Volume II Quick Start

Deep Dive into DeepSeek — Volume II Quick Start

Yin Yang

Foundation Books Project

The one rule of this chapter

A handoff may change the request's *representation* or *owner*.
It may not change its *identity* or *meaning*.

- Release pages and model cards state claims about behavior, interfaces, architecture, and performance. This quick start calls each one a **stated capability**.
- Volume I explained the model and training mechanisms behind those capabilities. Volume II asks what must happen for one of them to stay *valid* and *correctly scoped* in a working stack.
- So the volume needs a request to follow, and a discipline for not quietly swapping one kind of evidence for another.

The one rule of this chapter

A handoff may change the request's *representation* or *owner*.
It may not change its *identity* or *meaning*.

- Release pages and model cards state claims about behavior, interfaces, architecture, and performance. This quick start calls each one a **stated capability**.
- Volume I explained the model and training mechanisms behind those capabilities. Volume II asks what must happen for one of them to stay *valid* and *correctly scoped* in a working stack.
- So the volume needs a request to follow, and a discipline for not quietly swapping one kind of evidence for another.

Goal: install `REQUEST-TRACE` — one logical request, three separately sourced evidence lanes — plus the carried objects and the identity relation every later systems chapter is audited against.

Roadmap

1. A capability becoming a system

Roadmap

1. A capability becoming a system
2. Objects that must stay attached

Roadmap

1. A capability becoming a system
2. Objects that must stay attached
3. The trace across Volume II

Roadmap

1. A capability becoming a system
2. Objects that must stay attached
3. The trace across Volume II
4. From concept to code

Roadmap

1. A capability becoming a system
2. Objects that must stay attached
3. The trace across Volume II
4. From concept to code
5. Carrying it forward

A capability becoming a system

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:

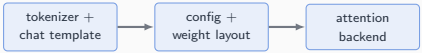


a tokenizer and chat template produce model input.

A config and weight layout identify the loaded model.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

A config and weight layout identify the loaded model.

An attention backend owns the resident state the next token can read.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

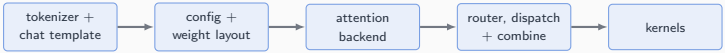
A config and weight layout identify the loaded model.

An attention backend owns the resident state the next token can read.

A router selects experts, dispatch moves token rows, combine returns them to the request.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

A config and weight layout identify the loaded model.

An attention backend owns the resident state the next token can read.

A router selects experts, dispatch moves token rows, combine returns them to the request.

Kernels interpret layouts, scales, dtypes, masks, and output destinations.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

A config and weight layout identify the loaded model.

An attention backend owns the resident state the next token can read.

A router selects experts, dispatch moves token rows, combine returns them to the request.

Kernels interpret layouts, scales, dtypes, masks, and output destinations.

Storage and measurement infrastructure decide what can be resumed, compared, or trusted later.

From a stated capability to an owned object

Definition (Stated capability)
Any claim a DeepSeek release page or model card makes about behavior, interfaces, architecture, or performance. Once the claim leaves the release page it becomes a chain of *owned objects*:



a tokenizer and chat template produce model input.

A config and weight layout identify the loaded model.

An attention backend owns the resident state the next token can read.

A router selects experts, dispatch moves token rows, combine returns them to the request.

Kernels interpret layouts, scales, dtypes, masks, and output destinations.

Storage and measurement infrastructure decide what can be resumed, compared, or trusted later.

Every arrow in that chain is a handoff, and every handoff is a place a capability can quietly stop being the thing that was claimed.

One logical request, one fixed prompt

The running example is `REQUEST-TRACE`. The observed run used this prompt, reproduced exactly:

Return a JSON object with keys status and note. Set status to ok and keep note under six words.

- The wording specifies a response *shape* and stipulates the value `ok`; the short `note` is otherwise unconstrained.

One logical request, one fixed prompt

The running example is `REQUEST-TRACE`. The observed run used this prompt, reproduced exactly:

Return a JSON object with keys status and note. Set status to ok and keep note under six words.

- The wording specifies a response *shape* and stipulates the value `ok`; the short `note` is otherwise unconstrained.
- The two fields are a *stable input marker* for the trace — not a test of model or serving-system health.

One logical request, one fixed prompt

The running example is `REQUEST-TRACE`. The observed run used this prompt, reproduced exactly:

Return a JSON object with keys status and note. Set status to ok and keep note under six words.

