

Orchestration Domain-Centric Pattern

ODCP

The Semantic Domain Orchestration of the SDIA Ecosystem

Ricardo Luz Holanda Viana

Independent Solo Researcher | Enterprise Integration Architect
SAP BTP Integration Suite Expert | SAP Press e-Bite Author — Enterprise Messaging (2021)
rhviana@gmail.com · ORCID: 0009-0009-9549-5862

DEIP Ecosystem DOI: 10.5281/zenodo.19004802 · CC BY 4.0 International

Webpage: www.domain-intent.com (under construction) · GitHub: github.com/rhviana/deip (under reformulation)

Edition: April 2026 — Warsaw, Poland

IP Note: Architectural patterns, governance framework, and naming specifications disclosed herein are publicly available prior art via timestamped Zenodo publication. This disclosure may serve as prior-art evidence under 35 U.S.C. § 102. This document does not constitute legal advice.

Author Note: The concepts, architectural models, governance principles, and naming structures presented in this work were conceived and developed by Ricardo Luz Holanda Viana (February / March 2026). Generative AI tools (Claude — Anthropic, Gemini, Genspark) were used solely as assistive instruments for drafting, language refinement, formatting, images, and structural organization. All substantive ideas, technical decisions, and final validation remain the sole responsibility of the author.

v3.0 Edition — Final — Scope Disclaimer

ODCP governs the orchestration layer of SDIA: domain package topology, integration flow DNA naming, supporting object naming, and credential/certificate segmentation by domain and sub-domain.

This specification is modelled primarily with SAP BTP Cloud Integration (CPI) terminology, because CPI was the original validation environment. ODCP is not SAP-specific. Other orchestration platforms may use different object names, separators, and classifications, but they must preserve the same structural proposal: domain primacy, deterministic naming, and segmented governance.

ODCP defines a vendor-agnostic reference model with SAP CPI as the primary example language for practical implementation.

Abstract

ODCP resolves three anti-patterns (Package/Flow Sprawl, Naming Chaos, Credential Blackout) through three invariants:

- Domain-Centric Package Consolidation — one package per domain
- Deterministic Artifact Naming — machine-readable, traceable naming grammar
- Sub-Domain Tiered Credential Governance — blast radius control per credential

Originally validated in SAP CPI (as PDCP), generalized to any platform with configurable containers and naming. Validated reduction: 39→4 packages, 39→12 credentials.

The domain is the primary key. Packages become stable. Artifacts become machine-readable. Credentials become governable.

Companion Documents — DEIP / SDIA Ecosystem

Component	Full Name	Scope	DOI
DEIP	Domain Enterprise Integration Pattern	Decision model — platform binding	zenodo.19004802
SDIA	Semantic Domain Integration Architecture	Umbrella architecture (5 layers)	zenodo.18877635
GDCR	Gateway Domain-Centric Routing	Layer 1 — Gateway	zenodo.18582492
DDCR	Domain Driven-Centric Router	Layer 2 — Resolution engine	zenodo.18864832
ODCP	Orchestration Domain-Centric Pattern	Layer 3 — Orchestration (this document)	zenodo.18876593
EDCP	Event Domain-Centric Pattern	Layer 4 — Events / Messaging / Streaming	zenodo.19068766
DDCP	Data Domain-Centric Pattern	Layer 5 — Data	zenodo.19730519

Independent Research Disclaimer & Legal Notice

Independent research from first principles. No vendor docs. Trademarks belong to respective owners. CC BY 4.0 — reuse with attribution.

Author's Note

ODCP emerged from a repeated observation: regardless of vendor, packages multiply by project, names become non-deterministic, and credentials concentrate risk. The solution is not a new tool. It is a new invariant: the business domain is the primary key for orchestration structure.

Once applied, packages stabilize, names become machine-readable, and blast radius becomes controllable. ODCP is not a vendor recipe. It is a structural governance principle.

Introduction

ODCP defines a reference naming model for orchestration artifacts, not a rigid template.

The grammar presented in this document establishes the structural invariant: the domain token must always be the first semantic segment. Everything else — sectors, divisions, regions, qualifiers, and the specific granularity of each object — must be defined by each enterprise based on its own business needs, organizational structure, and governance maturity.

Some enterprises may need only `domain.subdomain.entity.purpose`.

Others may require `domain.sector.division.region.subdomain.entity.purpose`.

