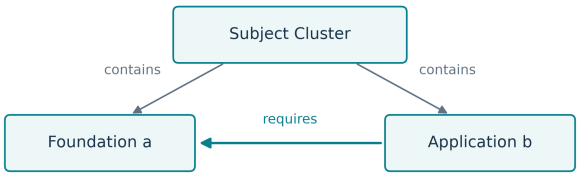
SKILLPILOT · TECHNICAL REFERENCE

# The SkillPilot Graph Definition

Structure, Mathematics, Recursive Learning Units, and Implicit Mastery




Grouping, prerequisite, and completion evidence are distinct claims.

**Scope and Status of This Edition**

Complete English reference, as of **11 November 2026**. SkillPilot models learning goals and their relationships: `contains` groups goals into a hierarchy, while `requires` identifies subject-matter prerequisites. Canonical goal identities are preserved across the canonical inventory, the Personal Curriculum, the active focus, and stored learner outcomes. **Learning units** (*Lerneinheiten*) bind a learning promise to directly owned goals and child units; optional schedules allocate their acquisition to available calendar periods and workload budgets. Explicitly verified understanding and independent application demonstrated through valid local assessments both count toward the effective mastered set.

**Normative force and implementation.** Atomic normal form $R \subseteq A \times A$ and the ordered migration path toward it are binding. Appendix A maps every rule to its validation area, makes actual hard enforcement visible via traffic-light (*Ampel*) status, and delineates existing CI gates. Global application recognition is **not yet implemented** on audited `main`; Appendix B documents how current runtime behavior differs. Completion of a learning unit is derived from its own obligations and required child units without persisting an additional mastery tier.

SkillPilot · enpasos – Enterprise Patterns & Solutions GmbH

Technical specification and build tooling: Apache-2.0.

Didactic illustrations: project content license CC BY 4.0; see Appendix.

## Reader's Guide and Scope

A SkillPilot skill graph describes *what* can be learned and which prerequisites govern it. Four further claims build on that foundation while remaining deliberately separate in the model:

- **Curricular views** determine which of these stable goals apply to a learner and where they sit within an educational program.
- **Learner state** records for each atomic goal whether it has been achieved.
- **Learning units** explain subject-matter coherence from a manageable topic up to a broader domain. Their completion is derived recursively from child units and directly owned obligations rather than persisted as a flat flag.
- **Planning** distributes acquisition and practice across available time.

Deriving implicit mastery connects explicit outcomes, transitive prerequisites, and subject-reviewed local contracts of learning units. All of these claims may reference one another, but none may silently substitute for another.

This reference is intended for joint review and further development of the model. It covers the graph definition in the broad sense: identity, edges, inheritance, validity, availability, projection, goal types, progress, scheduling, completion, provenance, and quality assurance. Complete syllabus texts and concrete task banks are not reproduced here. Worked examples serve as illustrations rather than approved curriculum subgraphs.

### The Model in Seven Steps

These seven steps preview the reasoning of the document. Concrete English goal names come from the Mechanics example in the original illustrations, starting in Figure 1.

1. **Goals and two relation kinds.** Every learning goal carries a stable identity. `contains` groups goals: *Kinematics* contains *Velocity*. `requires` states subject prerequisites: *Velocity* requires *Vectors*. Grouping alone never establishes a prerequisite (Sections 1 and 2).
2. **Atoms and clusters.** Goals without children are atomic; learner outcomes are stored only on them. Clusters group goals. In the canonical target state, every authored requirement connects an atom to an atom, and none is redundant. Legacy requirements on or toward clusters remain readable as transition data (Sections 2 and 3).
3. **Learner state and frontier.** Each atomic goal has exactly one stored binary result. An open atom becomes available next – in the *frontier* – only once all atoms it requires directly or across prerequisite chains are mastered (Section 4).
4. **Scoped views.** The Personal Curriculum fixes a learner's longer-term scope, while focus selects a temporary working subset. Derivations always run on the full graph first and restrict afterwards. Normal (*optimistic*) mode checks prerequisites inside the scope; strict mode checks all prerequisites globally (Sections 5 to 7).
5. **Goal types.** A goal's pedagogical type defines what counts as achieving it: verified understanding for a content goal, completed motivational entry for an orientation goal, verified recall for a memory goal, and a passed exam for an assessment goal (Section 8).
6. **Learning units.** A learning unit binds a coherent learning promise to its own goals and required child units. Its completion is derived from those obligations rather than stored separately (Section 11).
7. **Implicit mastery.** Once no open assessment and no open memory goal of a learning unit still depends on one of its own content goals, that content goal counts as independently applied – provided the unit's local contract has passed subject review and is valid. Explicitly verified

understanding and application demonstrated in this way together form the effective mastered set; both satisfy later prerequisites and count toward progress (Section 13).

