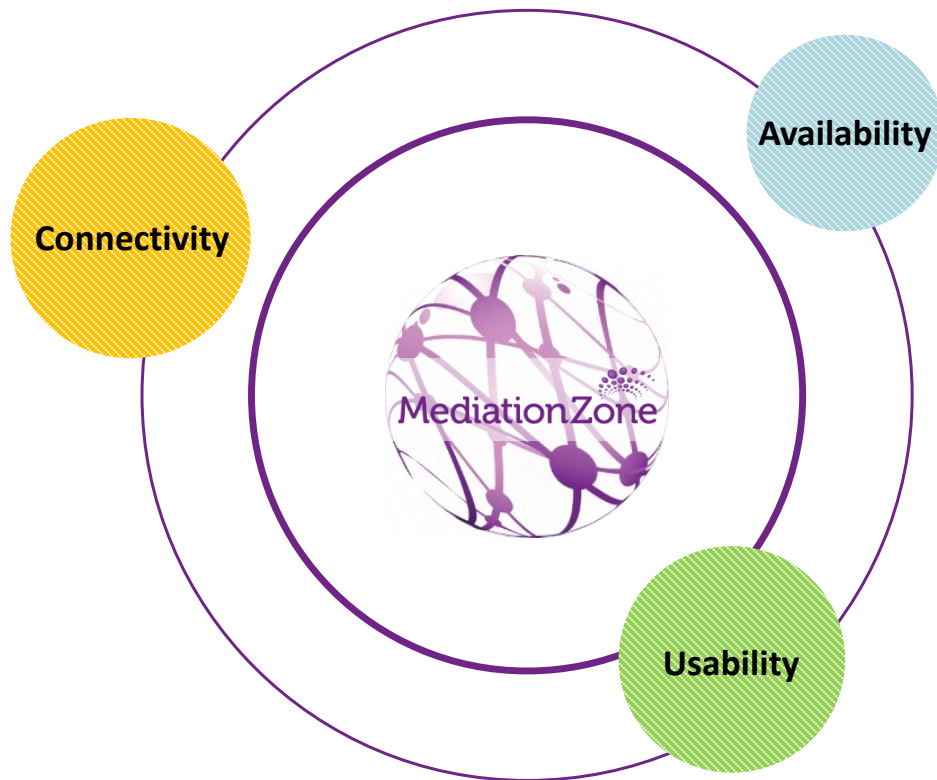




MediationZone 7.0

By Markus Henriks & Irene Gonzalvez

Development themes





Availability

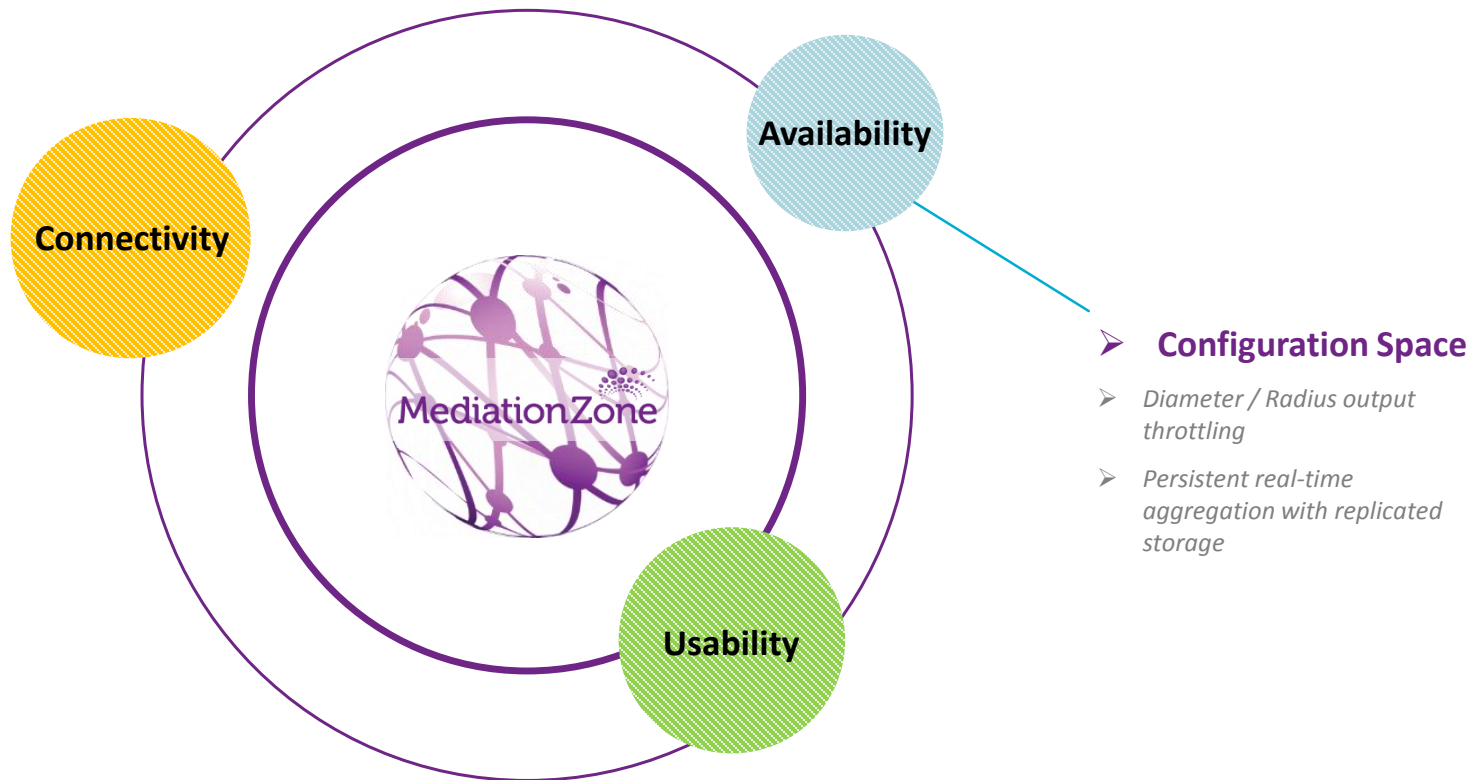
- Configuration Space
- Diameter / Radius output throttling
- Persistent real-time aggregation with replicated storage

Connectivity

- HFS (Hadoop), Amazon S3, SMPP, FTAM/ISODE and HiCAP agents
- RESTful/JSON support
- Google Protocol Buffers format