Both are valid as long as the domain remains the anchor.

The starting point is this document. The final shape is yours.

1 The Orchestration Problem

Enterprise integration platforms accumulate orchestration artifacts over long operational lifecycles. As integration volume grows, structural organization degrades packages multiply, artifact names become inconsistent, credentials concentrate operational risk, and supporting objects such as mappings, scripts, variables, certificates, and stores lose semantic cohesion. The root cause is not technological. It is organizational. Orchestration artifacts are typically grouped by vendor, project, or team rather than by business domain.

The consequence is an orchestration layer that answers only technical questions, not business ones. When an architect asks, “Where are all Sales integrations?”, the answer is rarely structural or deterministic. It is fragmented across packages, flows, mappings, variables, and credentials named after systems, temporary projects, and local implementation choices rather than domain semantics.

ODCP addresses this problem by making the business domain the primary organizational key of the orchestration layer.

Anti-Pattern	Manifestation	Consequence
Package Sprawl	Packages per vendor, project, or team	Manual discovery, fragmented ownership
Artifact Naming Chaos	Salesforce_Order_Sync_v3_FINAL	Non-deterministic, fragile automation
Technical User Blackout Risk	One credential shared across many flows	Single rotation = total blackout
Supporting Object Sprawl	Mappings, scripts, stores named ad hoc	Reuse collapses, lineage lost
Certificate and Trust store Sprawl	Certificates duplicated per project	Rotation risky, trust opaque

These are not tooling problems. They are structural problems caused by a single root cause: business semantics coupled to technical artifacts. The domain does not change. Technology does.

2. Core Governance Principle

The domain is the primary key of the orchestration layer.

Normative corollaries:

- **MUST:** One package per domain
- **MUST:** All artifacts contain domain in their name
- **MUST NOT:** Project names, vendor names, or team names as primary segment
- **SHOULD:** Credentials segmented by sub-domain tier

Conformance language: RFC 2119 (MUST / MUST NOT / SHOULD / MAY)

3. ODCP within SDIA

ODCP is Layer 3 of the SDIA ecosystem — the semantic topology governance layer for orchestration artifacts.

Layer	Component	Responsibility
1	GDCR	Gateway facade
2	DDCR	Deterministic routing
3	ODCP	Orchestration governance
4	EDCP	Event fabrics
5	DDCP	Data persistence

Semantic continuity: The domain token sales appear unchanged from GDCR URL → DDCR key → ODCP iFlow DNA → EDCP topic → DDCP table.

Platform Equivalence Note

3. The Three Artifacts and Their Grammar

The ODCP grammar governs three orchestration artifacts simultaneously:

- **Package DNA** — domain-scoped container for all integration flows
- **iFlow DNA** — deterministic flow identifier
- **Credential DNA** — domain-scoped technical user

All three share the same domain invariant and lexical constraints.

Concrete Example — Sales Domain (all granularity levels)

Level	Granularity	Package DNA	iFlow DNA	Credential DNA
1 — Baseline	domain only	acme.sales.o2c.integrations	id01.o2c.salesforce.order.s4hana.c.in.sync	cr.sales.orders.rw

Level	Granularity	Package DNA	iFlow DNA	Credential DNA
2 — With subdomain	domain + subdomain	acme.sales.o2c.integrations	id01.o2c.salesforce.order.s4hana.c.in.sync	cr.sales.orders.rw
3 — With sector	sector + domain	acme.retail.sales.o2c.integrations	id01.o2c.retail.salesforce.order.s4hana.c.in.sync	cr.sales.orders.retail.rw
4 — With division	division + domain	acme.consumer.sales.o2c.integrations	id01.o2c.consumer.salesforce.order.s4hana.c.in.sync	cr.sales.orders.consumer.rw
5 — With region	region + domain	acme.emea.sales.o2c.integrations	id01.o2c.emea.salesforce.order.s4hana.c.in.sync	cr.sales.orders.emea.rw
6 — Full combination	sector + division + region + domain	acme.retail.consumer.emea.sales.o2c.integrations	id01.o2c.retail.consumer.emea.salesforce.order.s4hana.c.in.sync	cr.sales.orders.retail.consumer.emea.rw

Alignment Invariant

Across all levels, the three artifacts remain aligned:

Level	Example
Baseline	acme.sales.o2c.integrations ↔ id01.o2c.salesforce.order.s4hana.c.in.sync ↔ cr.sales.orders.rw
Extended	acme.retail.emea.sales.o2c.integrations ↔ id01.o2c.retail.emea.salesforce.order.s4hana.c.in.sync ↔ cr.sales.orders.retail.emea.rw

The domain token sales appears in identical form across all three artifacts — unchanged regardless of optional segments added before it.