- The wording specifies a response *shape* and stipulates the value `ok`; the short `note` is otherwise unconstrained.
- The two fields are a *stable input marker* for the trace — not a test of model or serving-system health.
- Generation is limited to two new tokens, so the run records request identity and internal handoffs rather than a completed answer. [OBSERVED: job 371933, two H200s]

One logical request, one fixed prompt

The running example is `REQUEST-TRACE`. The observed run used this prompt, reproduced exactly:

Return a JSON object with keys status and note. Set status to ok and keep note under six words.

- The wording specifies a response *shape* and stipulates the value `ok`; the short `note` is otherwise unconstrained.
- The two fields are a *stable input marker* for the trace — not a test of model or serving-system health.
- Generation is limited to two new tokens, so the run records request identity and internal handoffs rather than a completed answer. [OBSERVED: job 371933, two H200s]

The fixture is deliberately boring. A running example earns its keep by being reproducible and checkable, not by being impressive.

Three lanes, one label

Lane	Fixed record	What the record supports
`/OBSERVED`	21 input ids; output ids $[185, 185]$; TP2/EP2; dynamic FP8 weights; BF16 KV cache; DeepEP normal mode; FlashMLA; page size 64	A two-H200 capability-and-handoff record for the listed state

Three lanes, one label

Lane	Fixed record	What the record supports
`/OBSERVED`	21 input ids; output ids [185, 185]; TP2/EP2; dynamic FP8 weights; BF16 KV cache; DeepEP normal mode; FlashMLA; page size 64	A two-H200 capability-and-handoff record for the listed state
`/ARCHITECTURE`	DeepSeek-V3 width 7168, 61 layers, 128 heads, KV latent rank 512 plus RoPE width 64, 256 routed experts, 1 shared, top-8 routing	Configuration coordinates for architecture and scaled worked projections

Three lanes, one label

Lane	Fixed record	What the record supports
`/OBSERVED`	21 input ids; output ids [185, 185]; TP2/EP2; dynamic FP8 weights; BF16 KV cache; DeepEP normal mode; FlashMLA; page size 64	A two-H200 capability-and-handoff record for the listed state
`/ARCHITECTURE`	DeepSeek-V3 width 7168, 61 layers, 128 heads, KV latent rank 512 plus RoPE width 64, 256 routed experts, 1 shared, top-8 routing	Configuration coordinates for architecture and scaled worked projections
`/SCALED`	Origin row ρ_7 on rank 0, route slot 0, logical expert 6, weight 0.65, selected owner rank 2, alternate eligible rank 6	Ownership, placement, dispatch, and replay arithmetic at a hand-checkable scale

Three lanes, one label

Lane	Fixed record	What the record supports
`/OBSERVED`	21 input ids; output ids [185, 185]; TP2/EP2; dynamic FP8 weights; BF16 KV cache; DeepEP normal mode; FlashMLA; page size 64	A two-H200 capability-and-handoff record for the listed state
`/ARCHITECTURE`	DeepSeek-V3 width 7168, 61 layers, 128 heads, KV latent rank 512 plus RoPE width 64, 256 routed experts, 1 shared, top-8 routing	Configuration coordinates for architecture and scaled worked projections
`/SCALED`	Origin row ρ_7 on rank 0, route slot 0, logical expert 6, weight 0.65, selected owner rank 2, alternate eligible rank 6	Ownership, placement, dispatch, and replay arithmetic at a hand-checkable scale

Three independently sourced inputs sharing one request label. Sharing a label does not make them one execution record.

The rule that keeps the lanes honest

Invariant (Lane discipline)
A field may move to another lane only as a labeled worked projection. It may never silently replace the canonical request.

- A useful trace keeps each value attached to its observation lane — so an observed cache shape supports only the backend that produced it.

The rule that keeps the lanes honest

Invariant (Lane discipline)
A field may move to another lane only as a labeled worked projection. It may never silently replace the canonical request.

- A useful trace keeps each value attached to its observation lane — so an observed cache shape supports only the backend that produced it.
- Later chapters may project a small matrix, page table, or routed row from one lane to expose arithmetic. They must *name* the projection.

The rule that keeps the lanes honest

Invariant (Lane discipline)
A field may move to another lane only as a labeled worked projection. It may never silently replace the canonical request.