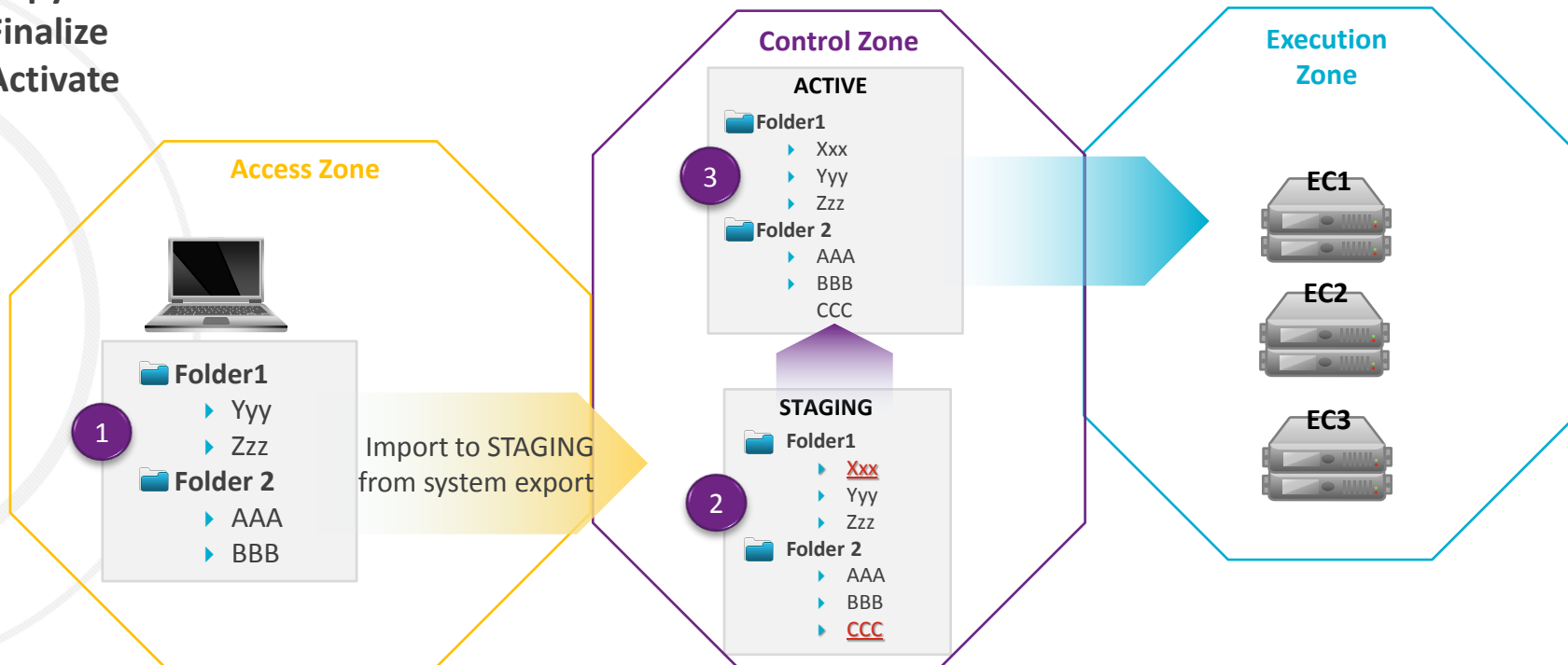
Usability

- Auto generated documentation
- New APL editor including auto completion
- APL for/foreach loop