4. Formal Naming Grammar (ABNF)

The following ABNF (RFC 5234) grammar defines ODCP naming patterns. Grammar is normative — non-conformant names are not ODCP-compliant.

Adaptation Note: This grammar must be adapted to each enterprise according to its specific needs and the levels of granularity required (sector, division, region, qualifiers). This document serves only as the **origin point** — a reference model. Enterprises may add, remove, or reorder optional segments as long as **the domain token remains the first structural segment**. The domain must always be respected.

PVRL Rules (apply to all segments):

- PVRL-1: Every segment MUST begin with [a-z0-9]
- PVRL-2: Hyphens only in non-initial, non-final positions; no consecutive hyphens
- PVRL-3: All segments MUST be lowercase
- PVRL-4: Platform restrictions documented as exceptions in SDIA Domain Catalog

```

=====
; ODCP — Orchestration Domain-Centric Pattern
; PVRL Level 3 — ABNF Grammar for Orchestration Artifacts
; RFC 5234 notation
=====

label    = ALNUM *( ALNUM / "-" ALNUM )
domain   = 1*ALPHA-LOWER ; business domain — never abbreviated

```

```

subdomain = 1*ALPHA-LOWER      ; e.g., o2c, r2r, s2p, payroll
sector    = 1*ALPHA-LOWER      ; optional · industry sector
division  = 1*ALPHA-LOWER      ; optional · business division
region    = 1*ALPHA-LOWER      ; optional · geography
entity    = 1*ALPHA-LOWER      ; business entity (order, invoice, employee)
actioncode = "c" / "r" / "u" / "d" / "s" / "a" / "n" / "t" / "e" / "b" / "v" / "w" / "x" / "z" / "f"
direction = "in" / "out"
mode      = "sync" / "async"
purpose   = 1*ALPHA-LOWER      ; operational purpose
role      = 1*ALPHA-LOWER      ; e.g., header, item, dim, fact
seq       = 2DIGIT             ; 01-99
permission = "rw" / "ro" / "rw-batch" / "ro-stream"
field     = 1*ALPHA-LOWER      ; e.g., status, type, amount
context   = 1*ALPHA-LOWER      ; e.g., order, payment, shipping
name      = 1*(ALPHA-LOWER / DIGIT / "-")
company   = 1*ALPHA-LOWER      ; organizational tenant prefix
sender    = 1*(ALPHA-LOWER / DIGIT) ; source system identifier
receiver  = 1*(ALPHA-LOWER / DIGIT) ; target system identifier
system    = 1*(ALPHA-LOWER / DIGIT / "-")
protocol  = "http" / "https" / "soap" / "jms" / "amqp" / "mqtt" / "sftp" / "odata"
qualifier = sector / division / region
ALNUM     = ALPHA / DIGIT
ALPHA-LOWER = %x61-7A          ; a-z only

; =====
; 4.2 Package DNA — One package per domain
; =====
package-name = company [ "." sector ] [ "." division ] [ "." region ] "." domain "." subdomain ".integrations"

; Examples — baseline
; acme.sales.o2c.integrations
; acme.finance.r2r.integrations
; acme.logistics.le.integrations

; Examples — extended with optional segments
; acme.retail.sales.o2c.integrations
; acme.emea.sales.o2c.integrations
; acme.retail.emea.sales.o2c.integrations

; =====
; 4.3 iFlow DNA — Deterministic flow naming
; =====
iflow-name = "id" seq "." subdomain [ "." sector ] [ "." division ] [ "." region ] "." sender "." entity "." receiver "." actioncode
            "." direction "." mode

; Examples — baseline
; id01.o2c.salesforce.order.s4hana.c.in.sync
; id02.o2c.sap.customer.salesforce.r.out.async
; id13.r2r.workday.invoice.s4hana.c.in.sync

; Examples — extended with optional segments
; id01.o2c.retail.salesforce.order.s4hana.c.in.sync
; id01.o2c.emea.salesforce.order.s4hana.c.in.sync
; id01.o2c.retail.emea.salesforce.order.s4hana.c.in.sync

; =====
; 4.4 Credential DNA — Segmented by sub-domain and permission
; =====
credential-name = "cr." domain "." subdomain [ "." sector ] [ "." division ] [ "." region ] "." permission

; Examples — baseline
; cr.sales.orders.rw
; cr.sales.analytics.ro
; cr.finance.payroll.rw-batch

