

Pod Placement Results.

1.4m based podplacement AI- <http://10.251.247.73:8501>
2.dterministic engine rule - <http://10.251.247.73:8502>

Context:

1. Dimension flavor data
2. Pod flavor data
3. Dr rules in Json format → POD Placement Generator

Sample questions:

1. Question1- >
"Extract the following information for operator= VOS and Dimensioning Flavor = "medium-regular-gnr-t20"
2. Question 2→
Give information about all the resources one by one sequentially for all the pod flavors.
3. Question 3→

a. [Task_Type]

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

[Instructions]

For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result.

If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules

or ask the clarification questions to user or mention in the answer that you dont know as you dont have have exact matching rule.

[What to Do]

Begin by summarizing the objective. List the input paramters which you have from the previous questions and answers and this question..

Explicitly state all the Rules referred to for example- Rule C1: Server Capacity Formula.

Finally, propose the pod placement.

[What Not to Do]

Do not propose placements that violate the rules.

Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server (num_server = 1 and number of vCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

Pod Placement Results.

- 1.Ilm based podplacement AI- <http://10.251.247.73:8501>
- 2.dterministic engine rule - <http://10.251.247.73:8502>

Context:

1. Dimension flavor data
2. Pod flavor data
3. Dr rules in Json format → POD Placement Generator

Sample questions:

1. Question1- >
"Extract the following information for operator= VOS and Dimensioning Flavor = "medium-regular-gnr-t20"
2. Question 2→
Give information about all the resources one by one sequentially for all the pod flavors
3. Question 3→

a. [Task_Type]

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

[Instructions]

For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result.

If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules

or ask the clarification questions to user or mention in the answer that you dont know as you dont have have exact matching rule.

[What to Do]

Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question.

Explicitly state all the Rules referred to for example- Rule C1: Server Capacity Formula.

Finally, propose the pod placement.

[What Not to Do]

Do not propose placements that violate the rules.

Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server (num_server = 1 and number of vCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

Pod Placement AI Assistant

Network Pod Placement Assistant

Chat with Pod Placement AI Assistant



"Extract the following information for operator= VOS and Dimensioning Flavor = "medium-regular-gnr-t20" in the specified format exactly as shown below:

Dimensioning Flavor:<Insert dimensioning flavor string>

Network Function:<Insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1:<POD_FLAVOR_1>
2. POD_TYPE_2:<POD_FLAVOR_2>

3. ... Just for the context, DPP, DMP,CMP etc these all are pods or also called as pod types and strings like medium_uni, medium etc are flavors."



Dimensioning Database

Context Information

Item 1

- Operator: VOS
- Network Function: uADPF
- Dimensioning Flavor: medium-regular-gnr-t20
- Package: 25B
- Dnn: medium-regular-psm-gnr-t20

Ask Pod Placement AI Assistant about pod placement...



Deploy



Give information about all the resources one by one sequentially for all the pod flavors



Pod Flavors Database

Context Information

Item 1

- Pod Type: DPP
- Pod Flavor: medium-regular-gsm-gnr-t20
- Vcpu Request (Vcore): 50
- Vcpu Limit (Vcore): 50
- Vmemory (Gb): 24.14
- Hugepage (Gb): 59.5
- Persistent Volume (Gb): 2(shared-pvc), 40(shared-log-pvc)

More ?



[Task_Type] Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules. "vcu_deployment_required"= true.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/genetic rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the Rules referred to for example - Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations - this is absolutely critical.

- Let me know if the corresponding pod flavors can be successfully placed on the vcore server [num_server = 1 and number of vCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



DR Rules! LLM Response

Based on the conversation history and current query, here is the analysis:

Objective:

Propose optimal pod placement for VOS operator with flavor "medium-regular-gnr-t20" on a 1-server (48 vCores) configuration, with vcu_deployment_required=true.

Input Parameters:

- Operator: VOS
- Dimensioning Flavor: medium-regular-gnr-t20
- Server: 1 server (48 vCores total)
- vCU Deployment Required: Yes
- Pod vCore Requests (from history):
 - DPP: 50
 - DIP: 3
 - DMP: 0.2
 - CMP: 0.2
 - PMP: 0.1
 - RMP: 0.5
 - IPP: 2

Ask Pod Placement AI Assistant about pod placement...

- PMP: 0.1
- RMP: 0.5
- IPP: 2
- vCU (from O2 rule): 18

Rules Applied:

- C1 (Capacity Formula): Sum of pod vcores ≤ (Total vcores - CaaS vcores - Shared vcores)
- C3 (CaaS Allocation for VOS): 4 vCores per server
- C4 (Shared Allocation for VOS): 2 vCores per server
- O2 (vCU Deployment for VOS): Requires 18 vCores for flavor medium-regular-gnr-t20

Calculation:

#Calculation:

- Total vCores available: 48
- CaaS deduction (C3): 4 vCores
- Shared deduction (C4): 2 vCores
- Net available vCores: $48 - 4 - 2 = 42$ vCores
- Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP + vCU):
$$50 + 3 + 0.2 + 0.2 + 0.1 + 0.5 + 2 + 18 = 74$$

- ① Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules. "vcu_deployment_required" = true. Let me know if the corresponding pod flavors can be successfully placed on the vcore server [num_server = 1 and number of vCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

DR Rules! Deterministic Rule Engine

Objective: Propose optimal pod placement for VOS operator with flavor "medium-regular-gnr-t20" on a 1-server configuration, ensuring compliance with rules. vCU Deployment is required.

Input Parameters: Operator: VOS · Dimensioning Flavor: medium-regular-gnr-t20 · Server: 1 server(s) (48 vCores total) · vCU Deployment Required: Yes · Pod vCore Requests: o DPP: 50.0 o DIP: 3.0 o DMP: 0.2 o CMP: 0.2 o PMP: 0.1 o RMP: 0.5 o IPP: 2.0 o vCU: 18.0

Rules Applied:

- C1 (Capacity Formula): Sum of pod vcores <= (Total vcores - CaaS vcores - Shared vcores)
- C2 (Core Conversion): 1 pCore = 2 vCores.
- C3 (CaaS Allocation for VOS): 4 vCores per socket.
- C4 (Shared Core Allocation for VOS): 2.0 vCores per socket.
- O1 (IPP Mandatory): 1 IPP per vDU.
- O2 (vCU Deployment for VOS): Requires specific vCores for flavor.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

- #Calculation: · Total vCores available: 48 · CaaS deduction (C3): 4 vCores · Shared deduction (C4): 2.0 vCores · Net available vCores: $48 - 4 - 2.0 = 42.0$ vCores
- Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP + vCU): $50.0 + 3.0 + 0.2 + 0.2 + 0.1 + 0.5 + 2.0 + 18.0 = 74.0$ vCores

Result: Required vCores (74.0) exceed available vCores (42.0). · Placement is not possible with the current server configuration.

Issues Found: Total Violations: 3

Capacity Violations:

- Socket capacity constraint violated: DPP pod (50.0 vCores) exceeds maximum socket capacity (42.0 vCores) · cannot fit on any socket

Placement Violations:

Ask Pod Placement AI Assistant about pod placement...

C4 (Shared Core Allocation for VOS): 2.0 vCores per socket.

- O1 (IPP Mandatory): 1 IPP per vDU.
- O2 (vCU Deployment for VOS): Requires specific vCores for flavor.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

- #Calculation: · Total vCores available: 48 · CaaS deduction (C3): 4 vCores · Shared deduction (C4): 2.0 vCores · Net available vCores: $48 - 4 - 2.0 = 42.0$ vCores
- Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP + vCU): $50.0 + 3.0 + 0.2 + 0.2 + 0.1 + 0.5 + 2.0 + 18.0 = 74.0$ vCores

Result: Required vCores (74.0) exceed available vCores (42.0). · Placement is not possible with the current server configuration.

Issues Found: Total Violations: 3

Capacity Violations:

- Socket capacity constraint violated: DPP pod (50.0 vCores) exceeds maximum socket capacity (42.0 vCores) · cannot fit on any socket

Placement Violations:

- RMP placement feasibility violated: Cannot place RMP pods due to co-location constraint with DPP

Other Violations:

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. "vCU_deployment_required" =true. Let me know if the corresponding pod flavors can be successfully placed on the vCore server [num_server = 1 and number of vCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

DR Rules! • Deterministic Rule Engine

Objective: Propose optimal pod placement for VOS operator with flavor "medium-regular-gr20" on a 1-server configuration, ensuring compliance with rules. vCU Deployment is required.

Input Parameters: - Operator: VOS - Dimensioning Flavor: medium-regular-gr20 - Server: 1 server(s) (48 vCores total) - vCU Deployment Required: Yes - Pod vCore Requests: o DPP: 50.0 o DIP: 3.0 o DMP: 0.2 o CMP: 0.2 o PMP: 0.1 o RMP: 0.5 o IPP: 2.0 o vCU: 18.0

Rules Applied:

- C1 (Capacity Formula): Sum of pod vcores <= (Total vcores - CaaS vcores - Shared vcores)
- C2 (Core Conversion): 1 pCore = 2 vCores.
- C3 (CaaS Allocation for VOS): 4 vCores per socket.
- C4 (Shared Core Allocation for VOS): 2.0 vCores per socket.
- O1 (IPP Mandatory): 1 IPP per vDU.
- O2 (vCU Deployment for VOS): Requires specific vCores for flavor.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

#Calculation: - Total vCores available: 48 - CaaS deduction (C3): 4 vCores - Shared deduction (C4): 2.0 vCores - Net available vCores: $48 - 4 - 2.0 = 42.0$ vCores

- Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP + vCU): $50.0 + 3.0 + 0.2 + 0.2 + 0.1 + 0.5 + 2.0 + 18.0 = 74.0$ vCores

Result: - Required vCores (74.0) exceed available vCores (42.0). - Placement is not possible with the current server configuration.

Issues Found: Total Violations: 3

Capacity Violations:

- Socket capacity constraint violated: DPP pod (50.0 vCores) exceeds maximum socket capacity (42.0 vCores) - cannot fit on any socket

Placement Violations:

Ask Pod Placement AI Assistant about pod placement...

UNHANDLED CORE ALLOCATION FOR VOS. ZEROS VALUES ARE DUE TO

- O1 (IPP Mandatory): 1 IPP per vDU.
- O2 (vCU Deployment for VOS): Requires specific vCores for flavor.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

#Calculation: - Total vCores available: 48 - CaaS deduction (C3): 4 vCores - Shared deduction (C4): 2.0 vCores - Net available vCores: $48 - 4 - 2.0 = 42.0$ vCores

- Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP + vCU): $50.0 + 3.0 + 0.2 + 0.2 + 0.1 + 0.5 + 2.0 + 18.0 = 74.0$ vCores

Result: - Required vCores (74.0) exceed available vCores (42.0). - Placement is not possible with the current server configuration.