### Structure, Status, and Rule Identifiers

Each part answers a guiding question and carries its own status. The final column lists the prefix under which Appendix A tracks that part's rules, such as `GD-C07` for a formal-core rule.

| Sections | Guiding question | Status | Rules |
|---|---|---|---|
| 1–5: Formal Core | What can be learned, what does it require, and what is available next – across the full graph and inside a scoped view? | Normative core | `GD-C` |
| 6–10: Context, Views, Goal Types, Quality | Where does a goal sit in an educational program, what belongs to personal scope, when does a goal count as achieved, and what is its provenance? | Adjacent contract | `GD-X` |
| 11: Learning Units | Which goals form a coherent subject-matter whole, and when is it complete? | Domain structure | `GD-U` |
| 12: Acquisition and Time Planning | Can acquisition be arranged into a valid order and calendar budget? | Optional profile | `GD-P` |
| 13: Implicit Mastery | When does a passed assessment also substantiate the goals it builds upon? | Binding concept, not yet implemented | `GD-A` |
| Appendices A–F | How is each rule checked, where does the implementation diverge, and what do terms and symbols mean? | Reference | `GD-D` |

| Status | Meaning in this reference |
|---|---|
| Normative core | Binding mathematical graph semantics from `graph-definition.md`; additional validator profiles may impose stricter constraints. |
| Adjacent contract | Semantics of projection, goal types, provenance, or learning workflows; actual implementation coverage must be verified per rule. |
| Domain structure | Recursive learning units with a learning promise, directly owned goals, and derived completion; not a claimed runtime feature today. |
| Optional profile | Calendar and workload budgeting plus conditional entry to learning units; hard technical locks and direct assessment entry remain undecided there. |
| Not yet implemented | The new implicit-mastery recognition contract is bindingly specified, but not yet deployed; current runtime code follows an earlier demand model (Appendix B). |

Curriculum authors primarily need Sections 1 to 3, 8, and 10. Runtime logic primarily rests on Sections 4, 5, 7, and 13; quality assurance and review additionally rely on Appendices A to C. Sections 11 and 12 address everyone who structures and schedules courses.

### Reading Conventions

**Normative force.** "Must", "must not", and "binding" denote normative model rules. "Not yet implemented" means bindingly specified, but not yet realized in the runtime. "Optional" marks an additional profile; "open" or "not yet decided" marks a deliberately unresolved question.

**Notation.** Italic uppercase letters denote sets and relations ($G$, $A$, $R$, $M$); italic lowercase letters denote arbitrary goals ($g$, $p$, $a$). Concrete example nodes from a figure are set upright in

sans-serif to match the figure labels: in the chain Q → B → A, A is one individual goal rather than the set $A$ of atomic goals. Spelled-out example names such as *Velocity* appear in italics. Arrows always point in the direction of the relation, from cluster to child and from goal to prerequisite. Monospace denotes field and identifier names such as `requires`. Figures 1 to 9 preserve the original English-labelled illustrations of the core specification; their captions explain every label.

**Terminology.** The distinctions below are easily conflated and are therefore used throughout this document with these exact meanings. Appendix D provides the complete glossary and symbol index.

| Terms | Meaning in this specification |
|---|---|
| Requirement, prerequisite | The edge $(g, p) \in R$ is a *requirement* of $g$; goal $p$ is a *prerequisite* of $g$. |
| authored, derived | Only the stored `contains` and `requires` edges are *authored*. Inherited, effective, and atomically normalized requirements as well as closures are *derived* from them. |
| target state, transitional data | *Target state* denotes the intended state of canonical data: atomic normal form and transitive minimality. *Transitional data* denotes current data that still contains cluster requirements. Neither term ever refers to the mastery state of an individual goal. |
| explicit mastery, implicit mastery, effectively mastered | *Explicit mastery* (*Mastery*) is the stored binary outcome (set $M$). *Implicit mastery* is derived (set $I_U$). *Effectively mastered* is the union of both (set $M_{\text{eff}}$). |
| content goal, protected goal | *Content goal* denotes an ordinary subject-matter learning goal; figures label it "ordinary". *Protected* goals are assessment and memory goals: they are never implicitly passed. |
| cluster, learning unit, program unit | A *cluster* is a goal with `contains` children. A *learning unit* (*Lerneinheit*) is an explicitly authorized educational unit with a stated learning outcome. A *program unit* is an organizational section such as a school year or semester. |
| level, stage | Level of personalization (Level 1–4), curriculum QA maturity level (M0–M7), planning period of a schedule, school stage, and the difficulty field `level` are five distinct concepts. |

## Sources and Limits of Claims

The domain contracts in the repository remain the authoritative primary sources. Appendix E maps every chapter to those sources. Appendix B compiles the dated Product Owner decisions cited in the text and documents the implementation divergences audited when preparing this edition. Mathematical examples, figure captions, and plausibility checks replace neither subject-matter task reviews nor empirical studies with learners.