```

```

; Examples — extended with optional segments
; cr.sales.orders.retail.rw
; cr.sales.orders.emea.ro
; cr.sales.orders.retail.emea.rw-batch

; =====
; 4.5 Value Mapping — Domain-scoped lookup tables
; =====
value-mapping = "vm." domain "." entity "." field [ "." qualifier ]

; Examples — baseline
; vm.sales.order.status
; vm.finance.invoice.type
; vm.logistics.shipment.carrier

; Examples — extended with optional segments
; vm.sales.order.status.retail
; vm.sales.order.status.emea
; vm.sales.order.status.retail.emea

; =====
; 4.6 Message Mapping — Transformation between sender and receiver
; =====
message-mapping = "mm." domain "." entity "." sender "." receiver [ "." qualifier ]

; Examples — baseline
; mm.sales.order.sfdc.s4hana
; mm.finance.invoice.workday.s4hana
; mm.logistics.shipment.sap.dynamics

; Examples — extended with optional segments
; mm.sales.order.sfdc.s4hana.retail
; mm.sales.order.sfdc.s4hana.emea
; mm.sales.order.sfdc.s4hana.retail.emea

; =====
; 4.7 Script Collection — Reusable scripts by domain and purpose
; =====
script-collection = "sc." domain "." entity "." purpose [ "." qualifier ]

; Examples — baseline
; sc.sales.order.transformation
; sc.finance.invoice.validation
; sc.logistics.shipment.routing

; Examples — extended with optional segments
; sc.sales.order.transformation.retail
; sc.sales.order.transformation.emea

; =====
; 4.8 Data Store — Persistent storage for orchestration state
; =====
data-store = "ds." domain "." entity "." purpose [ "." qualifier ]

; Examples — baseline
; ds.sales.order.pending
; ds.finance.invoice.archive
; ds.logistics.shipment.tracking

; Examples — extended with optional segments
; ds.sales.order.pending.retail
; ds.sales.order.pending.emea

```

```

; =====
; 4.9 Variable — Flow-level variables scoped by domain context
; =====
variable = "var." domain "." context "." name [ "." qualifier ]

; Examples — baseline
; var.sales.order.batchsize
; var.finance.invoice.retrycount
; var.logistics.shipment.timeout

; Examples — extended with optional segments
; var.sales.order.batchsize.retail
; var.sales.order.batchsize.emea

; =====
; 4.10 Certificate and Trust Store — Security material scoped by domain
; =====
certificate-name = "cert." domain "." subdomain "." purpose [ "." qualifier ]
truststore-name = "trust." domain "." subdomain [ "." qualifier ]

; Examples — baseline
; cert.sales.o2c.client
; cert.finance.r2r.server
; trust.sales.o2c

; Examples — extended with optional segments
; cert.sales.o2c.client.retail
; trust.sales.o2c.emea

; =====
; 4.11 Channel / Connection — Connectivity configuration
; =====
channel-name = "ch." domain "." subdomain "." system "." protocol [ "." qualifier ]

; Examples — baseline
; ch.sales.o2c.salesforce.http
; ch.finance.r2r.s4hana.soap
; ch.logistics.le.dhl.sftp

; Examples — extended with optional segments
; ch.sales.o2c.salesforce.http.retail
; ch.sales.o2c.salesforce.http.emea

; =====
; 4.12 Extension to Other Orchestration Artifacts
; =====
; The same domain-first naming convention applies to all orchestration layer
; objects, including but not limited to:
;
; Queues (orchestration internal queues):    q.domain.subdomain.entity.purpose
; Schedules (timers, event triggers):        sch.domain.subdomain.purpose
; Correlations (message correlation identifiers): corr.domain.subdomain.entity
; Adapters (protocol adapter configurations):  adp.domain.subdomain.system.protocol
; Error Handlers (exception subprocesses):    err.domain.subdomain.entity
; Fragments (reusable sub-flows):            frg.domain.subdomain.entity.purpose
; Properties (configuration properties):      prop.domain.context.name
; Resource Dictionaries (i18n):              res.domain.language

; The specific prefix is platform-dependent and must be adapted according to
; the target orchestration platform's naming constraints.
;
; The invariant remains: the domain token is the first semantic segment
; after any required platform prefix.