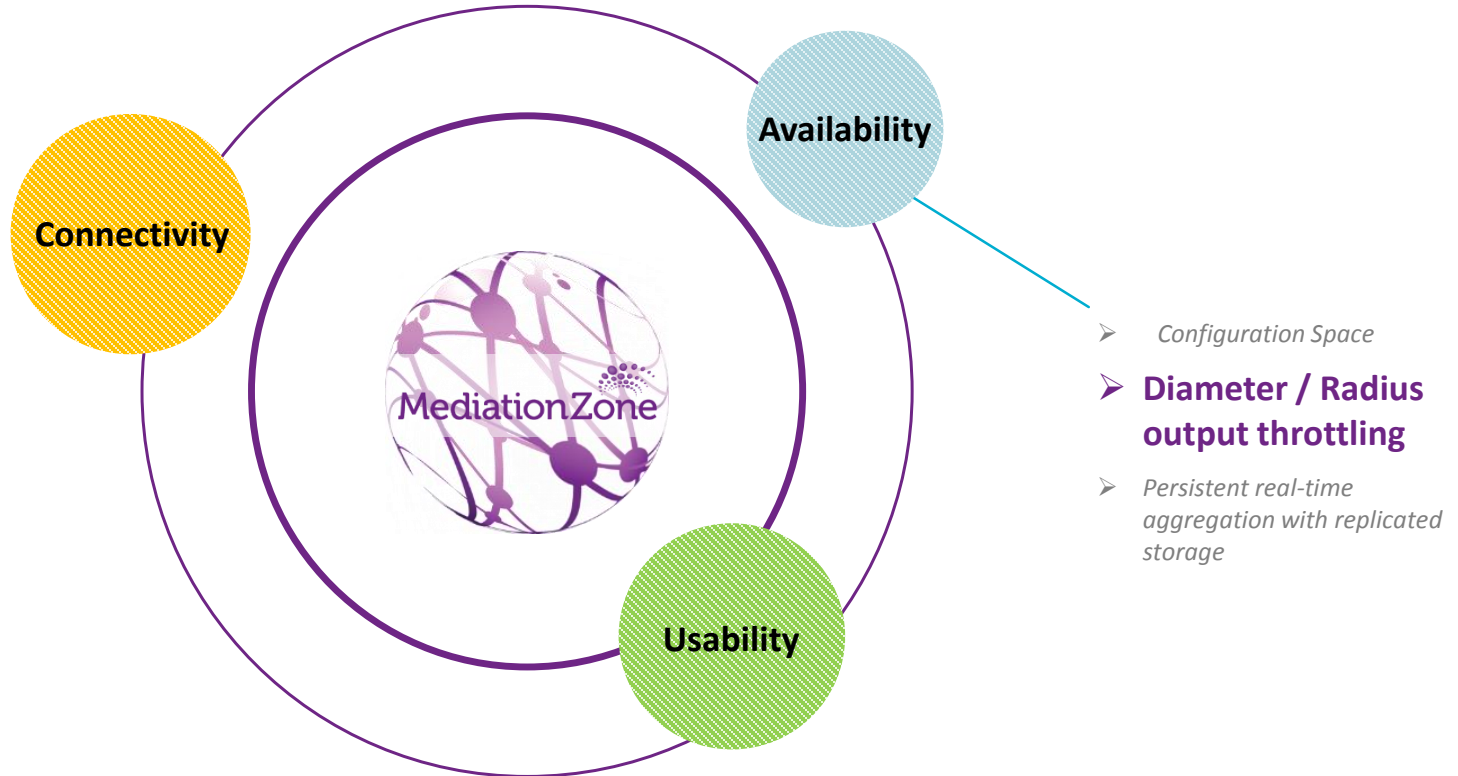
Configuration Space

1. Copy
2. Finalize
3. Activate

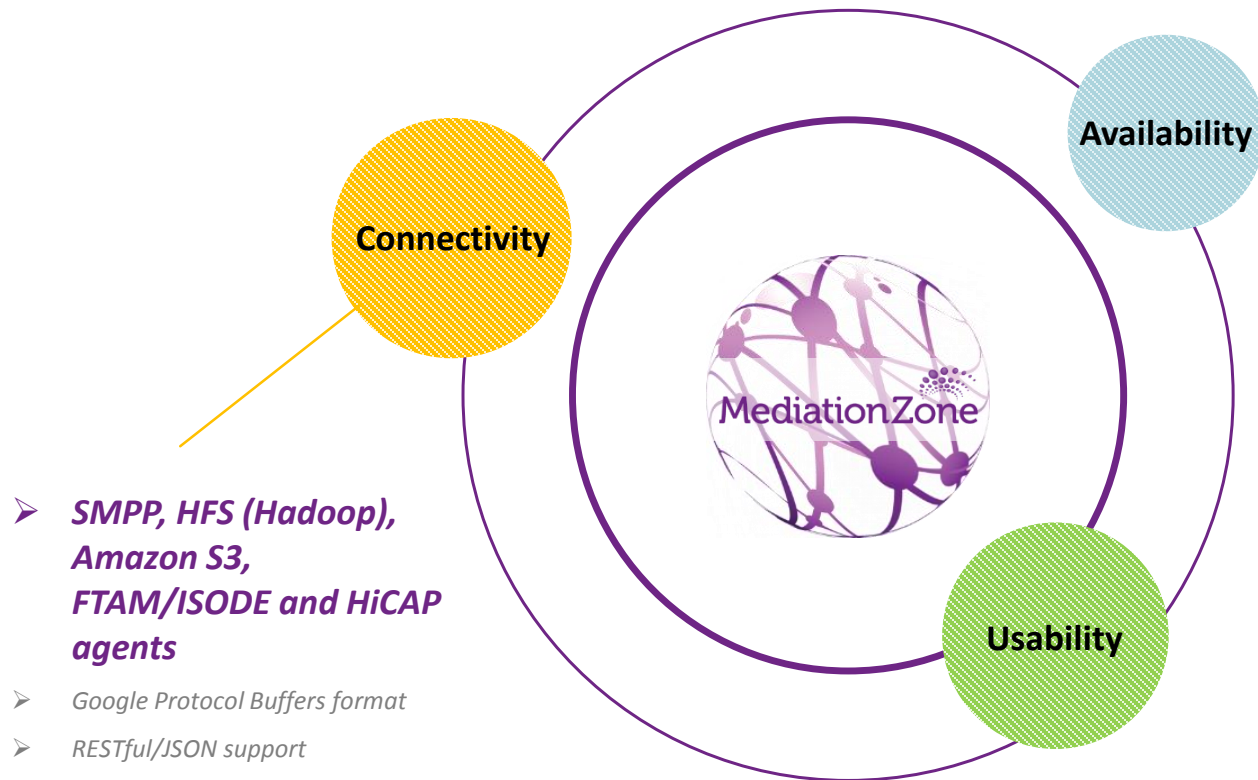


What has been changed in MediationZone?

- Major changes internally
- New MZSH commands
- Select which space to use in GUI and MZSH



- MediationZone has excellent performance
 - Often downstream systems can not handle the load
- Protect downstream systems
 - Outbound user defined throttling of requests to downstream systems
 - Threshold configuration per downstream instance
 - Available in Diameter Routing Profile and Radius Forwarding agent

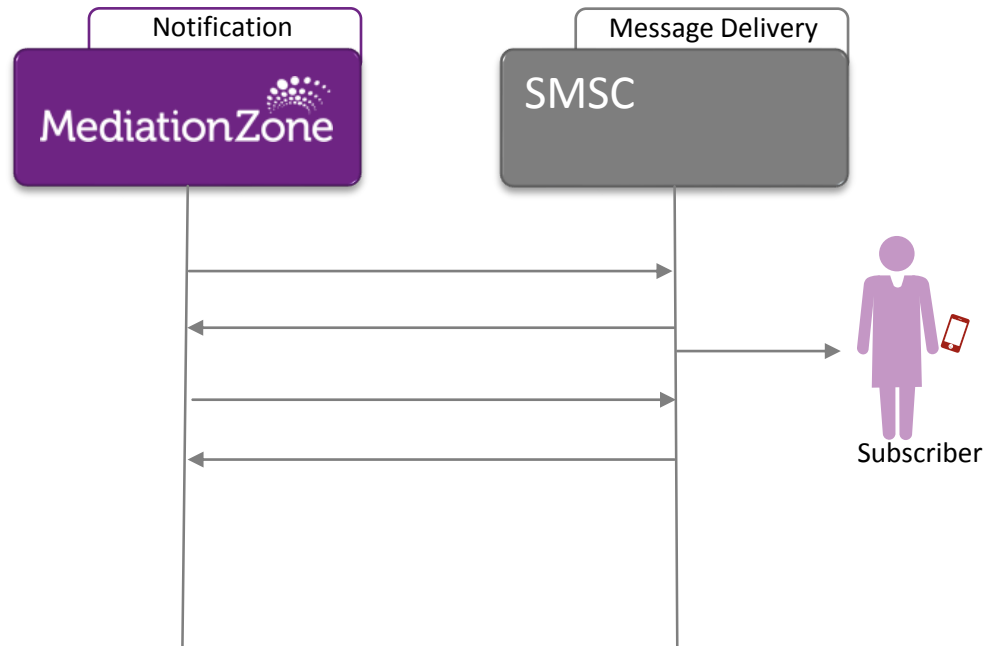


SMPP collection and forwarding

- Support for sending and receiving SMS over the SMPP protocol
- SMPP Receiver (receive SMS)
- SMPP Transmitter (send SMS)

Business Value:

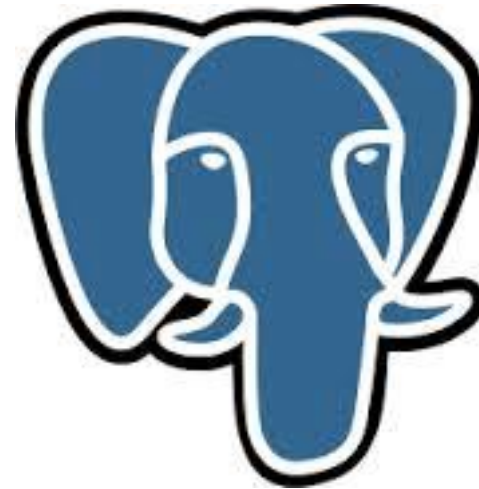
- Lower integration cost
- End user transparency
- Notification management using MZ