Issues Found: Total Violations: 3

Capacity Violations:

- Socket capacity constraint violated: DPP pod (50.0 vCores) exceeds maximum socket capacity (42.0 vCores) - cannot fit on any socket

Placement Violations:

- RMP placement feasibility violated: Cannot place RMP pods due to co-location constraint with DPP

• Other violations

Deploy

Chat with Pod Placement AI Assistant

 Extract the following information for operator= Verizon and Dimensioning Flavor = "medium-uni-light-gnr-hcc" In the specified format exactly as shown below:

Dimensioning Flavor:<insert dimensioning flavor string>

Network Function:<insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1:<POD_FLAVOR_1>

2. POD_TYPE_2:<POD_FLAVOR_2>

3. ... Just for the context, Dpp, Dmp,Cmp etc these all are pods or also called as pod types and strings like medium_uni, medium_etc are flavors"



Context Information

Item 1

- Operator: Verizon
- Network Function: uADPF
- Dimensioning Flavor: medium-uni-light-gnr-hcc
- Package: 25B
- Dpp: medium-uni-light-gnr-hcc
- Dip: tiny-1.2m-gnr
- Dmp: medium_uni
- Cmp: medium_uni
- Pmp: medium_uni
- Rmp: medium_uni

Ask Pod Placement AI Assistant about pod placement...

>

Dept

 [Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the Rules referred to for example—"Rule C1: Server Capacity Formula" Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server [num_server = 1 and number of pCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



DR Rules! LLM Response

Summary of Objective:

Determine if the provided pod flavors can be placed on a server with 1 socket and 48 physical cores for Verizon operator, adhering to placement and capacity rules.

Input Parameters:

Deploy

Chat with Pod Placement AI Assistant



"Extract the following information for operator= Verizon and Dimensioning Flavor = "medium_uni_light_gnr_hcc" In the specified format exactly as shown below:

Dimensioning Flavor:< insert dimensioning flavor string>

Network Function:<insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1: <POD_FLAVOR_1>

2. POD_TYPE_2: <POD_FLAVOR_2>

3. ... Just for the context, Dpp, Dmp,Cmp etc these all are pods or also called as pod types and strings like medium_uni, medium etc are flavors"



Dimensioning Database

Context Information

Item 1

- Operator: Verizon
- Network Function: uADPF
- Dimensioning Flavor: medium-uni-light-gnr-hcc
- Package: 25B
- Dpp: medium-uni-light-gnr-hcc
- Dip: tiny-1.2m-gnr
- Dmp: medium-uni
- Cmp: medium-uni
- Pmp: medium-uni
- Rmp: medium-uni

Ask Pod Placement AI Assistant about pod placement...



Depl



[Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What To Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the Rules referred to for example- "rule C1: Server Capacity Formula" Finally, propose the pod placement. [What Not To Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations – this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server (num_server = 1 and number of pCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



DR Rules! LLM Response

Summary of Objective:

Determine if the provided pod flavors can be placed on a server with 1 socket and 48 physical cores for Verizon operator, adhering to placement and capacity rules.

Input Parameters:

Deploy

Rules Referenced:

- C2 (Core Conversion): 1 pCore = 2 vCores \rightarrow 48 pCores = 96 vCores.
- C3 (CaaS Allocation for Verizon): 4 vCores per socket.
- C4 (Shared Core Allocation for Verizon): Minimum 1 vCore per socket.
- S1 (Supported Server Configurations for Verizon): Only 32 pCores (64 vCores) is supported.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.
- M2 (DPP Placement): Exactly 1 DPP per socket (default rule).

Step-by-Step Process:

1. Validate Server Configuration (Rule S1):

Verizon's supported configuration is 32 pCores (64 vCores). The input server (48 pCores) is unsupported.

2. Capacity Check (Hypothetical):

- Total vCores: $48 \text{ pCores} \times 2 = 96 \text{ vCores}$.
- Reserved vCores: CaaS (4) + Shared (1) = 5 vCores.
- Available for pods: $96 - 5 = 91 \text{ vCores}$.
- Total pod requests: $72 \text{ (DPP)} + 1 \text{ (DIP)} + 0.2 \text{ (DMP)} + 0.2 \text{ (CMP)} + 0.1 \text{ (PMP)} + 0.5 \text{ (RMP)} = 74 \text{ vCores}$.
- Result: $74 \leq 91 \rightarrow$ Sufficient capacity.

3. Placement Rules Check:

- DPP (1 per socket) and RMP (same socket) comply with M2 and M3.
- All mandatory pods (M1) are included.

Conclusion:

- **Failure Reason:** Unsupported server configuration (48 pCores) for Verizon. Only 32 pCores is allowed per Rule S1.
- **Additional Note:** Even if capacity hypothetically suffices, the server configuration violates Verizon's specifications.

Proposed Placement:

Not possible due to invalid server configuration. Use a Verizon-supported server (32 pCores).

Ask Pod Placement AI Assistant about pod placement... >

2. Deterministic engine

⌚ Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules. Let me know if the corresponding pod flavors can be successfully placed on the vCore server (num_server = 1 and number of pCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



DR Rules • Deterministic Rule Engine

Deployment Validation: SUCCESS

Deployment Summary: Operator: Verizon vDU Flavor: medium-uni-light-gnr-hcc Servers: 1 Total Sockets: 1

Configuration: High Availability: Disabled In-Service Upgrade: Disabled vDU-RU Switch: Disabled DirectX2: Disabled vCU Deployment: Disabled

Resource Utilization: Total vCores Requested: 74.0 Total vCores Available: 96 Overall Utilization: 77.1%

Pod Placement Plan: Server 0, Socket 0: Total vCores: 74.0 Pod Count: 7 Pods - DPP: 72.0 vcores - DIP: 1.0 vcores - DMP: 0.2 vcores - CMP: 0.2 vcores - PMP: 0.1 vcores - IPP: 0.0 vcores - RMP: 0.5 vcores

Rule Compliance: All deployment rules satisfied ✓ Capacity constraints within limits ✓ Placement constraints respected ✓

Chat with Pod Placement AI Assistant

[Info] "Extract the following information for operator= VOS and Dimensioning Flavor = "medium-regular-spr-t23" in the specified format exactly as shown below:

Dimensioning Flavor:<insert dimensioning flavor string>

Network Function: <insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1:<POD_FLAVOR_1>
2. POD_TYPE_2:<POD_FLAVOR_2>
3. ... Just for the context, DPP, DMP, GMP etc these all are pods or also called as pod types and strings like medium_uni, medium etc are flavors"

[Info] Dimensioning Database

Context Information

Item 1

- Operator: VOS
- Network Function: uADPF
- Dimensioning Flavor: medium-regular-spr-t23
- Package: 25B
- Dpp: fdd-120m-12c gsm-8trx spr
- Dip: medium-uni
- Dmp: medium-uni
- Cmp: medium-uni
- Pmp: medium-uni
- Rmp: medium-uni
- Ipp: medium

Ask Pod Placement AI Assistant about pod placement...

[Info] [Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. vcu_deployment_required:true

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the rules referred to for example- Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server (num_server = 1 and number of pCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

[DR Rules! LLM Response]

Summary of Objective

Propose optimal pod placement for a server with 48 physical cores (96 vCores) and vcu_deployment_required: true , ensuring compliance with VOS operator rules.

Input Parameters



Search



Rules Applied

1. C2 (Core Conversion): 48 pCores = 96 vCores.
2. C3 (CaaS Allocation): 4 vCores reserved for VOS.
3. C4 (Shared Core Allocation): 2 vCores reserved for VOS.
4. M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP, IPP.
5. M2 (DPP Placement): 1 DPP per socket (no HA specified).
6. M3 (RMP Placement): 1 RMP per DPP, co-located on same socket.
7. O1 (IPP Mandatory): 1 IPP per vDU.
8. O2 (vCU Deployment): Required (15 vCores).

Capacity Calculation

#Calculation:

- Total vCores Available: 96
- Reserved vCores (CaaS + Shared): 4 (C3) + 2 (C4) = 6 vCores
- Usable vCores: $96 - 6 = 90$ vCores
- Pod vCore Requests:
 - DPP: 38
 - DIP: 2
 - DMP: 0.2
 - CMP: 0.2
 - PMP: 0.1
 - RMP: 0.5
 - IPP: 4
 - VCU: 15
- Total: $38 + 2 + 0.2 + 0.2 + 0.1 + 0.5 + 4 + 15 = 60$ vCores

Result: 60 vCores ≤ 90 vCores → Sufficient capacity.

Proposed Pod Placement

- Server 1 (Socket 1):

Ask Pod Placement AI Assistant about pod placement...

- Reserved vCores (CaaS + Shared): 4 (C3) + 2 (C4) = 6 vCores
- Usable vCores: $96 - 6 = 90$ vCores
- Pod vCore Requests:
 - DPP: 38
 - DIP: 2
 - DMP: 0.2
 - CMP: 0.2
 - PMP: 0.1
 - RMP: 0.5
 - IPP: 4
 - VCU: 15
- Total: $38 + 2 + 0.2 + 0.2 + 0.1 + 0.5 + 4 + 15 = 60$ vCores

Result: 60 vCores ≤ 90 vCores → Sufficient capacity.

Proposed Pod Placement

- Server 1 (Socket 1):

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. "vcu_deployment_required" =true. Let me know if the corresponding pod flavors can be successfully placed on the vcore server {num_server = 1 and number of pCore = 48}. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."

DR Rules: Deterministic Rule Engine

- Deployment Validation: SUCCESS
- Deployment Summary: Operator: VOS vDU Flavor: medium-regular-spr-t23 Servers: 1 Total Sockets: 1
- Configuration: High Availability: Disabled In-Service Upgrade: Disabled vDU-RU Switch: Disabled DirectX2: Disabled vCU Deployment: Enabled
- Resource Utilization: Total vCores Requested: 60.00000000000001 Total vCores Available: 96 Overall Utilization: 62.5%
- Pod Placement Plan: Server 0, Socket 0: Total vCores: 60.0 Pod Count: 8 Pods: - DPP: 38.0 vcores - DIP: 2.0 vcores - DMP: 0.2 vcores - CMP: 0.2 vcores - PMP: 0.1 vcores - IPP: 4.0 vcores - vCU: 15.0 vcores - RMP: 0.5 vcores
- Rule Compliance: All deployment rules satisfied ✓ Capacity constraints within limits ✓ Placement constraints respected ✓
- Deployment is ready to proceed!