- A useful trace keeps each value attached to its observation lane — so an observed cache shape supports only the backend that produced it.
- Later chapters may project a small matrix, page table, or routed row from one lane to expose arithmetic. They must *name* the projection.
- The observed V2-Lite FlashMLA and MoE handoffs, the V3 dimensions in `ARCHITECTURE`, and the scaled expert route stay three independently sourced inputs.

The rule that keeps the lanes honest

Invariant (Lane discipline)
A field may move to another lane only as a labeled worked projection. It may never silently replace the canonical request.

- A useful trace keeps each value attached to its observation lane — so an observed cache shape supports only the backend that produced it.
- Later chapters may project a small matrix, page table, or routed row from one lane to expose arithmetic. They must *name* the projection.
- The observed V2-Lite FlashMLA and MoE handoffs, the V3 dimensions in `ARCHITECTURE`, and the scaled expert route stay three independently sourced inputs.

The Systems Foundations chapter turns this plain-language rule into formal trace and handoff contracts.

One numerical rule, carried the whole volume

Name	Payload width	Role in this book
BF16	two bytes	the higher-precision comparison path

- Those widths count *payload*. They do not count scale metadata, packing, padding, or alignment.

One numerical rule, carried the whole volume

Name	Payload width	Role in this book
BF16	two bytes	the higher-precision comparison path
FP8	one byte	one-byte payload formats

- Those widths count *payload*. They do not count scale metadata, packing, padding, or alignment.

One numerical rule, carried the whole volume

Name	Payload width	Role in this book
BF16	two bytes	the higher-precision comparison path
FP8	one byte	one-byte payload formats
FP4	four bits	four-bit payload schemes

- Those widths count *payload*. They do not count scale metadata, packing, padding, or alignment.

One numerical rule, carried the whole volume

Name	Payload width	Role in this book
BF16	two bytes	the higher-precision comparison path
FP8	one byte	one-byte payload formats
FP4	four bits	four-bit payload schemes

- Those widths count *payload*. They do not count scale metadata, packing, padding, or alignment.
- So a dtype label is never the whole operator. Every later low-precision account follows payload and scale interpretation through multiply, accumulation, and output storage.

One numerical rule, carried the whole volume

Name	Payload width	Role in this book
BF16	two bytes	the higher-precision comparison path
FP8	one byte	one-byte payload formats
FP4	four bits	four-bit payload schemes

- Those widths count *payload*. They do not count scale metadata, packing, padding, or alignment.
- So a dtype label is never the whole operator. Every later low-precision account follows payload and scale interpretation through multiply, accumulation, and output storage.

This is why the observed lane records "dynamic FP8 weights, FP8 dispatch payloads with FP32 scales, BF16 KV cache" instead of a single precision adjective. [OBSERVED: job 371933 protocol record]

Objects that must stay attached

The minimum systems set

Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?

The minimum systems set

Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?

The minimum systems set

Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?
Model identity	config, weight layout, tokenizer family, license surface, runtime registration	does the runtime name the same model object?

The minimum systems set

Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?
Model identity	config, weight layout, tokenizer family, license surface, runtime registration	does the runtime name the same model object?
Resident attention state	dense, latent, sparse, quantized, or compressed state the next step may read	do cache arithmetic and layout metadata refer to what is *actually* resident?

The minimum systems set

Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?
Model identity	config, weight layout, tokenizer family, license surface, runtime registration	does the runtime name the same model object?
Resident attention state	dense, latent, sparse, quantized, or compressed state the next step may read	do cache arithmetic and layout metadata refer to what is *actually* resident?
Routed token rows	token-to-expert records, scores, dispatch metadata, combine rule for one sparse step	can transport and placement reconstruct the original ownership?

The minimum systems set

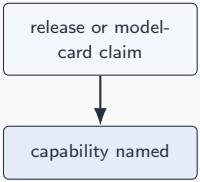
Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?
Model identity	config, weight layout, tokenizer family, license surface, runtime registration	does the runtime name the same model object?
Resident attention state	dense, latent, sparse, quantized, or compressed state the next step may read	do cache arithmetic and layout metadata refer to what is *actually* resident?
Routed token rows	token-to-expert records, scores, dispatch metadata, combine rule for one sparse step	can transport and placement reconstruct the original ownership?
Observed behavior	output, latency, throughput, correctness result, or storage state a later component trusts	same workload, backend, and hidden state?