- DB profile option
- Allow access to PostgreSQL 9.3 Databases
- Using SQL Collection and SQL Forwarding

Business Value:

- More Integration options



FTAM over TCP/IP (RFC1006)

- ISO Transport Service on top of TCP
- Offer low cost FTAM Solution
- Less 3PP dependent using ISODE framework
- Using FTAM over TCP/IP
 - RFC-1006

Business Value:

- Low TCO legacy connectivity



HiCAP connectivity

- Support for Lucent High Capacity Automatic Message Accounting Protocol (HiCAP) AMATS protocol
- Collection agent through RPC
- Modeled after AMATPS
- Insert into a wf

Business Value:

- Legacy connectivity

Agent - HiCAP_1 (New Workflow [1])

Name HiCAP_1

HiCAP

Connection File Polling Trace

Host

DCF

Timeout (s) 120

Password

Sensor Id 0

Sensor Type 0

Sending Unit 0

Connect Session Retries 5

Connect Sleep Time (ms) 10000

Files to Test 3

Reset Server Password arpcs01

Reset Server Sleep Time (s) 300

OK Cancel Help

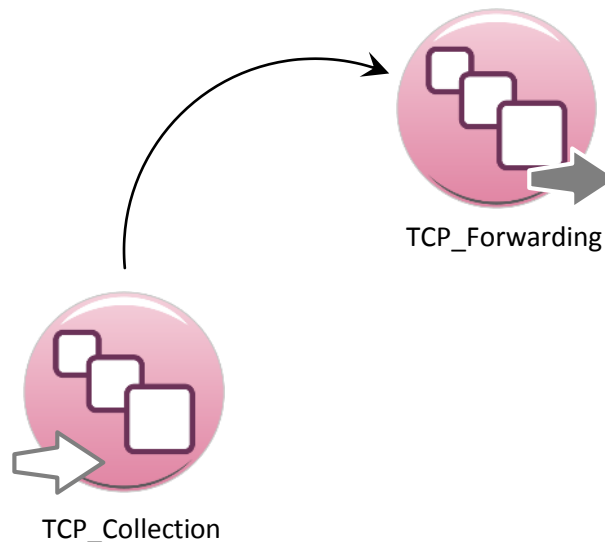


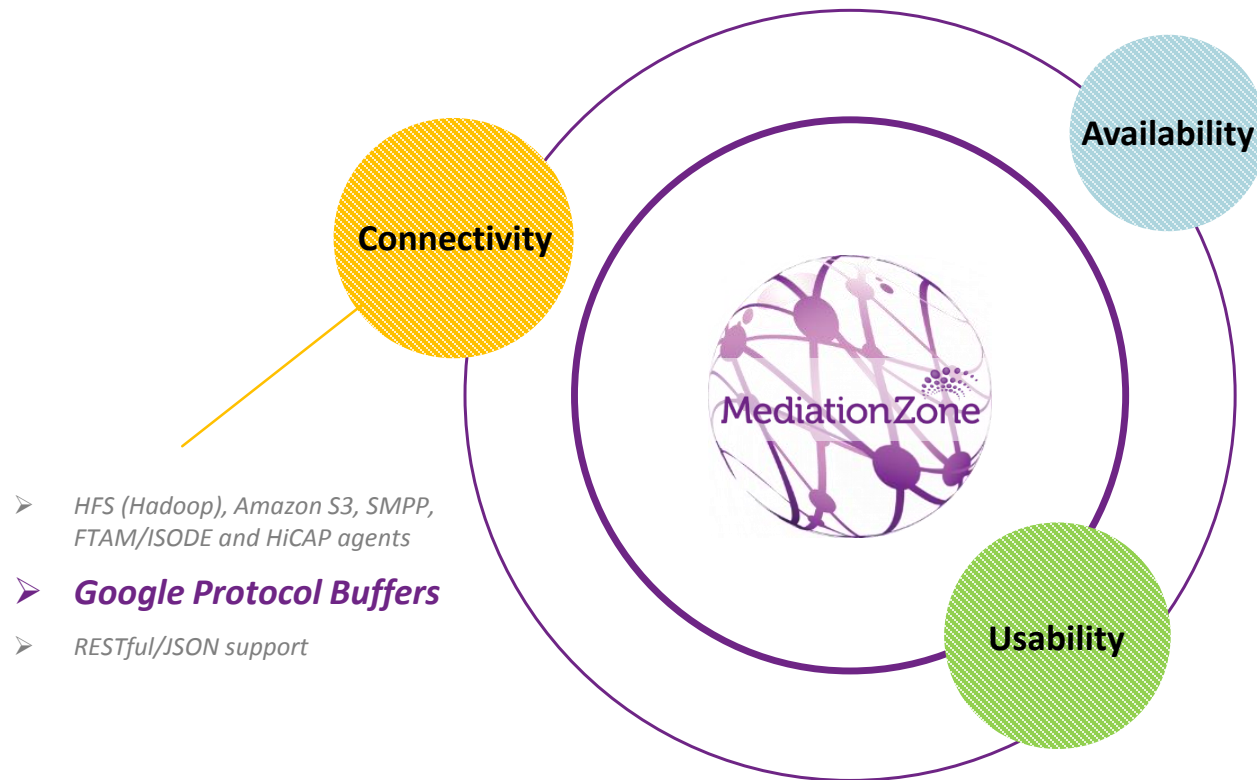
TCP/IP forwarding agent

- New generic agent to send data over TCP/IP to external systems
- Bi-directional support
- Support for multiple receiving agents
- Complement to our TCP/IP collector

Business Value:

- Proxy for any type of TCP traffic





Google's Protocol Buffers

- Support for decoding/encoding of Google's protocol buffers in Ultra
- Language-neutral, platform-neutral, extensible mechanism for serializing structured data
- Think XML, but smaller, faster and simpler