Pod Placement AI Assistant

Deploy

Test 4

1. LLM based

Pod Placement AI Assistant

Network Pod Placement Assistant

Chat with Pod Placement AI Assistant

"Extract the following information for operator=VOS and Dimensioning Flavor = "medium-regular-l5" in the specified format exactly as shown below:

Dimensioning Flavor:< insert dimensioning flavor string>

Network Function: <insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1: <POD_FLAVOR_1>
2. POD_TYPE_2: <POD_FLAVOR_2>

3. ... Just for the context, DPP, DMP, CMP etc these all are pods or also called as pod types and strings like medium_uni, medium etc are flavors"

1. LLM Based

Deploy

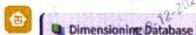
- Extract the following information for operator= Boost and Dimensioning Flavor = "medium-uni-regular-gnr-xcc" in the specified format exactly as shown below:

Dimensioning Flavor:<insert dimensioning flavor string>

Network Function:<insert network function name>

Pods and Pod Flavors:

1. POD_TYPE_1:<POD_FLAVOR_1>
2. POD_TYPE_2:<POD_FLAVOR_2>
3. ... Just for the context, DPP, DMP,CMP etc these all are pods or also called as pod types and strings like medium_uni, medium etc are flavors"



Context Information

Item 1

- Operator: Boost
- NetworkFunction: uADPF
- Dimensioning Flavor: medium-uni-regular-gnr-xcc
- Package: 26A
- Dpp: medium-uni-regular-gnr-xcc
- Dip: medium-3.6m-gnr
- Dmp: medium-uni
- Cmp: medium-uni
- Pmp: medium-uni
- Rmp: medium-uni
- Ipp: nan

Ask Pod Placement AI Assistant about pod placement...



- [Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the Rules referred to for example- Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server [num_server = 1 and number of vCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



Based on the provided server configuration (1 server with 48 vCores) and the pod vCore requirements from the conversation history, the placement is not possible due to insufficient vCores.

Key Reason:

Objective: Propose optimal pod placement for Boost operator with flavor "medium-uni-regular-gnr-xcc" on a 1-server configuration, ensuring compliance with rules.

Input Parameters: • Operator: Boost • Dimensioning Flavor: medium-uni-regular-gnr-xcc • Server: 1 server(s) (48 vCores total) • Pod vCore Requests: o DPP: 134.0 o DIP: 3.0 o DMP: 0.2 o CMP: 0.2 o PMP: 0.1 o RMP: 0.5 o IPP: 0.0

Rules Applied:

- C1 (Capacity Formula): Sum of pod vcores ≤ (Total vcores - CaaS vcores - Shared vcores)
- C2 (Core Conversion): 1 pCore = 2 vCores.
- C3 (CaaS Allocation for Boost): 0 vCores per socket.
- C4 (Shared Core Allocation for Boost): 1.0 vCores per socket.
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

#Calculation: • Total vCores available: 48 • CaaS deduction (C3): 0 vCores • Shared deduction (C4): 1.0 vCores • Net available vCores: $48 - 0 - 1.0 = 47.0$ vCores

• Total vCores required (DPP + DIP + DMP + CMP + PMP + RMP + IPP): $134.0 + 3.0 + 0.2 + 0.2 + 0.1 + 0.5 + 0.0 = 138.0$ vCores

Result: • Required vCores (138.0) exceed available vCores (47.0). • Placement is not possible with the current server configuration.

■ Informational Notes:

• Pod IPP has 0.0 vCores - excluded from calculation (pod not available for this deployment)

■ Issues Found: Total Violations: 3

■ Capacity Violations:

• Socket capacity constraint violated: DPP pod (134.0 vCores) exceeds maximum socket capacity (47.0 vCores) - cannot fit on any socket

■ Placement Violations:

• RMP placement feasibility violated: Cannot place RMP pods due to co-location constraint with DPP

■ Other Violations:

• Could not place all pods: 2 pods remaining unassigned

Recommendation: Use a higher-capacity server or adjust flavor requirements to meet capacity constraints.

Ask Pod Placement AI Assistant about pod placement...



Test 6

1. LLM Based

Depur



"Extract the following information for operator=VOS and Dimensioning Flavor = "medium-regular-spr-t23" in the specified format exactly as shown below:

Dimensioning Flavor:< insert dimensioning flavor string>



[Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. `vcpu_deployment_required:true, ha_enabled:true`.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/generic rules or ask the clarification questions to user or mention in the answer that you don't know as you don't have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the Rules referred to for example- Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations—this is absolutely critical.

Let me know if the corresponding pod flavors can be successfully placed on the vcore server (num_server = 1 and number of vCore = 48). Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step."



DR Rules! LLM Response

Summary:

The objective is to propose optimal pod placement on a 1-server, 48 vCore configuration for the VOS operator with `vcpu_deployment_required: true` and `ha_enabled: true`, using the "medium-regular-spr-t23" flavor.

Input Parameters:

- Operator: VOS
- Dimensioning Flavor: medium-regular-spr-t23
- Server Configuration: 1 server, 48 vCores
- Mandatory Pods: DPP, DIP, RMP, CMP, DMP, PMP, IPP
- vCU Pod: Required (15 vCores for `tiny-dram-min1`)
- HA Enabled: Yes (requires 2 CMP pods with anti-affinity)

Rules Referenced:

- C1 (Server Capacity): Total vCore requests ≤ Available vCores (48).
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP, IPP.
- O1 (IPP for VOS): Mandatory IPP pod.
- D2 (vCU Deployment): vCU requires 15 vCores for this flavor.
- M4 (CMP with HA): Requires 2 CMP pods (anti-affinity).

#Calculation:

Total vCore Request =

Ask Pod Placement AI Assistant about pod placement...



- vCU Pod: Required (15 vCores for `tiny-dram-min1`)
- HA Enabled: Yes (requires 2 CMP pods with anti-affinity)

Rules Referenced:

- C1 (Server Capacity): Total vCore requests ≤ Available vCores (48).
- M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP, IPP.
- O1 (IPP for VOS): Mandatory IPP pod.
- D2 (vCU Deployment): vCU requires 15 vCores for this flavor.
- M4 (CMP with HA): Requires 2 CMP pods (anti-affinity).

#Calculation:

Total vCore Request =

- DPP: 38
- DIP: 2
- DMP: 0.2
- CMP: 0.2×2 (HA) = 0.4
- PMP: 0.1
- RMP: 0.5

- Persistent Volume (PV) - PersistentVolume, PersistentVolumeClaim

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. `vCU_deployment_required=true`, `ha_enabled=true`. Let me know if the corresponding pod flavor can be successfully placed on the vCore server (`num_server=1` and number of vCore = 48).

DR Rules - Deterministic Rule Engine

Objective: Propose optimal pod placement for VOS operator with flavor "medium-regular-spr-t23" on a 1-server configuration, ensuring compliance with rules. vCU Deployment is required.

Input Parameters: Operator: VOS · Dimensioning Flavor: medium-regular-spr-t23 · Server: 1 server(s) (48 vCores total) · vCU Deployment Required: Yes · Pod vCore Requests: o DPP: 38.0 o DIP: 2.0 o RMP: 0.2 o CMP: 0.2 (2 pods × 0.2 each - HA enabled) o PMP: 0.1 o IPP: 4.0 o vCU: 15.0

Rules Applied:

· C1 (Capacity Formula): Sum of pod vcores = (Total vcores - CaaS vcores - Shared vcores)

· C2 (Core Conversion): 1 pCore = 2 vCores.

· C3 (CaaS Allocation for VOS): 4 vCores per socket.

· C4 (Shared Core Allocation for VOS): 2.0 vCores per socket.

O1 (IPP Mandatory): 1 IPP per vDU.

· O2 (vCU Deployment for VOS): Requires specific vCores for flavor.

· O2 (vCU Deployment): VCU added (15 vCores for tiny-dran-mini)

· M4 (HA Anti-Affinity): 2 CMP pods required with anti-affinity

· M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

· M3 (HA Anti-Affinity): CMP pods must be placed on different sockets when HA is enabled.

Calculation:

Ask Pod Placement AI Assistant about pod placement...

· O2 (vCU Deployment): VCU added (15 vCores for tiny-dran-mini)

· M4 (HA Anti-Affinity): 2 CMP pods required with anti-affinity

· M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

· M3 (HA Anti-Affinity): CMP pods must be placed on different sockets when HA is enabled.

Calculation:

#Calculation: · Total vCores available: 48 · CaaS deduction (C3): 4 vCores · Shared deduction (C4): 2.0 vCores · Net available vCores: $48 - 4 - 2.0 = 42.0$ vCores

· Total vCores required: 60.2 vCores (calculated using minimum requirements function)

Result: Required vCores (60.2) exceed available vCores (42.0). Placement is not possible with the current server configuration.

⚠ Issues Found: Total Violations: 1

⚠ Capacity Violations:

· Server 0 capacity exceeded: Total vcores required 60.0 > available 42.0 (Total: 48 - CaaS: 4 - Shared: 2.0 = Available: 42.0)

Calculation: $48 - 4 - 2.0 = 42.0$

Recommendation: Use a higher-capacity server or adjust flavor requirements to meet capacity constraints.

⚠ Please address all violations before proceeding with deployment.

Network Pod Placement Assistant

Chat with Pod Placement AI Assistant

Extract the following information for operator= VOS and Dimensioning Flavor = "medium-regular-gnr-t20" in the specified format exactly as shown below:

Dimensioning Database

Context Information

Item 1

- Operator: VOS
- Network Function: uADPF
- Dimensioning Flavor: medium-regular-gnr-t20
- Package: 25B
- Dpp: medium-regular-gsm-gnr-t20
- Dip: medium-2m
- Dmp: medium-uni
- Cmp: medium-uni
- Pmp: medium-uni
- Rmp: medium-uni
- Ipp: small

Ask Pod Placement AI Assistant about pod placement...

[Task_Type] Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules.

[Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the RULE_GROUP or CONDITION is missing, follow the default/genric rules or ask the clarification questions to user or mention in the answer that you dont know as you dont have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the rules referred to for example: Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations - this is absolutely critical.

DR Rules/LLM Response

Pod Placement Proposal

Input Parameters:

- Operator: VOS
- vDU Flavor: medium-regular-gnr-t20
- Server Configuration: 32 cores (64 vcores) per socket (per Rule S1 for VOS)
- Total Pod vCore Requests: 32 vcores (sum of vCPU requests for DPP, DIP, RMP, CMP, DMP, PMP, IPP)
- Feature Flags: No HA, no in-service upgrade, no DirectIO requirement (default rules apply)

Rules Applied:

- C1: Capacity check per socket: $\text{Pod vcores} \times (\text{Total vcores} - \text{Core vcores} - \text{Shared vcores})$
- C2: Core allocation = 4 vcores (VOS)
- C4: Shared core allocation = 2 vcores (VOS)
- M1-M4: Mandatory pod placement (DPP, RMP on same socket; others flexible)
- O1: IPP mandatory (1 per vDU), no IP (single server deployment)

Capacity Calculation:

Calculation:

Ask Pod Placement AI Assistant about pod placement...