The minimum systems set

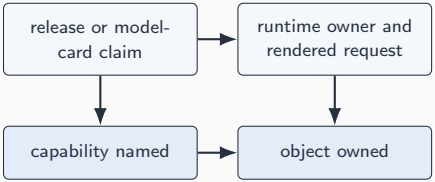
Carried object	Minimal meaning	Handoff test
Stated capability	long-context, MLA, routed-expert, API, or performance behavior being preserved	do mechanisms keep it *specific* rather than generic support?
Rendered request	tokenizer, chat template, message schema, generation settings, context accounting	is the same input contract preserved before behavior is compared?
Model identity	config, weight layout, tokenizer family, license surface, runtime registration	does the runtime name the same model object?
Resident attention state	dense, latent, sparse, quantized, or compressed state the next step may read	do cache arithmetic and layout metadata refer to what is *actually* resident?
Routed token rows	token-to-expert records, scores, dispatch metadata, combine rule for one sparse step	can transport and placement reconstruct the original ownership?
Observed behavior	output, latency, throughput, correctness result, or storage state a later component trusts	same workload, backend, and hidden state?

Each row names state that later chapters may *transform* but must not silently *replace*.

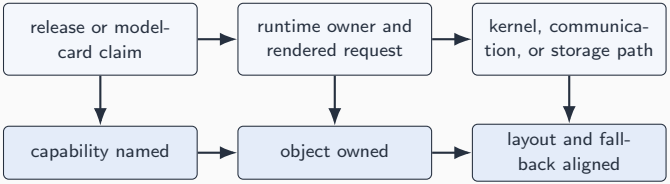
The capability bridge



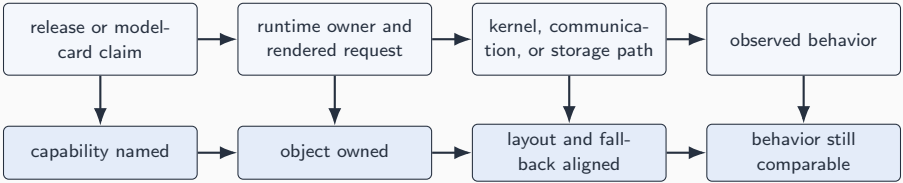
The capability bridge



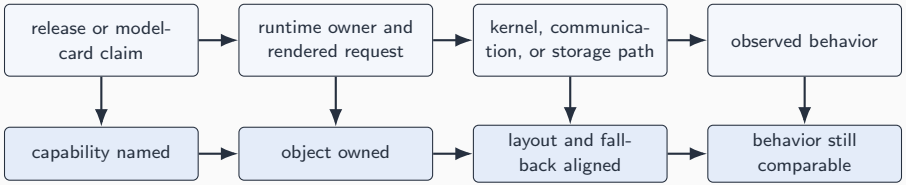
The capability bridge



The capability bridge



The capability bridge



Top row: the object moving into infrastructure. Bottom row: the check that keeps the handoff attached to the same capability.

This is a handoff diagram, not a new architecture. Each check identifies the object, its owner, and the state the next component may safely rely on.

The trace across Volume II

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured
Architecture projection	V3 latent width 512, RoPE width 64, 256 routed experts, top-8 routing	shapes and conditional schemas only; never an observed route

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured
Architecture projection	V3 latent width 512, RoPE width 64, 256 routed experts, top-8 routing	shapes and conditional schemas only; never an observed route
Scaled ownership	ρ_7, expert 6, weight 0.65, owner rank 2, eligible ranks $\{2, 6\}$	combine returns the row only for the matching generation and completion

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured
Architecture projection	V3 latent width 512, RoPE width 64, 256 routed experts, top-8 routing	shapes and conditional schemas only; never an observed route
Scaled ownership	ρ_7, expert 6, weight 0.65, owner rank 2, eligible ranks $\{2, 6\}$	combine returns the row only for the matching generation and completion
Reduced precision	observed FP8 weights and payloads with FP32 scales; BF16 KV cache	accuracy still requires its own numerical check