```



```

;=====
; 4.13 Valid and Invalid Examples
;=====
; Artifact Name      | Verdict | Rationale
;-----|-----|-----
; acme.sales.o2c.integrations | VALID | Domain first; valid structure
; cr.sales.orders.rw | VALID | Valid credential pattern
; id01.o2c.salesforce.order.s4hana.c.in.sync | VALID | Valid iFlow DNA
; vm.sales.order.status | VALID | Valid value mapping
; Sales.Sales.Order.Package | INVALID | Uppercase rejected (PVRL-3)
; project-phoenix.integrations | INVALID | No domain token (MUST NOT)
; cr.sales.rw | INVALID | Empty segment (PVRL-2)
; id01..salesforce.order.s4hana.c.in.sync | INVALID | Consecutive separators (PVRL-2); RFC 5234 ABNF notation

```

5. Platform Equivalence Note

The examples above use SAP CPI terminology (Package, iFlow, Value Mapping, etc.). Other platforms have equivalent artifacts:

SAP CPI	MuleSoft	Boomi	Azure Logic Apps
Package	Application	Folder	Logic App
iFlow	Flow	Process	Workflow
Value Mapping	Lookup Table	Cross Reference Table	Mapping
Script Collection	Script	Custom Scripting	Inline Code
Data Store	Object Store	Atom Queue	Storage Account

The structural invariant is identical across all platforms. Adapt the separator and object name, preserve the domain-first naming convention.

6. PVRL Validation Rules

These rules are normative and **MUST** be enforced at provisioning time.

Rule	Specification
PVRL-1	Every segment MUST begin with [a-z0-9]
PVRL-2	Hyphens only in non-initial, non-final positions; no consecutive hyphens
PVRL-3	All segments MUST be lowercase
PVRL-4	Platform restrictions documented as exceptions in SDIA Domain Catalog
Data Store	Object Store

7. Sub-Domain Tiered Credential Governance

ODCP segments credentials by business criticality and sub-domain scope. This isolates the blast radius of credential rotation events.

Tier	Technical User	Scope	Rotation Cycle
Tier 1	cr.sales.orders.rw	Transactional — Order-to-Cash	30 days
Tier 2	cr.sales.analytics.ro	Reporting / Analytics	90 days
Tier 3	cr.sales.vendor.rw-batch	Third-party / Batch	60 days

A password rotation failure in Tier 1 affects only order-management flows — not analytics, not other domains.

8. Empirical Validation

Metric	Before ODCP	After ODCP	Improvement
Packages	39	4	↓ 90%
Technical Users	39	12	↓ 69%
Discovery Time	Manual search	Deterministic by domain	~90% faster
Deployment Time (4 domains)	273 min	14.5 min	↓ 95%

Validated on SAP BTP CPI as the reference platform. The same pattern applies to any orchestration platform.

9. Conformance Checklist

#	Criterion	MUST/SHOULD
1	One package per domain	MUST
2	Package name follows company.domain.subdomain.integrations	MUST
3	iFlow name follows iFlow DNA grammar	MUST
4	Credentials follow cr.domain.subdomain.permission	MUST
5	Supporting objects follow domain-first naming	SHOULD
6	No project/vendor/team names as primary segment	MUST NOT
7	Certificates follow cert.domain.subdomain.purpose	SHOULD
8	All artifacts registered in SDIA Domain Catalog	SHOULD

10. Conclusion

ODCP completes the SDIA semantic governance stack for the orchestration layer. From packages to flows to credentials to mappings to certificates — the domain is the primary key.

ODCP does not introduce new technology. It introduces structural discipline. The domain is the map. The orchestration platform is the territory. A map that matches the territory never lies.