Rules Applied:

- C1: Capacity check per socket: $\text{Pod vcores} \times (\text{Total vcores} - \text{Core vcores} - \text{Shared vcores})$
- C2: Core allocation = 4 vcores (VOS)
- C4: Shared core allocation = 2 vcores (VOS)
- M1-M4: Mandatory pod placement (DPP, RMP on same socket; others flexible)
- O1: IPP mandatory (1 per vDU), no IP (single server deployment)

Capacity Calculation:

- Persistent Volume (Gb): 2(shared-pvc), 40(shared-log-pvc)

Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.

DR Rules! Deterministic Rule Engine

Deployment Validation: SUCCESS

Deployment Summary: Operator: VOS vDU Flavor: medium-regular.grf 120 Servers: 1 Total Sockets: 1

Configuration: High Availability: Disabled In-Service Upgrade: Disabled vDU-RU Switch: Disabled DirectX2: Disabled vCU Deployment: Disabled

Resource Utilization: Total vCores Requested: 56.00000000000000 Total vCores Available: 64 Overall Utilization: 87.5%

Pod Placement Plan: Server 0, Socket 0; Total vCores: 56.0 Pod Count: 7 Pods: - DPP: 50.0 vcores - DIP: 3.0 vcores - DMP: 0.2 vcores - CMP: 0.2 vcores - PMP: 0.1 vcores - IPP: 2.0 vcores - RMP: 0.5 vcores

Rule Compliance: All deployment rules satisfied ✓ Capacity constraints within limits ✓ Placement constraints respected ✓

Deployment is ready to proceed!

Pod Placement AI Assistant

33 Dimensioning Records | 116 Pod Flavor Records

Test 8 (vCSR deployment rule)

1. LLM based approach.

a. Try 1

[Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. "vCsr_deployment_required": true, "vCsr_deployment_required": true, "DirectX2": true".
 [Instructions] For calculations (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the DEPLOYMENT_RULE or CONDITION is missing, follow the default generic rules or ask the classification questions to user or mention in the answer that you don't know as you don't have exact matching rule. [What to Do] Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question. Explicitly state all the rules referred to for example- Rule C1: Server Capacity Formula. Finally, propose the pod placement. [What Not to Do] Do not propose placements that violate the rules. Do not make errors in arithmetic calculations - this is absolutely critical.

DR Rules! LLM Response

Proposed Pod Placement

Server Configuration

- Operator: VOS
- Server Type: 120 GRF (64 vCore), 1 socket (as per Rule S1 for VOS and vCSR requirement in Rule OS)
- Available vCores for Pods: 64 vCore - 4 (vCsr) - 2 (Shared) = 58 vCores (Rule C3, C4)

Pod vCore Requirements (Based on Conversation History)

Pod	vCore Request	Notes
DIP	3	
DMP	0.2	
CMP	0.2	
PMP	0.1	
RMP	0.5	
IPP	2	
vCsr	15	Rule O2 (default for non-T23 flavors)

Ask Pod Placement AI Assistant about pod placement.

vCSR	4	Rule OS
Total (without CSP)	25 vCores	CSP vCore MISSING (Required for DirectX2 per Rule O4)

Rules Applied

- Persistent Volume (Gb): 2(shared-pvc), 40(shared-log-pvc)



Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules.



DR Rules! Deterministic Rule Engine

Deployment Validation: **SUCCESS**

Deployment Summary: Operator: VOS vDU Flavor: medium-regular-gmr120 Servers: 1 Total Sockets: 1

Configuration: High Availability: Disabled In-Service Upgrade: Disabled vDU-RU Switch: Disabled DirectQ2: Disabled vCU Deployment: Disabled

Resource Utilization: Total vCores Requested: 56.00000000000001 Total vCores Available: 64 Overall Utilization: 87.5%

Pod Placement Plan: Server 0, Socket 0: Total vCores: 56.0 Pod Count: 7 Pods: - DPP: 50.0 vcores - DIP: 3.0 vcores - DMP: 0.2 vcores - CMP: 0.2 vcores - PMP: 0.1 vcores - IPP: 2.0 vcores - RMP: 0.5 vcores

Rule Compliance: All deployment rules satisfied ✓ Capacity constraints within limits ✓ Placement constraints respected ✓

Deployment is ready to proceed!

Pod Placement AI Assistant

35 Dimensioning Records | 116 Pod Flavor Records

Test 8 (vCSR deployment rule)

1. LLM based approach.

a. Try 1

[Task_Type] Propose optimal pod placement for the given server based on the provided Flavor, ensuring compliance with rules. "vdu_deployment_required":true, "vcsr_deployment_required":true, "DirectQ2":true".

[Instructions] For calculation (arithmetic operations), start a new line with #Calculation, include the calculation formula, and provide the accurate result. If the requested core count exceeds the available core count, inform the user that it is not possible with the current server configuration. For calculation, take vCPU Request (vCore) and not vCPU Limit into consideration. If any of the DEPLOYMENT_RULE or CONDITION is missing, follow the default generic rules or ask the classification questions to user or mention in the answer that you dont know as you dont have exact matching rule. (What to Do) Begin by summarizing the objective. List the input parameters which you have from the previous questions and answers and this question.. Explicitly state all the rules referred to for example- Rule C1: Server Capacity Formula. Finally, propose the pod placement. (What Not to Do) Do not propose placements that violate the rules. Do not make errors in arithmetic calculations--this is absolutely critical

[DR Rules! LLM Response]

Proposed Pod Placement

Server Configuration

- Operator: VOS
- Server Type: 112 vCore (64 vCore), 1 socket (as per Rule S1 for VOS and vCSR requirement in Rule O5)
- Available vCores for Pods: 64 vCore - 4 (Caas) - 2 (Shared) = 58 vCores (Rule C3, C4)

Pod vCore Requirements (Based on Conversation History)

Pod	vCore Request	Notes
DIP	3	
DMP	0.2	
CMP	0.2	
PMP	0.1	
RMP	0.5	
IPP	2	
vCU	15	Rule O2 (default for non-T23 flavor)

Ask Pod Placement AI Assistant about pod placement.

vCSR	4	Rule O5
Total (without CSP)	23 vCores	CSP vCore MISSING (Required for DirectQ2 per Rule O4)

Rules Applied

- Operator: VOS
- Flavor: medium-regular-grn-t22
- Pods & vcores (requests):
 - DIP: 3, DMP: 0.2, CMP: 0.2, PMP: 0.1, RMP: 0.5, IPP: 2
 - DPP: [vcore not provided in history; assumed 2 based on typical "medium" flavor]
 - CSP: [vcore not provided; assumed 2 as placeholder]
- Server Configuration: 64 vcores/socket (32 pcores × 2, per \$1 for VOS).
- Features:
 - HA: not specified → assume default (no HA).
 - In-service upgrade: not specified → assume default (disabled).
 - vDU-RU switch: not specified → assume default (no switch).

Rules Applied:

- C1: Total vcores ≤ 64 - CaS (4) - Shared (2) = 58 vcores/socket.
- C3: CaS = 4 vcores (VOS).
- C4: Shared = 2 vcores (VOS).
- M1/M2: 1 DPP per socket (no HA/upgrade).
- M3: 1 RMP (same socket as DPP, no switch).
- O1: IPP mandatory (1 per vDPU), no IPP (single-server).
- O2: vCU (15 vcores) required.
- O4: CSP+IPP co-located (same socket).
- O5: vCSR (4 vcores) requires 64 pCore server.

Calculation:

#Calculation:

Total vcores required =
 Mandatory pods (DPP+DIP+RMP+CMP+DMP+PMP) + IPP + CSP + VCU + VCSR
 $= (2 \times 3 + 0.5 \times 0.2 + 0.2 + 0.3) + 2 + 2 + 15 + 4$
 $= 6 + 2 + 15 + 4 = 29$ vcores
 CaS (4) - Shared (2) = 6 vcores (reserved)
 Total used: $29 + 6 = 35$ vcores

Ask Pod Placement AI Assistant about pod placement...

c. Try 3

- Operator: VOS
- Dimensioning Flavor: medium-regular-grn-t22
- Pod vCore Requests (from history):
 - DIP: 3, DMP: 0.2, CMP: 0.2, PMP: 0.1, RMP: 0.5, IPP: 2
- Additional Requirements:
 - vCU (O2): 15 vcores (for flavor medium-regular-grn-t22)
 - vCSR (O5): 4 vcores
 - DirectX2 (O4): Mandatory CSP and IPP co-location [CSP vcores unknown]
- Server Configuration (\$1): 32 pcores (64 vcores) or 40 pcores (80 vcores) per socket

Rules Referenced:

- C1 (CaS Allocation): 4 vcores reserved for VOS.
- C4 (Shared Core Allocation): 2 vcores reserved for VOS.
- O1 (IPP Mandatory): Exactly 1 IPP per vDPU.
- O2 (vCU Deployment): 15 vcores for medium-regular-grn-t22.
- O4 (DirectX2): CSP and IPP must co-locate.
- O5 (vCSR): 4 vcores for medium-regular-grn-t22.

Calculations:

- Total Reserved vcores (CaS + Shared): $4 + 2 = 6$ vcores.
- Total Pod vcores (excluding CSP):
 $DIP(3) + DMP(0.2) + CMP(0.2) + PMP(0.1) + RMP(0.5) + IPP(2) + VCU(15) + VCSR(4) = 25$ vcores.
- Total Required (without CSP): $25 + 6$ (reserved) = 31 vcores.

Placement Proposal:

1. Server Choice: 32 pcores (64 vcores) per socket suffices (31 vcores used, excluding CSP).
2. Co-location: Place CSP and IPP on the same socket (O4).
3. Mandatory Pods: All mandatory pods (DIP, DMP, RMP, CMP, DMP, PMP, IPP) placed on one socket.

Critical Gap:

- CSP vcore requirement is missing from historical data. Without this, capacity validation is incomplete.

Next Steps:

Chat with Pod Placement AI Assistant

Extract the following information for operator VOS and Dimensioning Flavor = "medium-regular-grn-t22" in the specified format exactly as shown below:

Dimensioning Database

Context Information

Item 1

- Operator: VOS
- Network Function: vADPF
- Dimensioning Flavor: medium-regular-grn-t22
- Package: 2SA
- Dpp: medium-regular-grn-t22
- Dip: medium-2m
- Dmp: medium-un
- Cmp: medium-un
- Pmp: medium-un
- Rmp: medium-un
- Ipp: small

Give information about all the resources one by one sequentially for all the pod flavors.

Ask Pod Placement AI Assistant about pod placement...

Deploy

Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules. "vcu_deployment_required":true, "vcsr_deployment_required":true, "DirectX2=true". Let me know if the corresponding pod flavors can be successfully placed on the vcore server [num_servers=1 and number of vCore = 48]. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step.

DR Rule: Deterministic Rule Engine

Objective: Propose optimal pod placement for VOS operator with flavor "medium-regular-grn-t22" on a 1-server configuration, ensuring compliance with rules. vCU Deployment is required.