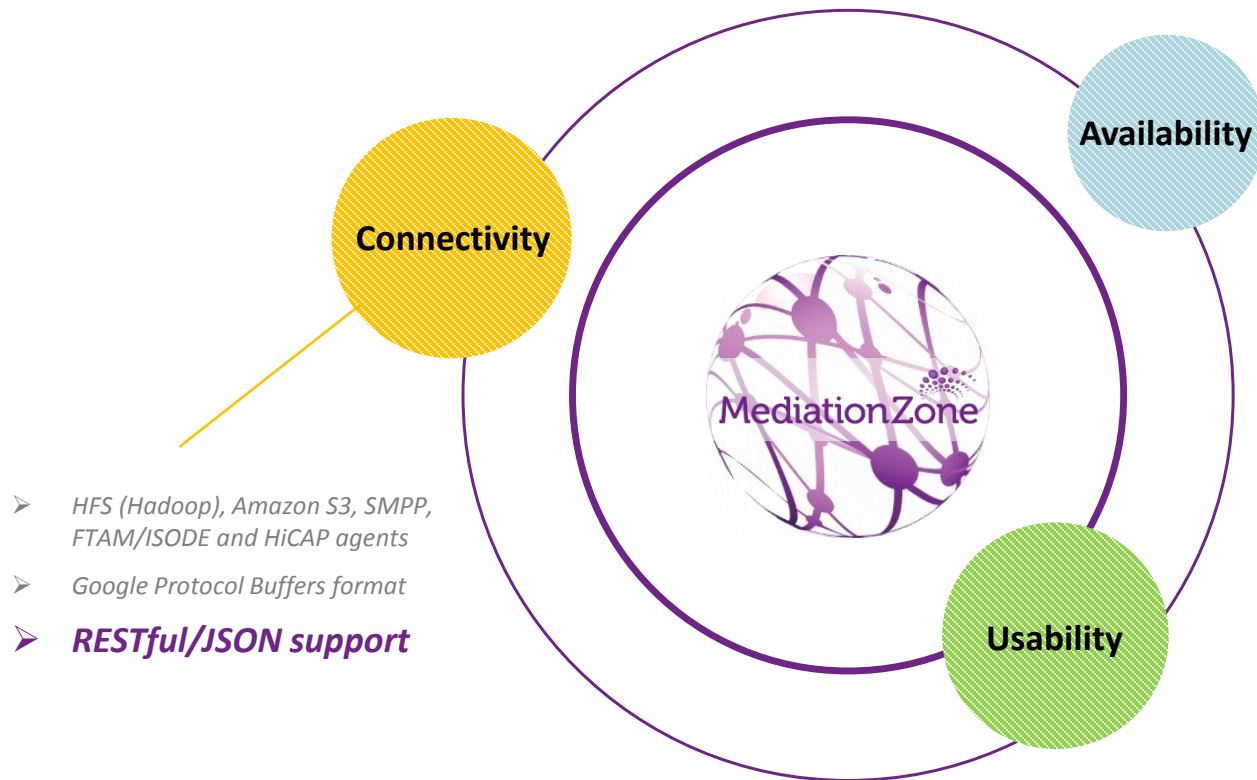
Business Value:

- Additional services and formats supported



```
message DataRequest {  
    required string destination = 1;  
    optional int32 code = 2;  
    optional int32 number = 3;  
}
```

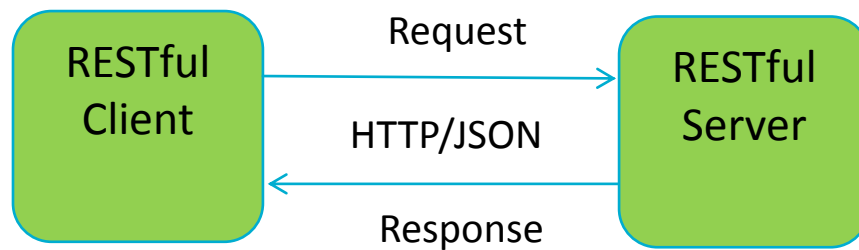
Same way as XML schema, copy/paste of GPB schema into Ultra Editor



- Optional APL functions for both HTTP and JSON
- Support for RESTful web services
 - Often used for provisioning
- JSON or XML

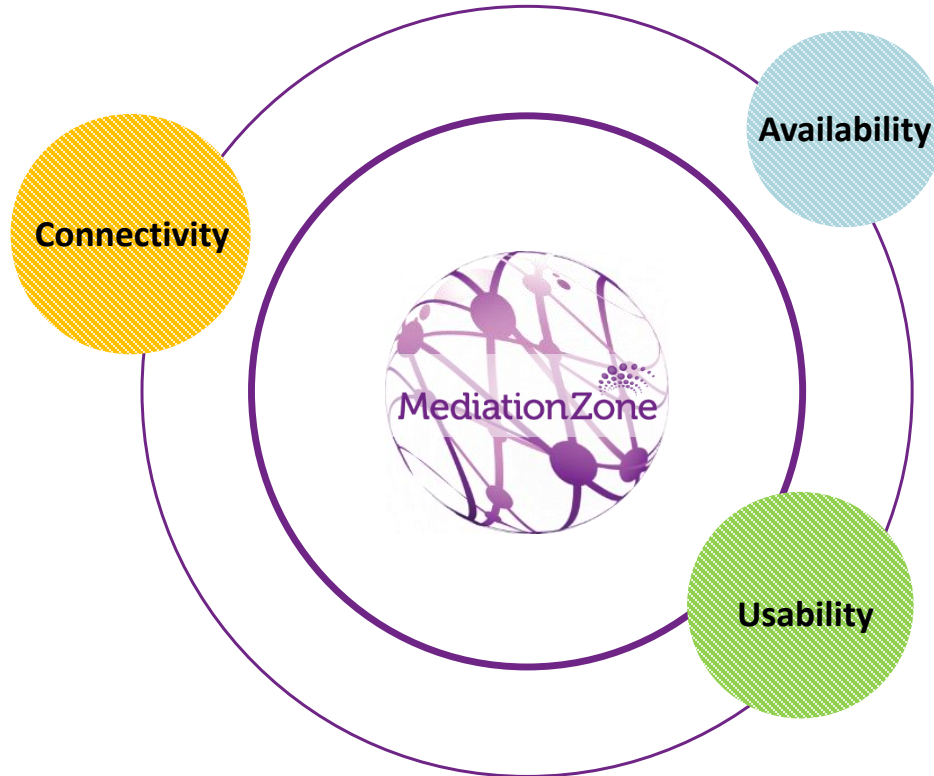
Business Value:

- Integration with RESTful web services



GET /customers/37283

```
{ id : 37283, profile: 123, value : "OCS1" }
```



- **Auto generated documentation**
- *New APL editor including auto completion*
- *APL for/foreach loop*

We need to collect data from our
200 Network Elements,
producing data in *50 formats*, forwarding
data to our *3 billing systems*, *revenue*
assurance and *fraud*.



No, problem!
We are the experts!
We can develop it for you and
then **you can support** it.
You will become **self sufficient**.







3 Months later

Here you are, we have developed your configuration, helped you to go into production. Now you are **self sufficient** and can do what ever you want.