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured
Architecture projection	V3 latent width 512, RoPE width 64, 256 routed experts, top-8 routing	shapes and conditional schemas only; never an observed route
Scaled ownership	ρ_7, expert 6, weight 0.65, owner rank 2, eligible ranks $\{2, 6\}$	combine returns the row only for the matching generation and completion
Reduced precision	observed FP8 weights and payloads with FP32 scales; BF16 KV cache	accuracy still requires its own numerical check
Adjacent workloads	Mega MoE fused execution, DualPipe microbatches, 3FS checkpoints, AtlasLedger long-context requests	vocabulary reuse only; no state enters the observed request

Active branches, with their boundaries

Active branch	State carried forward	Handoff boundary
Observed attention	108 FlashMLA calls; BF16 cache shape $[47746, 64, 1, 576]$, stride $[36864, 576, 576, 1]$, page size 64	scheduler admission and block-table values stay outside
Observed routed MoE	156 route records; ordered DeepEP handoffs on ranks 0 and 1	physical traffic, timing, later placement updates unmeasured
Architecture projection	V3 latent width 512, RoPE width 64, 256 routed experts, top-8 routing	shapes and conditional schemas only; never an observed route
Scaled ownership	ρ_7, expert 6, weight 0.65, owner rank 2, eligible ranks $\{2, 6\}$	combine returns the row only for the matching generation and completion
Reduced precision	observed FP8 weights and payloads with FP32 scales; BF16 KV cache	accuracy still requires its own numerical check
Adjacent workloads	Mega MoE fused execution, DualPipe microbatches, 3FS checkpoints, AtlasLedger long-context requests	vocabulary reuse only; no state enters the observed request

Inactive mechanisms add no state to the observed lane.

Two jobs, two denominators

Record	What it bounds	Stamp
job 371933	request-scoped identity and ordered handoffs for one request	[OBSERVED: capability and handoff trace]

- The throughput record stays a *separate* observed anchor. Each job keeps its own denominator.

Two jobs, two denominators

Record	What it bounds	Stamp
job 371933	request-scoped identity and ordered handoffs for one request	[OBSERVED: capability and handoff trace]
job 348267	aggregate throughput over 16 measured requests on a closely matched TP2/EP2 stack	[OBSERVED: throughput record]

- The throughput record stays a *separate* observed anchor. Each job keeps its own denominator.

Two jobs, two denominators

Record	What it bounds	Stamp
job 371933	request-scoped identity and ordered handoffs for one request	[OBSERVED: capability and handoff trace]
job 348267	aggregate throughput over 16 measured requests on a closely matched TP2/EP2 stack	[OBSERVED: throughput record]

- The throughput record stays a *separate* observed anchor. Each job keeps its own denominator.
- Current-head behavior, production capacity, quality, and full-model DeepSeek-V3 remain outside *both* jobs. [OPEN: not executed in either job]

Two jobs, two denominators

Record	What it bounds	Stamp
job 371933	request-scoped identity and ordered handoffs for one request	[OBSERVED: capability and handoff trace]
job 348267	aggregate throughput over 16 measured requests on a closely matched TP2/EP2 stack	[OBSERVED: throughput record]

- The throughput record stays a *separate* observed anchor. Each job keeps its own denominator.
- Current-head behavior, production capacity, quality, and full-model DeepSeek-V3 remain outside *both* jobs. [OPEN: not executed in either job]

Two observed records about the same stack are still two records. Merging their denominators is the most tempting mistake in this volume.

The relation every later chapter is audited against

$$\text{next-owner identity} \;=\; \text{origin identity after the declared transformation}$$

- Every packing, routing, overlap, storage, or execution-window change must preserve **payload**, **metadata**, **order**, and **dependencies**.

The relation every later chapter is audited against

$$\text{next-owner identity} \;=\; \text{origin identity after the declared transformation}$$

- Every packing, routing, overlap, storage, or execution-window change must preserve **payload**, **metadata**, **order**, and **dependencies**.
- After a declared transformation, request, token, position, route, expert, placement generation, and completion identity all stay reconstructible.

The relation every later chapter is audited against

$$\text{next-owner identity} = \text{origin identity after the declared transformation}$$

- Every packing, routing, overlap, storage, or execution-window change must preserve **payload**, **metadata**, **order**, and **dependencies**.
- After a declared transformation, request, token, position, route, expert, placement generation, and completion identity all stay reconstructible.

Later chapters apply this audit with named objects, evidence, interpretations, owners, release conditions, oracles, and measurement boundaries.