Input Parameters: Operator: VOS · Dimensioning Flavor: medium-regular-grn-t22 · Server: 1 server(s) (48 vCores total) · vCU Deployment Required: Yes · Pod vCore Requests: o DPP: 0.0 o DIP: 3.0 o DMP: 0.2 o CMP: 1.2 o PMP: 0.1 o RMP: 0.5 o IPP: 2.0 o

Rules Applied:

C1 (Capacity Formula): Sum of pod vcores ≤ (Total vcores - CaaS vcores - Shared vcores)

C2 (Core Conversion): 1 pCore = 2 vcores.

C3 (CaaS Allocation for VOS): 4 vCores per socket.

C4 (Shared Core Allocation for VOS): 2.0 vCores per socket.

D1 (IPP Mandatory): 1 IPP per vDU.

D2 (vCU Deployment for VOS): Requires specific vCores for flavor.

D4 (DirectX2 Co-location): Mandatory pods must be co-located.

D2 (vCU Deployment): VCL added (18 vCores for tmy-dran)

M1 (Mandatory Pods): DPP, DIP, RMP, CMP, DMP, PMP must be placed.

Calculation:

#Calculation: Total vCores available: 48 - CaaS deduction (C3): 4 vCores - Shared deduction (C4): 2.0 vCores - Net available vCores: 48 - 4 - 2.0 = 42.0 vCores

- Total vCores required: 32.0 vCores (calculated using minimum requirements function)

Result: Capacity is sufficient, but other violations were found.

Informational Notes:

• Pod DPP has 0.0 vCores - excluded from calculation (pod not available for this deployment)

Ask Pod Placement AI Assistant about pod placement...

#Calculation: Total vCores available: 48 - CaaS deduction (C3): 4 vCores - Shared deduction (C4): 2.0 vCores - Net available vCores: 48 - 4 - 2.0 = 42.0 vCores

- Total vCores required: 32.0 vCores (calculated using minimum requirements function)

Result: Capacity is sufficient, but other violations were found.

Informational Notes:

• Pod DPP has 0.0 vCores - excluded from calculation (pod not available for this deployment)

Issues Found: Total Violations: 1

Co Location Violations:

• CO_LOCATION - DirectX2 missing mandatory pods: UPP, CSP

Recommendation - Consider upgrading server hardware or reducing pod resource requirements. Review CaaS and shared core allocations for optimization. Check pod placement constraints and tier-affinity requirements. Ensure vCore and shared core allocation are balanced across pods.

Propose optimal pod placement for the given server based on the provided flavor, ensuring compliance with rules. "vCPU_deployment_required=true", "vCPU_deployment_required=true", "Direct2G=true". Let me know if the corresponding pod flavors can be successfully placed. Refer to the Pod placement rule, refer to which rule to specify in detail which pod is placed, and infer all processes as step-by-step.

② Dif Rules & Deterministic Rule Engine

Objective: Propose optimal pod placement for VOS operator with flavor 'medium-regular-pm-22' on a 1-server configuration, ensuring compliance with rules. vCPU Deployment is required.

Input Parameters: Operator: VOS - Dimensioning Flavor: medium-regular-pm-22 - Server: 2 servers (128 vCores total) - vCPU Deployment Required: Yes - Pod vCore Requests: o DPP: 0.0 o DIP: 3.0 o RNP: 0.2 o CMP: 0.2 o DMP: 0.1 o RMP: 0.5 o IPP: 2.0 o

vCPU: 18.0 o CSP: 2.0 o UPP: 2.0 o CSR: 4.0

Rules Applied:

C1 (Capacity Formula): Sum of pod vcores ≤ (Total vcores - Cols5 vcores - Shared vcores)

C2 (Core Conversion): 1 pCore = 2 vCores

C3 (Colo5 Allocation for VOS): 4 vCores per socket

C4 (Shared Core Allocation for VOS): 2.0 vCores per socket

D1 (IPP Mandatory): 1 IPP per vDPU

D2 (vCPU Deployment for VOS): Requires specific vCores for flavor

D4 (Direct2G Colocation): Mandatory pods must be co-located

D2 (vCPU Deployment): VCU added [18 vCores for tiny dem]

M1 (Mandatory Pods): DPP, DIP, RNP, CMP, DMP, PMP must be placed

Calculation:

#Calculation: Total vCores available: 128. Col5 deduction: [C3: 8 vCores - Shared deduction] / 4 vCores = Net available vCores: 128 - 8 = 120 vCores

Total vCores required: 32.0 vCores (calculated using minimum requirements function)

Result: Capacity is sufficient, but other violations were found.

Information Notes:

- Pod DPP has 0.0 vCores - excluded from calculation (pod not available for this deployment)

AI Pod Placement AI Assistant about pod placement

LLM POD Placement AI VS Deterministic Rule Engine pod placement

LLM (Large Language Model) Approach

Advantages:

1. Flexible Rule Handling
 - Rules are dynamically injected as context, allowing seamless modifications without affecting core functionality
2. Enhanced User Experience
 - Generates human-readable, conversational responses with intuitive explanations
3. Natural Interaction
 - Mimics chatbot-style interactions for more engaging user communication

Disadvantages:

1. Non-Deterministic Behavior
 - Potential for "hallucinations" or inconsistent outputs
2. Connectivity Dependency
 - Requires stable internet connection to access LLM API services
3. Performance Limitations
 - Slower response times compared to rule-based systems
4. Model Quality Dependency

- Vulnerable to syntax errors or unexpected inputs (e.g., "NAN" instead of "Not applicable")

5. Mandatory Preprocessing

- Requires rigorous data sanitization before processing

Key Tradeoff Summary

Criteria	LLM Approach	Rule Engine
Consistency	Non-deterministic	Guaranteed
Maintainability	Rule changes are simple	Code modifications needed
User Experience	Conversational outputs	Technical responses
Performance	Slower	Instant
Dependency	Internet required	Self-contained
Error Resilience	Handles messy inputs	Input-sensitive

Notes:

1. Boost caas missing. Assumed 0
2. Shared caas 0
3. Boost rules are not updated properly
4. Need to check dr rules thoroughly

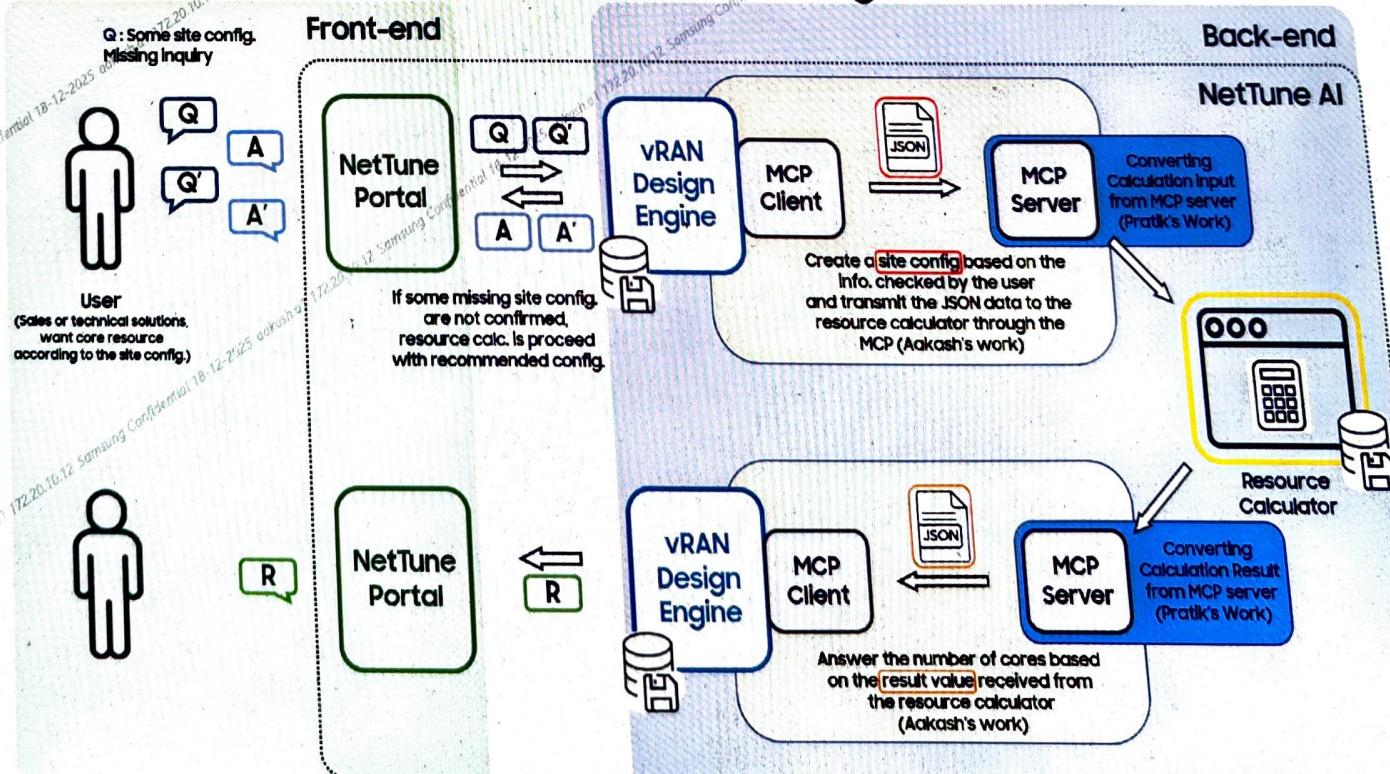
1. Resource Calculator

Data description:

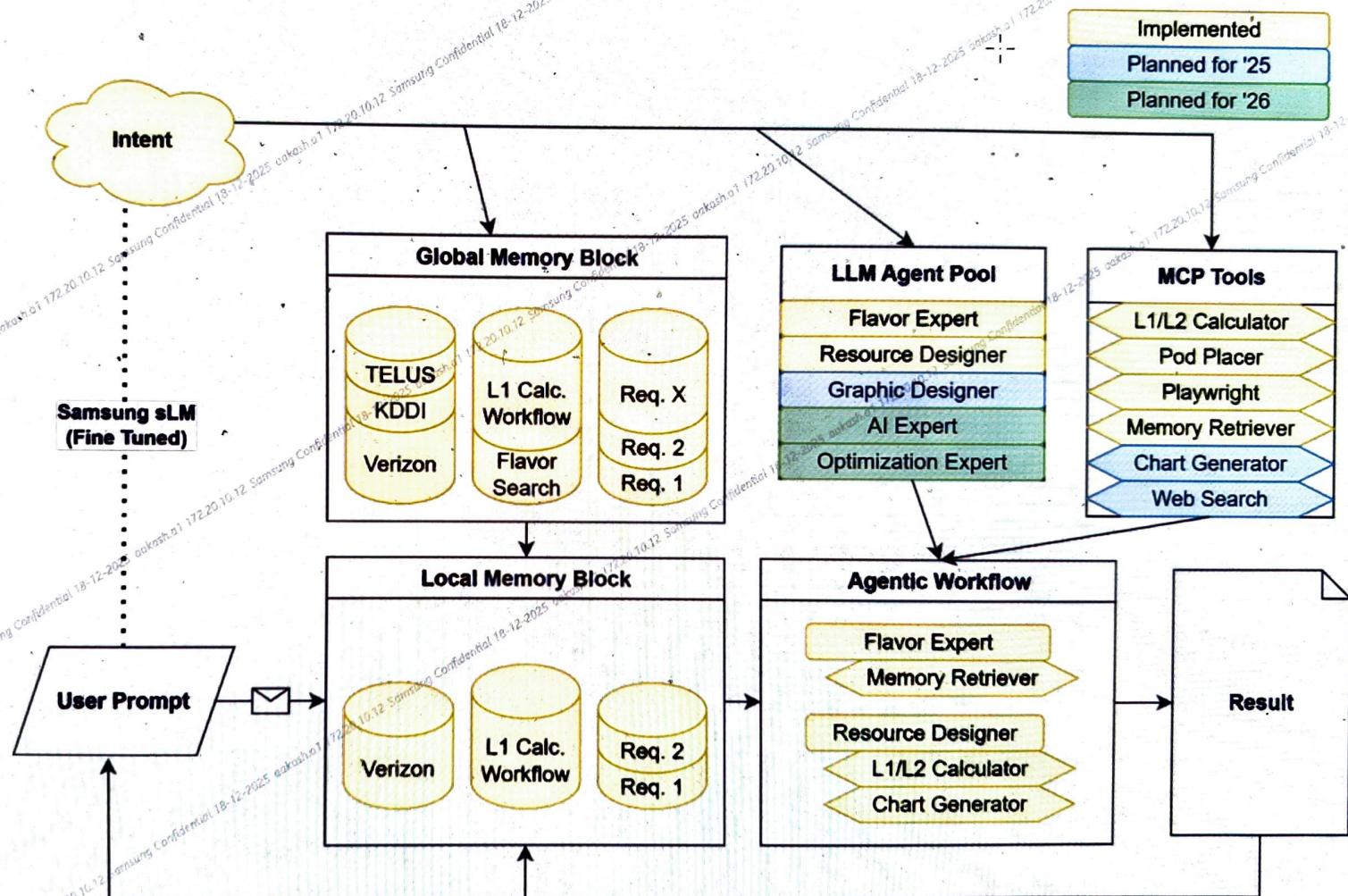
1. Update Pipeline:
 - a. L1/L2 Calculator backend update → Resource calculator field update → Data description update → Agent update
2. In-context learning:
 - a. RAG: getting only a snippet is a bad idea.
 - b. System Prompt: tends to skip some parameters
 - c. System Prompt + User Prompt: Best Performance.
 - d. Prompt Template:
 - i. Examples tend to confuse LLM.
3. Fine-tuning (TBD)
4. UE count agent:
 - a. Sometimes UE needs to be calculated
 - b. Simple LLM+agent seems to perform OK

Interwork flow (simple case): NetTune AI - Resource calculator_interworking workflow.pptx

NetTune AI - Resource calculator interworking workflow



3. MCP Tool List

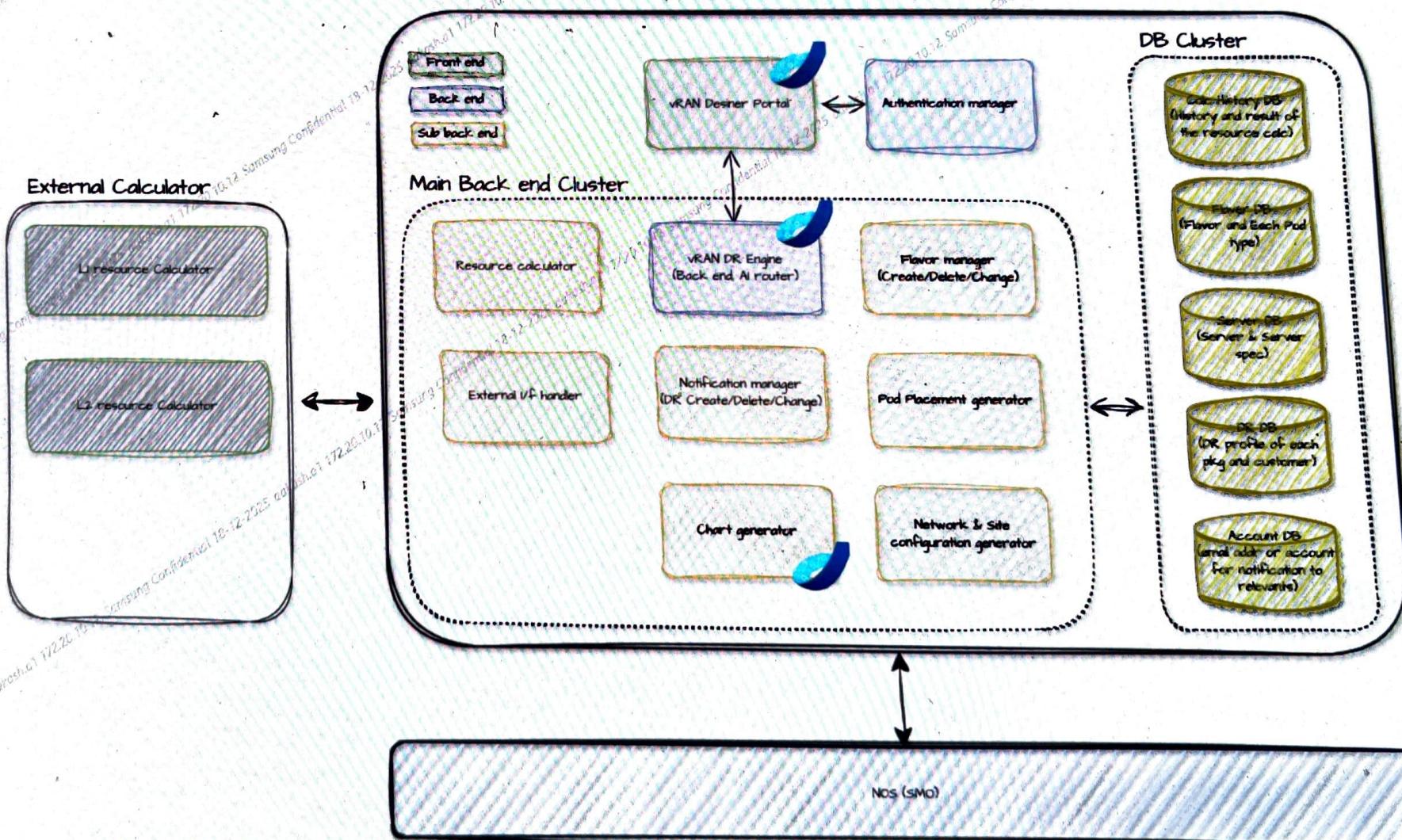


1. Resource Calculator
2. Pod Placer
3. Playwright
4. Memory Retriever
5. Chart Generator
6. Web Search

System Architecture of NetTune (updating)

NetTune Architecture

NetTune is a system consisted of Portal, Back end cluster and DB cluster.



Description of each component

- vRAN Designer Portal: is a gate of user who want to access this system and input any data to get information such as resource, server, and DR. (**Item applying AI**)
- Authentication manager: is an engine which provides join and login services and restricts to access service according to user level.
- Main back end cluster: is a group consisted of core back-ends of vRAN design system.
 - vRAN Designer engine: is a main back end service or agent to make the output for the user request. It provides various services; resource calculation, server search and recommend, DR design, and chart generation by interacting with various sub back ends. (**Item applying AI**)
 - Resource calculator: calculates the required core for the given cell combination and condition.
 - Flavor manager: creates/deletes/changes flavor. New flavor can be created with existing pod type or the result calculated by resource calculator. It also has a role to update Flavor DB when any flavor or pod is created/deleted/changed.

- Pod placement generator: creates pod placement which is included in DR (dimension rule). It shows how each pod configured in vRAN can be deployed on Kubernetes (K8s) cluster and the pod placement is used to assign a condition such as anti affinity rule when generating the chart.
- Network & Site configuration generator: creates network port and site configuration which is included in DR according to server spec and port condition such as the number of mh and fh. This is used by creating chart.
- Chart generator: creates vRAN chart which is used by SMO when deploying vRAN CNF to K8s cluster. Flavor, Pod placement and Network configuration are used as the basis for creating chart. (**Item applying AI**)
- Notification manager: informs the history and details to relevant by e-mail or any available account when DR is created/deleted/changed.
- External I/F handler: is a back end component which transmits and receives any message/event/data to/from external components. Various API and protocols can be defined and processed in this.
- Database (DB) cluster: is a group consisted of multiple DBs which support back end components.
 - Calc. History DB: manages database for the result of calculated resource.
 - Flavor DB: manages database for each of pod types and flavor list with consisted of specific pods.
 - DR DB: manages database for DR and its change history of each customer and PKG.
 - Server DB: manages database for server specification which is used for vRAN.
 - Account DB: manages database for email or account to give notification to the person in charge when DR is created/deleted/changed. This also includes user account used by Authentication manager.