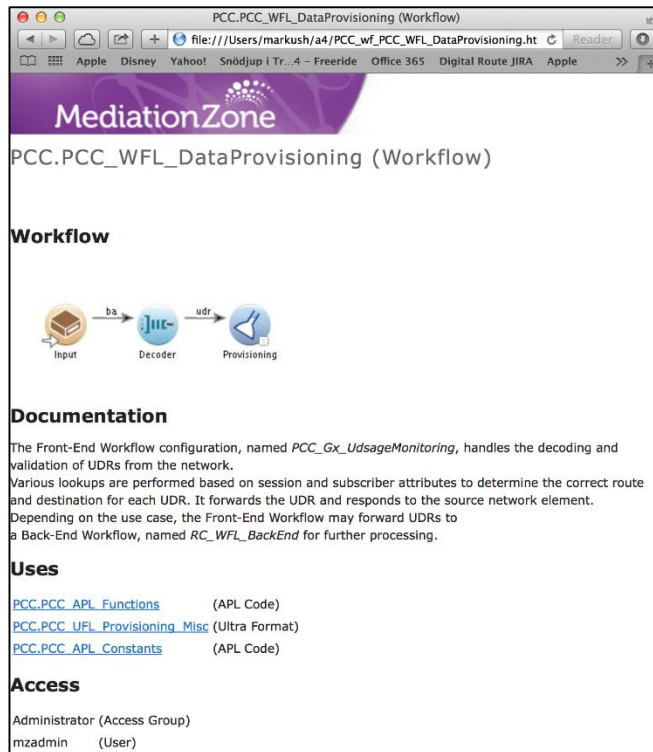
50 workflows,
14 Ultras
15 workflow groups,

But no documentation






- Demo



PCC.PCC_WFL_DataProvisioning (Workflow)

Workflow



```
graph LR; Input[Input] -- ba --> Decoder[Decoder]; Decoder -- udr --> Provisioning[Provisioning]
```

Documentation

The Front-End Workflow configuration, named *PCC_Gx_UdsageMonitoring*, handles the decoding and validation of UDRs from the network. Various lookups are performed based on session and subscriber attributes to determine the correct route and destination for each UDR. It forwards the UDR and responds to the source network element. Depending on the use case, the Front-End Workflow may forward UDRs to a Back-End Workflow, named *RC_WFL_BackEnd* for further processing.

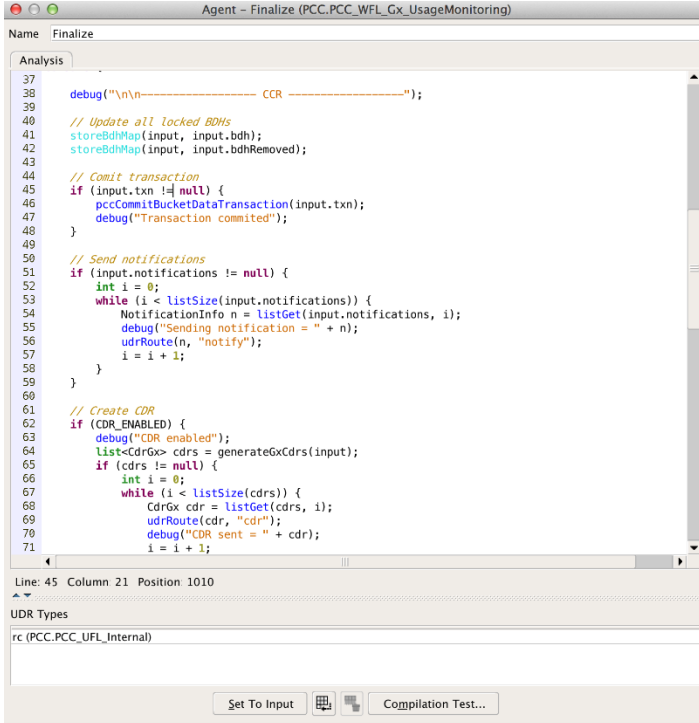
Uses

- [PCC.PCC_APL_Functions](#) (APL Code)
- [PCC.PCC_UFL_Provisioning_Misc](#) (Ultra Format)
- [PCC.PCC_APL_Constants](#) (APL Code)

Access

- Administrator (Access Group)
- mzadmin (User)

- Demo





```
Agent - Finalize (PCC.PCC_WFL_Gx_UsageMonitoring)
Name Finalize
Analysis
37
38
39
40 // Update all locked BDHs
41 storeBdhMap(input, input.bdh);
42 storeBdhMap(input, input.bdhRemoved);
43
44 // Commit transaction
45 if (input.txn != null) {
46     pccCommitBucketDataTransaction(input.txn);
47     debug("Transaction committed");
48 }
49
50 // Send notifications
51 if (input.notifications != null) {
52     int i = 0;
53     while (i < listSize(input.notifications)) {
54         NotificationInfo n = listGet(input.notifications, i);
55         debug("Sending notification = " + n);
56         udrRoute(n, "notify");
57         i = i + 1;
58     }
59 }
60
61 // Create CDR
62 if (CDR_ENABLED) {
63     debug("CDR enabled");
64     list<CDrGx> cdrs = generateGxCdrs(input);
65     if (cdrs != null) {
66         int i = 0;
67         while (i < listSize(cdrs)) {
68             CDrGx cdr = listGet(cdrs, i);
69             udrRoute(cdr, "cdr");
70             debug("CDR sent = " + cdr);
71             i = i + 1;
72         }
73     }
74 }
75 }
76 }
77 }
78 }
79 }
80 }
81 }
82 }
83 }
84 }
85 }
86 }
87 }
88 }
89 }
90 }
91 }
92 }
93 }
94 }
95 }
96 }
97 }
98 }
99 }
100 }
```

Line: 45 Column: 21 Position: 1010

UDR Types

rc (PCC.PCC_UFL_Internal)

Set To Input   Compilation Test...