From concept to code

From concept to code: the ARCHITECTURE lane's coordinates

The ARCHITECTURE lane is not a paraphrase — it is a pinned configuration file.

from **DeepSeek-V3/inference/configs/config_671B.json**

```
"dim": 7168, "n_layers": 61, "n_dense_layers": 3, "n_heads": 128,
"n_routed_experts": 256, "n_shared_experts": 1, "n_activated_experts": 8,
"kv_lora_rank": 512, "qk_rope_head_dim": 64, "dtype": "fp8"
```

From concept to code: the ARCHITECTURE lane's coordinates

The ARCHITECTURE lane is not a paraphrase — it is a pinned configuration file.

from **DeepSeek-V3/inference/configs/config_671B.json**

```
"dim": 7168, "n_layers": 61, "n_dense_layers": 3, "n_heads": 128,
"n_routed_experts": 256, "n_shared_experts": 1, "n_activated_experts": 8,
"kv_lora_rank": 512, "qk_rope_head_dim": 64, "dtype": "fp8"
```

These fields give the lane its shapes — and nothing else. They do not say that the logical request executed on V3, selected a backend, produced output, or achieved a latency. [INSPECTED: DeepSeek-V3 inference/configs/config_671B.json]

From concept to code: combine returns the row

"Combine returns token rows to the request" has a literal home in the reference MoE forward pass.

from **DeepSeek-V3/inference/model.py**

```
weights, indices = self.gate(x)
y = torch.zeros_like(x)
for i in range(self.experts_start_idx, self.experts_end_idx):
    idx, top = torch.where(indices == i)
    y[idx] += expert(x[idx]) * weights[idx, top, None]
```

From concept to code: combine returns the row

"Combine returns token rows to the request" has a literal home in the reference MoE forward pass.

from **DeepSeek-V3/inference/model.py**

```
weights, indices = self.gate(x)
y = torch.zeros_like(x)
for i in range(self.experts_start_idx, self.experts_end_idx):
    idx, top = torch.where(indices == i)
    y[idx] += expert(x[idx]) * weights[idx, top, None]
```

`idx` is the origin row identity and `top` is the route slot. The expert output re-enters the request at its *own* row, scaled by its *own* gate weight. Any faster transport must reproduce exactly that. [INSPECTED: DeepSeek-V3 `inference/model.py`]

Carrying it forward

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** → `book/_metadata/volume_ii_request_trace.json`

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** → `book/_metadata/volume_ii_request_trace.json`
- **the observed two-H200 request record** → `experiments/volume_ii_request_trace_panther/results/`

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** →
  `book/_metadata/volume_ii_request_trace.json`
- **the observed two-H200 request record** →
  `experiments/volume_ii_request_trace_panther/results/`
- **ARCHITECTURE lane configuration coordinates** →
  `DeepSeek-V3/inference/configs/config_671B.json`

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** → `book/_metadata/volume_ii_request_trace.json`
- **the observed two-H200 request record** → `experiments/volume_ii_request_trace_panther/results/`
- **ARCHITECTURE lane configuration coordinates** → `DeepSeek-V3/inference/configs/config_671B.json`
- **MLA cache objects an attention backend owns** → `DeepSeek-V3/inference/model.py`

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** → `book/_metadata/volume_ii_request_trace.json`
- **the observed two-H200 request record** → `experiments/volume_ii_request_trace_panther/results/`
- **ARCHITECTURE lane configuration coordinates** → `DeepSeek-V3/inference/configs/config_671B.json`
- **MLA cache objects an attention backend owns** → `DeepSeek-V3/inference/model.py`
- **the handle that keeps a routed row attached** → `DeepEP/README.md`

Concept-to-code map for the quick start

- **lane definitions, provenance classes, claim boundaries** → `book/_metadata/volume_ii_request_trace.json`
- **the observed two-H200 request record** → `experiments/volume_ii_request_trace_panther/results/`
- **ARCHITECTURE lane configuration coordinates** → `DeepSeek-V3/inference/configs/config_671B.json`
- **MLA cache objects an attention backend owns** → `DeepSeek-V3/inference/model.py`
- **the handle that keeps a routed row attached** → `DeepEP/README.md`

Change the representation, change the owner — keep the identity.

Next: Systems Foundations develops the CUDA, fabric, distributed-systems, pipeline, and measurement vocabulary used throughout Volume II, and turns this quick start's plain-language rule into the volume's formal trace and handoff contracts.