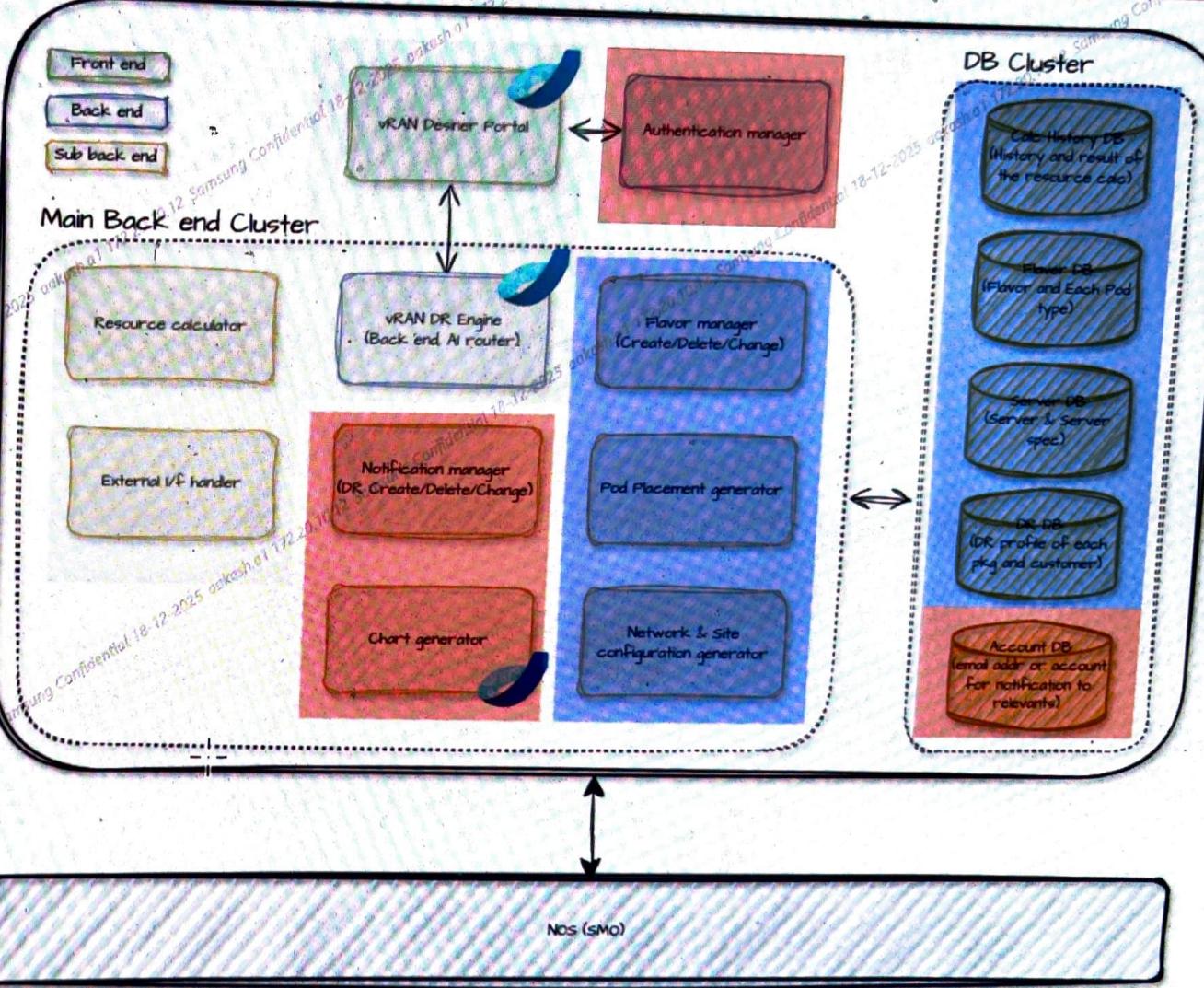
Plan of vRAN DR System development

- Ph1: vRAN designer portal / vRAN DR engine / Resource calc. / Ext I/F handler development

External Calculator



Main Back end Cluster



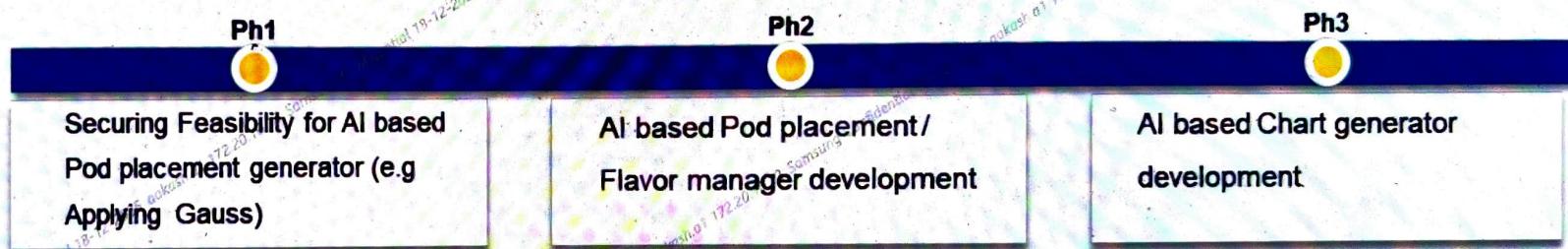
AI based interactive system

The goal is to build an AI based conversational system, but we plan to proceed with step-by-step development considering the infrastructure

AI based interactive system

The goal is to build an AI based conversational system; but we plan to proceed with step-by-step development considering the infrastructure and current situation.

- AI based Sub back end
 - Target back end: Pod placement generator / Flavor manager / Chart generator



- AI based Pod Placement generator: It generates pod placement for DR (Disaster Recovery) and shows how each pod configured in vRAN (Virtualized Radio Access Network) is deployed to a Kubernetes (K8s) cluster, with conditions such as anti-affinity rules assigned when generating charts.



AI based pod placement

LLM based Pod Placer

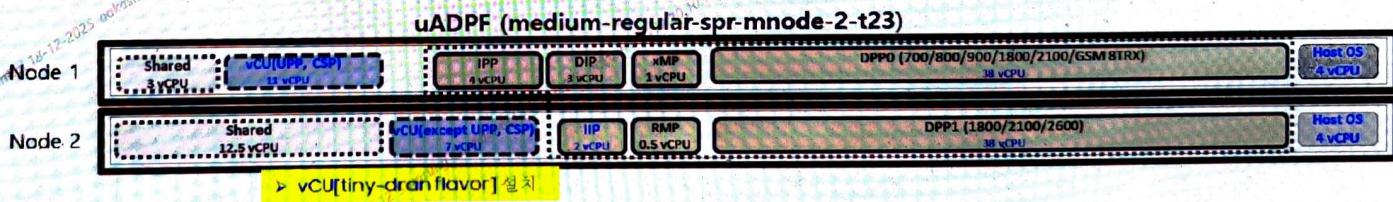
- Input: Operator, # Server, # CPU/server
- Output: Placement possibility, and pod placement result

Example:

Question

- vDÜ flavor: medium-regular-spr-mnode-2-t23
- Server spec: 32pcore
- Additional conditions: multi-server configuration, vCU deployment, DirectX2 applied

Answer



DB Format (subject to change)

DR Flavor Sheet

Operator	Network Function	Dimensioning Flavor	Package	DPP	DIP	DMP	CMP	PMP	RMP	IPP
Verizon	ADFP	medium-tdd-regular	25A	medium-tdd-re	medium-1.8m-s	medium-uni	medium-uni	medium-uni	medium-uni	none
VOS	uADPF	medium-regular-spr	25A	fdd-270m-18c-g	medium-uni	medium-uni	medium-uni	medium-uni	medium-uni	medium

Pod Flavor Resource Sheet

Pod type

- DPP
- DPP
- DIP
- DIP
- DMP

Ask AI Assistant

Summarize this document

By using AI Assistant, you agree to [Generative AI User Guidelines](#).

with a document summary using AI Assistant.

ce, you agree to the Adobe [Terms of Use](#) and [Privacy Policy](#)

▶ vCPU:tiny-dram flavor | 글씨

DB Format (subject to change)

DR Flavor Sheet

Operator	Network Function	Dimensioning Flavor	Package	DPP	DIP	DMP	CMP
Verizon	ADFP	medium-tdd-regular	25A	medium-tdd-re	medium-1.8m-s	medium-uni	med
VOS	uADFP	medium-regular-spr	25A	fdd-270m-18c-g	medium-uni	medium-uni	med

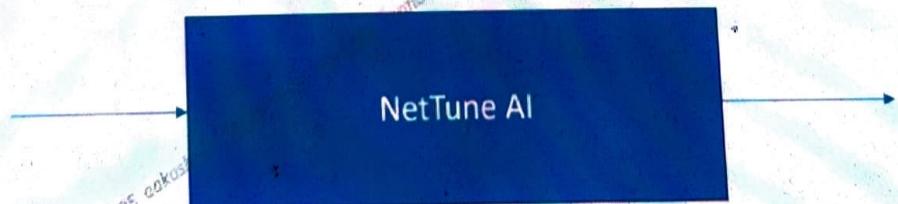
Pod Flavor Resource Sheet

Pod type	Pod flavor	vCPU Req	vCPU Limit	vMemory	Hugepage	Persistent Volume (GB)	Packet
DPP	medium-tdd-regular-spr	56	56	18.94	42.31	2(shared-pvc), 40(shared-log-pvc)	25A
DPP	fdd-270m-18c-gsm-6trx-spr	52	52	22.04	42.7	2(shared-pvc), 40(shared-log-pvc)	25A
DIP	medium-1.8m-spr	1	2	12	N/A	2(shared-pvc), 40(shared-log-pvc)	25A
DIP	medium-uni	2	3	12	N/A	2(shared-pvc), 40(shared-log-pvc)	25A
DMP	medium-uni	0.2	2	2	N/A	2(db-pvc), 4(shared-pvc), 40(shared-log-pvc)	25A
CMP	medium-uni	0.2	2	2	N/A	4(shared-pvc), 40(shared-log-pvc)	25A
PMP	medium-uni	0.1	2	2	N/A	4(shared-pvc), 40(shared-log-pvc)	25A
RMP	medium-uni	0.5	2	2	N/A	4(shared-pvc), 40(shared-log-pvc)	25A
IPP	medium	4	4	2	2	2(shared-pvc), 40(shared-log-pvc)	25A

Architecture (subject to change)

User Input:

1. Operator,
2. # Server,
3. # CPU/server



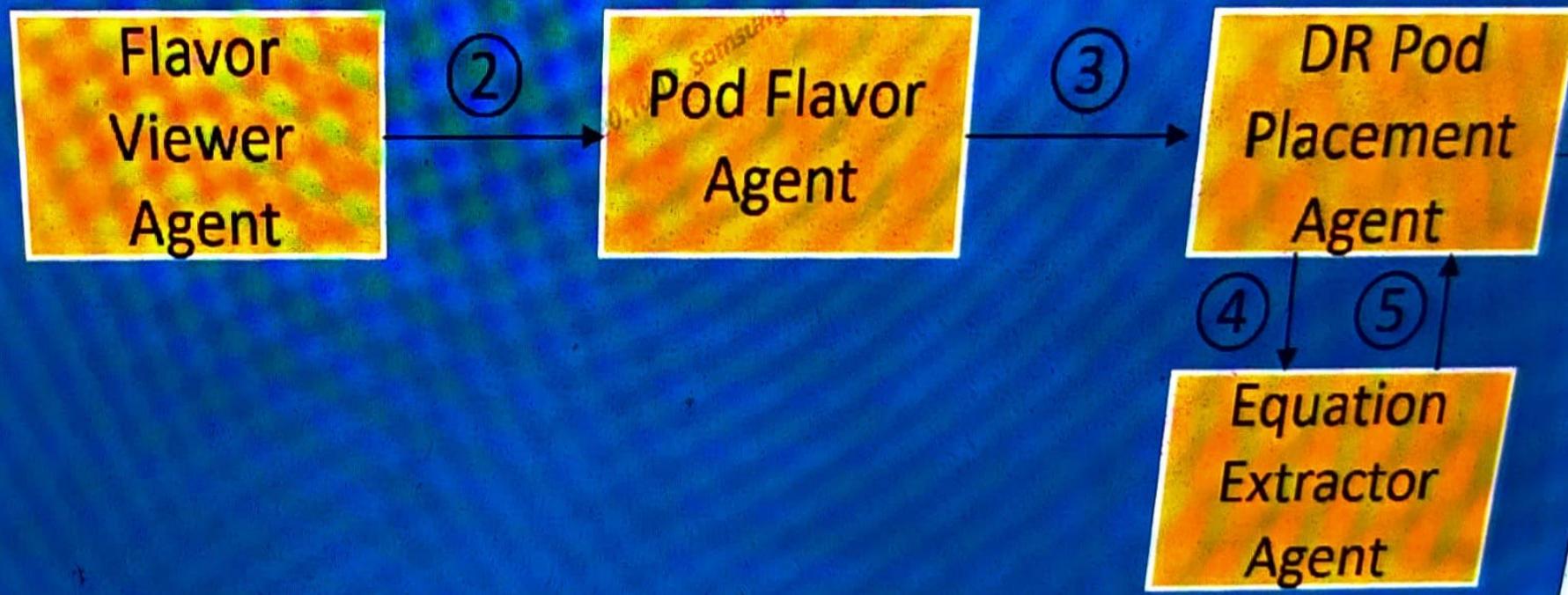
DPP: 52 core

...