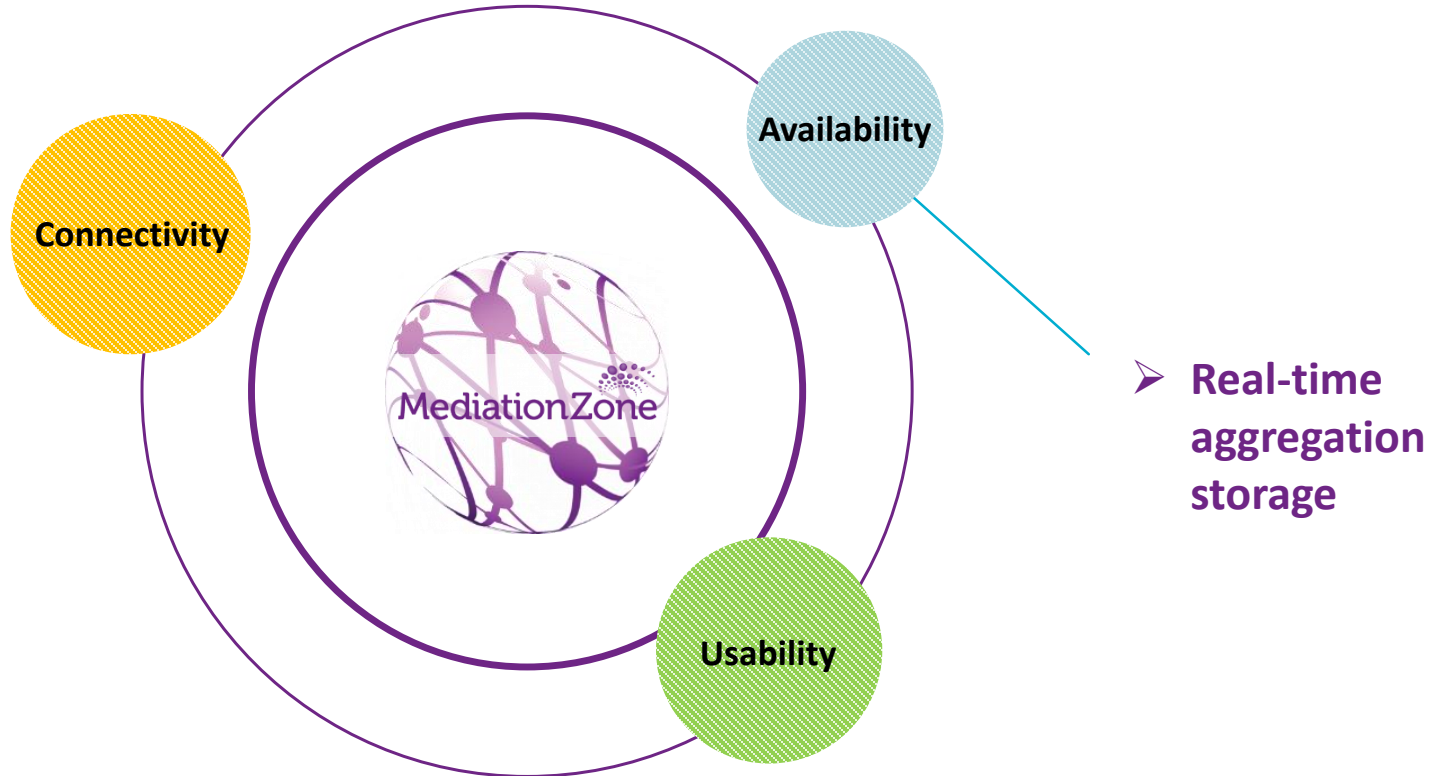
Classic for loop:

```
list<int> numbers = listCreate(int);  
listAdd(numbers, 77);  
listAdd(numbers, 24);  
  
for( int i; i<listSize(numbers); i++)  
{  
    debug(listGet(numbers, i));  
}
```

For each loop:

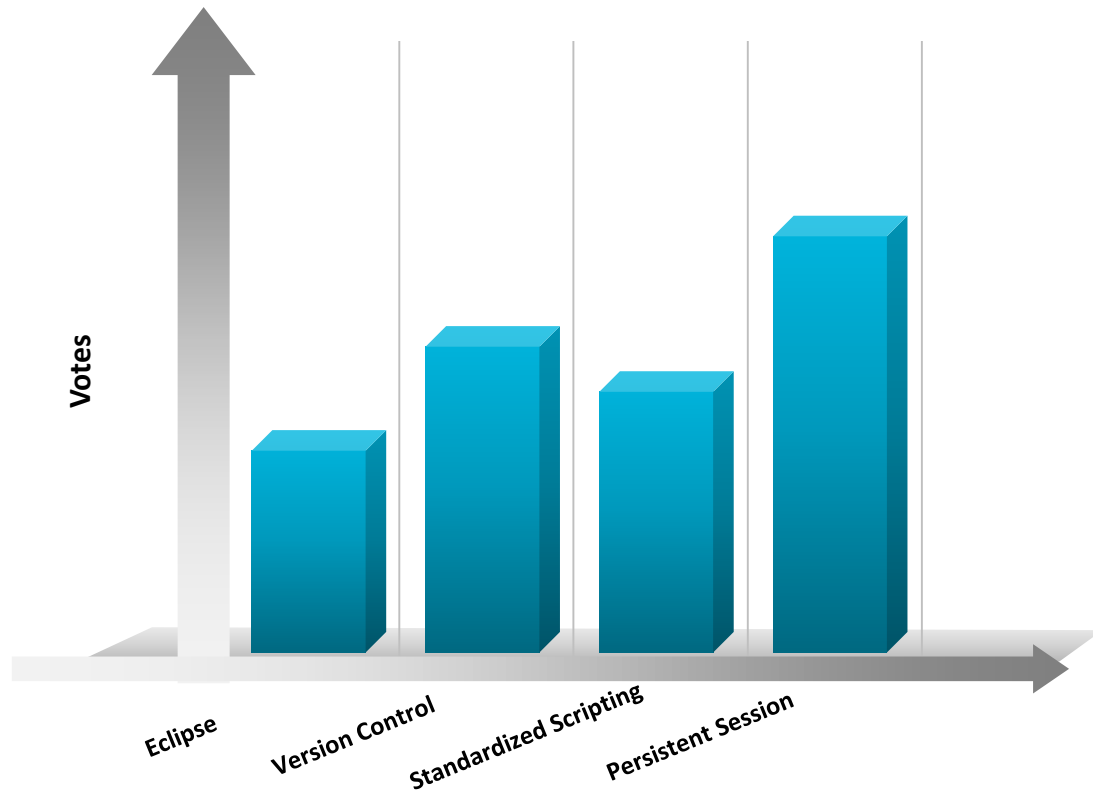
```
list<int> numbers = listCreate(int);  
listAdd(numbers, 77);  
listAdd(numbers, 24);  
  
for (int n : numbers)  
{  
    debug(n);  
}
```

Development themes



From the last UGM..

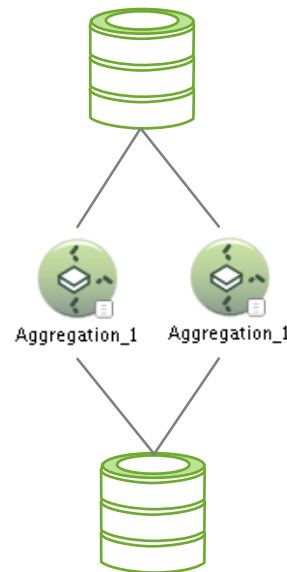
Functionality that you voted to see it as part of MediationZone!