11 References

- [1] R. L. H. Viana, "Gateway Domain-Centric Routing (GDCR) — v8.0," Zenodo, April 2026. DOI: 10.5281/zenodo.18582492
- [2] R. L. H. Viana, "Domain Driven-Centric Router (DDCR) — v3.0," Zenodo, 2026. DOI: 10.5281/zenodo.18864832
- [3] R. L. H. Viana, "Orchestration Domain-Centric Pattern (ODCP) — v2.0," Zenodo, 2026. DOI: 10.5281/zenodo.18876593
- [4] R. L. H. Viana, "Event Domain-Centric Pattern (EDCP) — v1.0," Zenodo, 2026. DOI: 10.5281/zenodo.19068766
- [5] R. L. H. Viana, "Semantic Domain Integration Architecture (SDIA) — v2.0," Zenodo, March 2026. DOI: 10.5281/zenodo.18877635
- [6] R. L. H. Viana, "Domain Enterprise Integration Pattern (DEIP) — v2.0," Zenodo, 2026. DOI: 10.5281/zenodo.19004802
- [7] E. Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software. Addison-Wesley, 2003.
- [8] G. Hohpe and B. Woolf, Enterprise Integration Patterns. Addison-Wesley, 2003.
- [9] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, UC Irvine, 2000.
- [10] O. Zimmermann et al., Patterns for API Design. Addison-Wesley, 2022.
- [11] D. Crocker and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," RFC 5234, IETF, January 2008.
- [12] Netflix Technology Blog, "Announcing Zuul: Edge Service in the Cloud," 2013.
- [13] R. L. H. Viana, SAP Press e-Bite — Enterprise Messaging. SAP Press, 2021.

Appendix A — Historical Context: PDCP

Patterns originated Feb 2026 on SAP SCN (two blog posts → GDCR + ODCP). Archived Feb 6, 2026 on Internet Archive — prior art under 35 U.S.C. § 102.

A.1 PDCP — Package Domain-Centric Pattern (February 2026)

The second blog (PDCP) addressed package sprawl: iFlows proliferating per vendor, version, environment. Proposed one package per domain process with metadata-governed sub-flows. PDCP was later renamed ODCP — Layer 3 of SDIA.

Archived (Internet Archive — Whitepaper Machine, 6 Feb 2026):

<https://web.archive.org/web/20260206123644/https://community.sap.com/t5/technology-blog-posts-by-members/sap-btp-cpi-package-domain-centric-pattern-pdcp-solving-package-sprawl-at/ba-p/14318864>