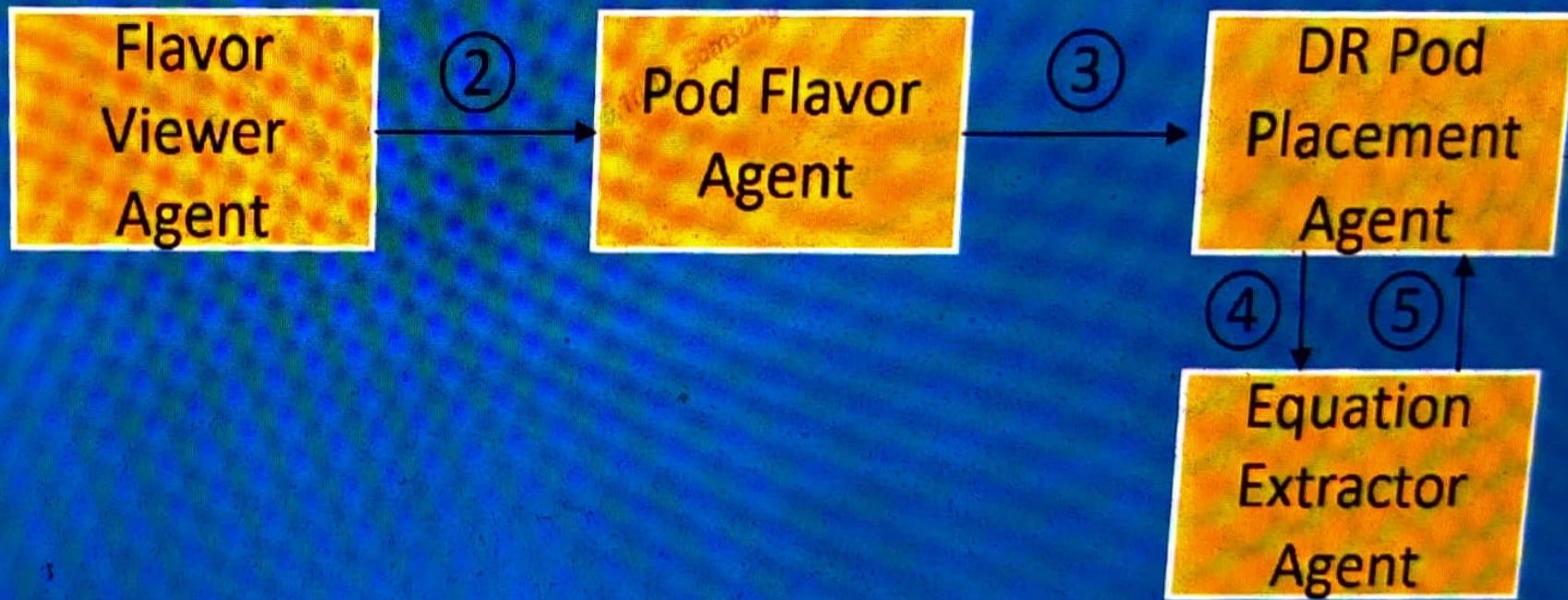
Thus,
Placement is p

LLM Agentic Workflow (subject to change)

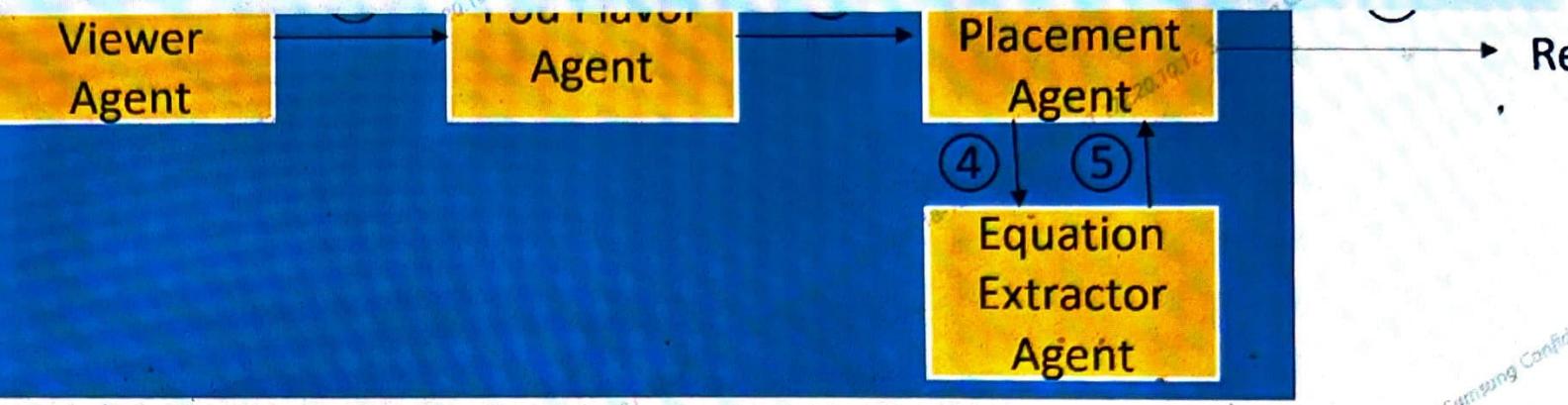
NetTune AI



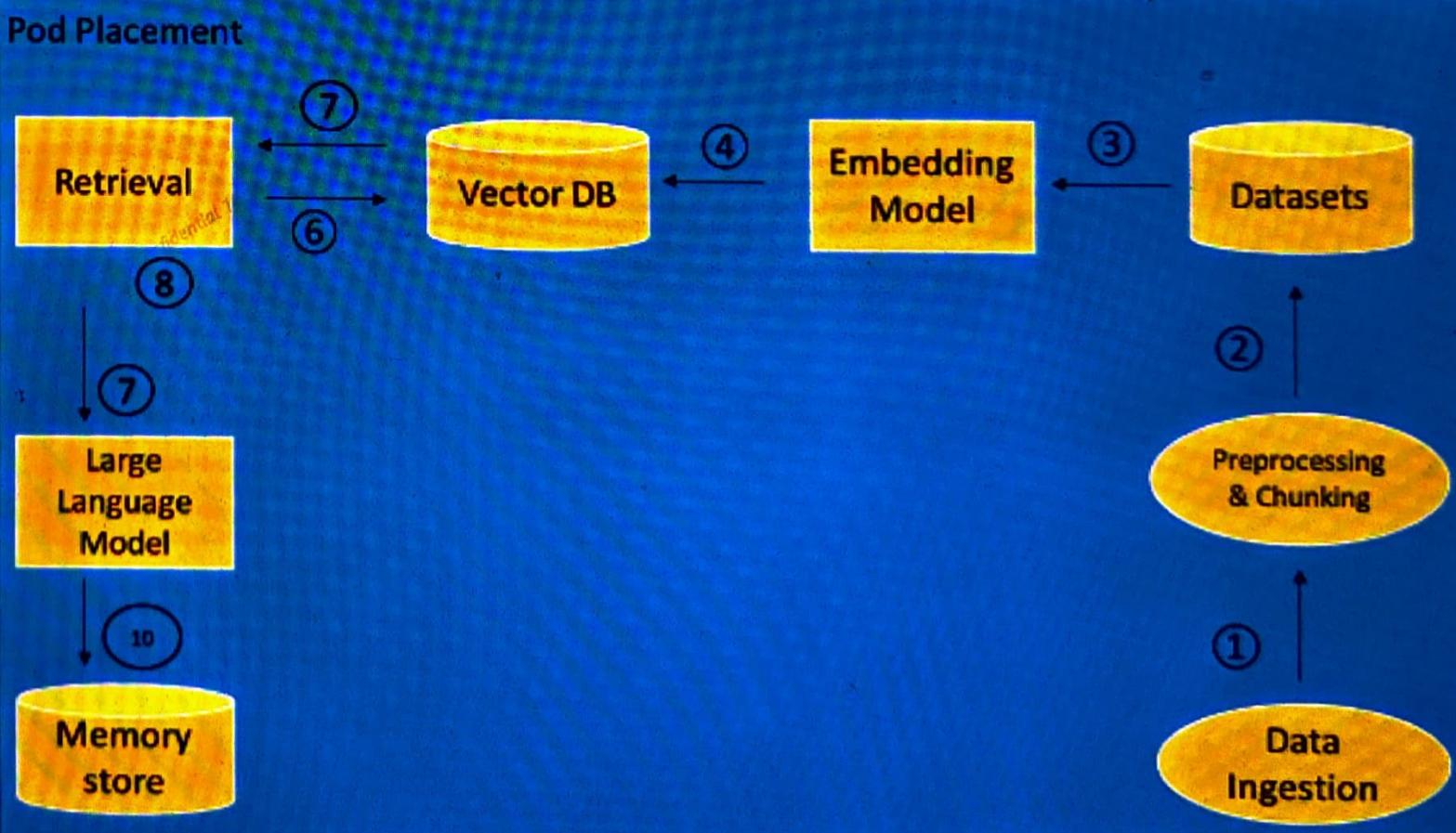
* ④, ⑤ are optional



* ④, ⑤ are optional



* ④, ⑤ are optional





Steps:-

1. Data Ingestion- load structured data.
2. Pre-processing & Chunking.
3. Embedding with Sentence Transformers-Convert each chunk into a vector(embedding).
4. Indexing in Vector DB- store embeddings in a searchable index.
5. User query.
6. Query Embedding-converts the user's query into a embedding.
7. Vector search – find similar chunks to the question.
8. Prompt Construction- Combine top chunks + user's question.
9. Call LLM- Sends query and relevant content as context.
10. Stores the query and response in memory.
11. Return response.

Github Repository - https://github.sec.samsung.net/aakash-a1/PodPlacement/tree/Qwen_RAG_revamped

*Please check the attachments for more detailed information.

*All information on this slide is tentative.

File

Modified