- Current real-time aggregation has excellent performance
 - Using heap memory aggregation cache
- Scales by adding new workflow instances or threads
- But the current real-time aggregation agent has some limitations
 - Not possible to access same aggregation cache from different workflows

Real-time aggregation storage

- Real-time and scalable storage support for our real-time aggregation agent
- Guaranteed service continuation
- Geographical replicated data
- Simplified Horizontal scale out
- Allows concurrent access from multiple real-time workflows
- Scales with additional workflows and storage nodes
- Uses clustered in-memory database technology
 - Couchbase NoSQL database



Document NoSQL database

- Value is read and written based in a key
- Documents can be any value
- JSON documents supports querying inside Couchbase
- Supports geo-redundancy

Architecture

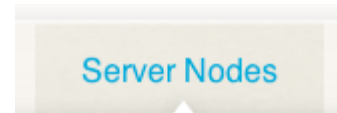
- Shared nothing architecture
- Horizontal scaling with additional nodes
- Asynchronous design





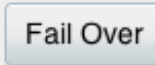

High availability

- Replication level (N) can be configured
- N+1 nodes have the data to support failover
- Supports geo-redundancy

Server Nodes

- Cluster consists of server nodes
- Data is spread across server nodes (the cluster) and replicated



Server Node Name	RAM Usage	Swap Usage	CPU Usage	Data/Disk Usage	Items (Active / Replica)	
 10.46.120.91	Up  29.5%	 0.252%	 0.9%	785MB / 1125MB	1.67 M / 1.67 M	 

IP of server

Active in cluster

Used memory

Stored on disk

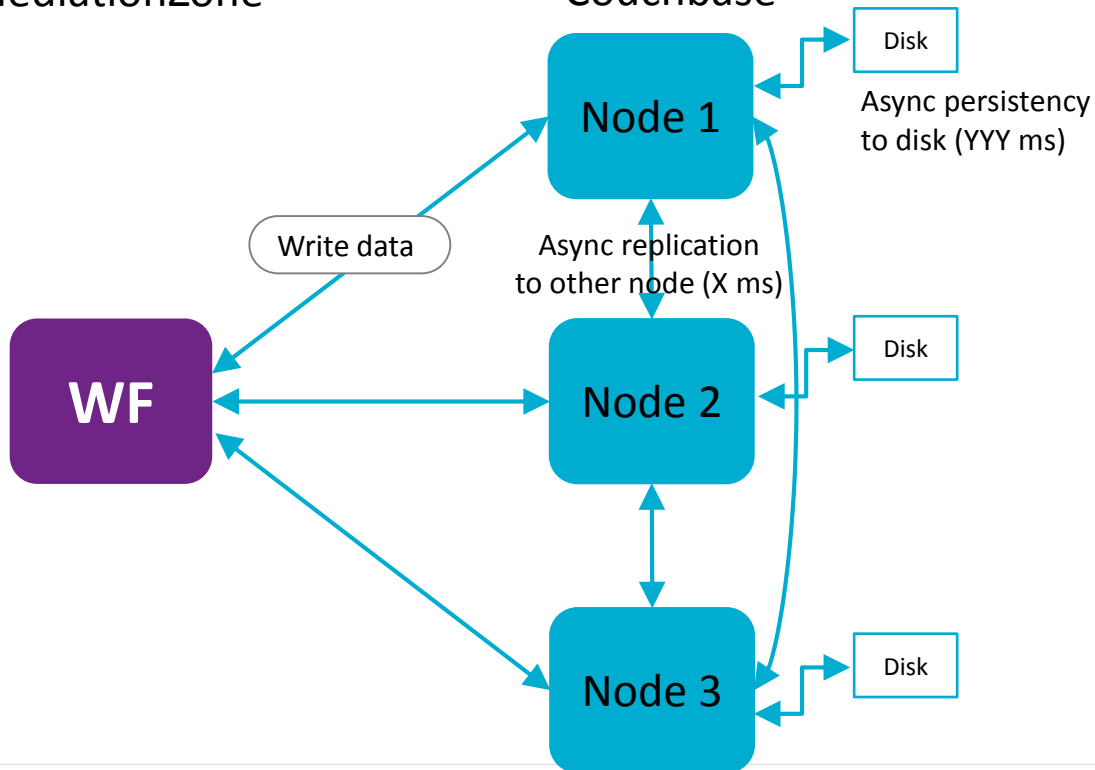
Replicated # documents

Owns # documents

Replication & Persistency

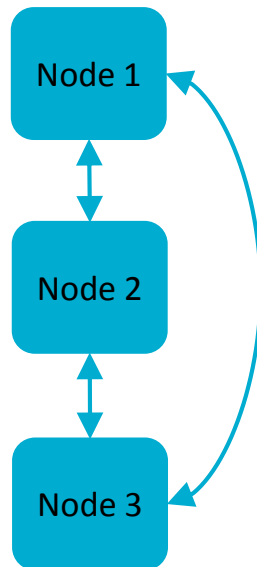
MediationZone

Couchbase

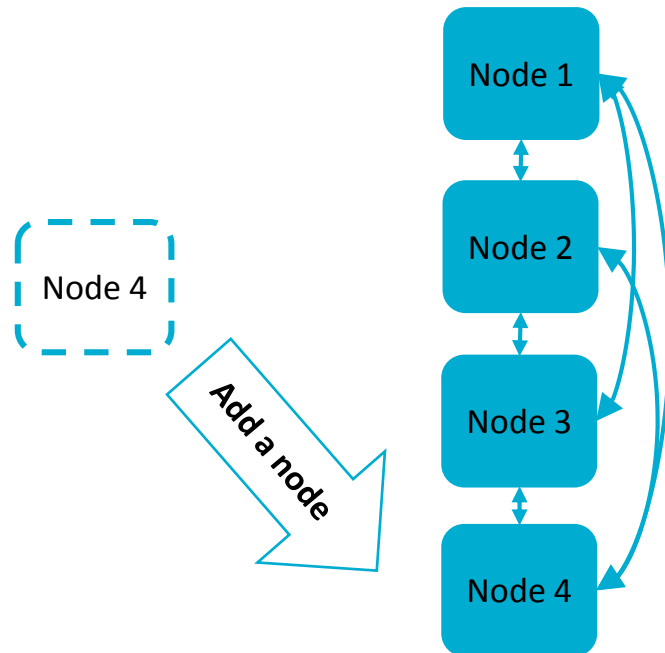


- Nodes can join the cluster at runtime
- Once the node joins the cluster, the cluster must be rebalanced
- Distribute data evenly among nodes
- Scales linearly

3 node cluster



4 node cluster





www.digitalroute.com