] <= bw[i,j]);

constraint sum (i in inNodes, j in outNodes) (x[i,j] * cost[i,j]) <= budget;

% another "stringent" service-specific policy
constraint sum (i in inNodes, j in outNodes) (x[i,j] * cost[i,j]) <= 0.8 * budget;

% each link cannot have more than 20% of traffic from a customer
var flow = sum (i in inNodes, j in outNodes) (x[i,j]);
constraint forall (i in inNodes, j in outNodes) (x[i,j] <= 0.2 * flow);

solve maximize sum (i in inNodes, j in outNodes) (x[i,j]);
```

Constraints from model designer

Constraints from developer or from service provider/vendor (can be from SDC or Policy Engine)

Data Template

```
N = {{inNodes}};
M = {{outNodes}};
maxbw = {{ max(max(bw) }};  % this can be automatically inferred anyway
budget = {{ budget }};

inCap = {{ inCap }};  % arrays in jinja templates and mzn have same format

outCap = {{ outCap }};

{% macro matrix(v) -%}
"[| + "\n|".join("{}".format(y)[1:-1] for y in x) + "|]"
{%- endmacro %}

bw = {{ matrix(bw) }} |];

cost = {{ matrix(cost) }};
```

Templating Support from OSDF

Macros from OSDF

Data from A&AI, SDN, DCAE or Multi-Cloud