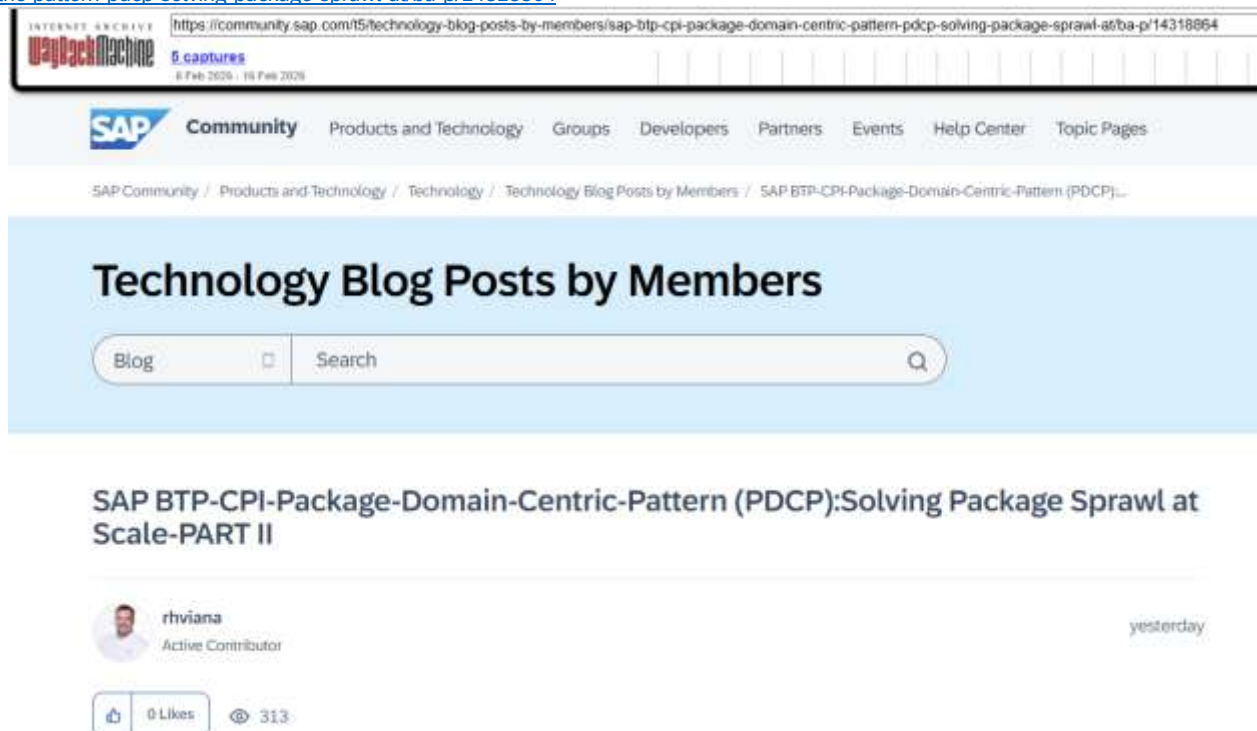


Figure A.2 — PDCP original publication, SAP Community, February 2026

A.2 From Platform-Scoped Patterns to the SDIA Ecosystem

Between Feb and Apr 2026, DCRP and PDCP were extended from SAP to multiple platforms. The core invariants — domain-as-primary-key, metadata-driven resolution, immutable artifacts — held across vendors. This produced the five-layer SDIA ecosystem.

Layer	Current Name	Evolution
1 — Gateway	GDCR	Generalization of DCRP beyond SAP APIM to all enterprise gateway platforms
2 — Resolution	DDCR	The runtime resolution engine extracted from DCRP and formalized as an independent specification
3 — Orchestration	ODCP	Generalization of PDCP beyond SAP CPI to all enterprise orchestration platforms
4 — Events	EDCP	New pattern extending the domain-invariant principle to event channels, topics, and messaging
5 — Data	DDCP	New pattern extending the domain-invariant principle to data persistence contracts

DEIP governs platform binding — which gateway a domain bind to — before any specification layer.

Archived SAP SCN publications (Feb 6, 2026) serve as dated prior art, preceding Zenodo publications by ~2 months. Independently verifiable via Wayback Machine.

Appendix B — Glossary

Definitions of terms used normatively throughout this specification. Terms are listed alphabetically.

Term	Definition
ABNF	Augmented Backus-Naur Form — formal grammar notation (RFC 5234)
Action Code	Single-character canonical code (c, r, u, d, s, n, a, t, e, b, v, w, x, z, f) normalized from 247 raw variants
Certificate Sprawl	Condition where certificates, trust stores, and security material are duplicated per project or vendor without domain governance
Company	Organizational tenant prefix in Package DNA (e.g., acme)
Credential Blackout	Operational failure where a single credential rotation cascades across multiple unrelated business processes
Credential DNA	Domain-scoped technical user name following cr.domain.subdomain.permission pattern
Data Store	Persistent storage for orchestration state (e.g., ds.sales.order.pending)
Division	Optional granularity segment representing business division (e.g., consumer, b2b)
Domain	Business domain token (e.g., sales, finance, logistics). Primary key of all ODCP artifacts
Domain Authority	Designated governance role responsible for approving artifact names within a domain
Flow DNA	Deterministic integration flow name following id##.subdomain.sender.entity.receiver.actioncode.direction.mode pattern
iFlow	SAP CPI term for integration flow. Equivalent to Flow (MuleSoft), Process (Boomi), Workflow (Azure)
Message Mapping	Transformation artifact between sender and receiver (e.g., mm.sales.order.sfdc.s4hana)
Package DNA	Domain-scoped container name following company.domain.subdomain.integrations pattern
Package Sprawl	Condition where orchestration packages multiply per vendor, project, or team without domain consolidation
Permission	Credential access level: rw (read-write), ro (read-only), rw-batch, ro-stream

Term	Definition
PVRL	Probabilistic Vocabulary Restriction Language — unified grammar family shared across SDIA ecosystem. Level 3 governs ODCP
Qualifier	Optional segment appended to artifact names (sector, division, region)
Region	Optional granularity segment representing geography (e.g., emea, apac, us)
Script Collection	Reusable script artifact grouped by domain and purpose (e.g., sc.sales.order.transformation)
Sector	Optional granularity segment representing industry sector (e.g., retail, corporate)
Subdomain	Operational subdivision within a domain (e.g., o2c, r2r, payroll)
Supporting Object Sprawl	Condition where value mappings, scripts, data stores, and variables are named ad hoc without domain alignment
Technical User Blackout Risk	Operational risk where a single credential rotation affects multiple unrelated processes
Value Mapping	Domain-scoped lookup table (e.g., vm.sales.order.status)