## Contents

**Reader's Guide and Scope** 2

**1 Goals, Identity, and Relations** 8
1.1 The Base Object . . . 8
1.2 Attributes and Stable Identity . . . 8
1.3 Atomic Goals and Clusters . . . 10

**2 Containment, Prerequisites, and Inheritance** 10
2.1 Hierarchy and Atomic Basis . . . 11
2.2 Direct Requirements Are Conjunctive . . . 12
2.3 Effective Requirements . . . 13

**3 Validity, Minimality, and Learnability** 15
3.1 Mandatory Conditions . . . 15
3.2 Additional Profiles and Counterexamples . . . 18
3.3 Didactic Routes from Orientation to Autonomy . . . 19

**4 Learner State, Satisfaction, and Progress** 19
4.1 Satisfaction Without Backward Prerequisite Inference . . . 20
4.2 Global Frontier . . . 21
4.3 Aggregation Without Double Counting . . . 21

**5 Filters, Focus, and Global Diagnostics** 22
5.1 Filters as Predicates and Compiled Applicability . . . 22
5.2 Inherit Globally First, Restrict Afterwards . . . 23
5.3 Normal Mode and Strict Mode . . . 24
5.4 Missing Prerequisites and Missing Atoms . . . 25

**6 Stable Goals, Contextual Structure** 27
6.1 Identity Across Different Placements . . . 27
6.2 Placements and Projections . . . 27
6.3 Reviewed Composition Views . . . 28

**7 Personal Scope, Focus, and Visible Goals** 29
7.1 Four Levels of Personalization . . . 29
7.2 Target Role and Prerequisite Support . . . 30
7.3 Applicability Is Not Placement . . . 31
7.4 Switching Focus Without Implicit Mastery . . . 31

**8 Goal Types and the Meaning of an Outcome** 32
8.1 Understanding and Application . . . 32
8.2 Orientation as a Distinct Completion Contract . . . 33
8.3 Memorization: Practice and Verified Recall . . . 33
8.4 Assessment Goals in the Current Workflow . . . 34

**9 Time Planning as a Complementary Overlay** 35

**10 Provenance, Resources, and Defensible Quality Claims** 36
10.1 Three Kinds of Links . . . 36
10.2 Source Rationales for Goals and Relations . . . 36
10.3 Algorithmic Validity and Domain Quality . . . 36

10.4 Publication and Interchange . . . 37
10.5 Licensing and Private Learner Data . . . 37

**11 Learning Units: From Learning Promise to Recursive Completion 39**
11.1 Motivation and Distinct Domain Claims . . . 39
11.2 Direct Acquisition, Child Units, and Total Scope . . . 40
11.3 Completion Without Stored Unit Mastery . . . 41
11.4 Recognition Before Full Unit Completion . . . 41
11.5 Prerequisites and the Limits of a Unit Sequence . . . 41

**12 Optional Acquisition and Time Planning: Sequence and Budget 44**
12.1 Plan Scope, Entry Basis, and Single Initial Acquisition . . . 44
12.2 Structural Feasibility Through a Completion Witness . . . 44
12.3 Effort, Reserve, and Budget . . . 46
12.4 Validity Claims and Completion Boundaries . . . 46

**13 Implicit Mastery Through Valid Local Assessments 48**
13.1 Local Contract, Target Scope, and Protected Goal Types . . . 48
13.2 Transitive Open Demand and Implicit Set . . . 49
13.3 Why the Skill Graph Alone Does Not Prove Application . . . 50
13.4 Global Effectiveness and Durable Contract Binding . . . 51
13.5 Protected Goals and Full Unit Completion . . . 52
13.6 Cumulative Attainment and Dated Period Events . . . 54

**A Rules and Milestones 56**
A.1 Traffic-Light Status: What Is Truly Hard-Enforced Today . . . 57
A.2 Overview of Validation Areas . . . 58
A.3 Gate Reference and Actual Scope . . . 59
A.4 Assignment Key . . . 60
A.5 Actual Promotion Chain and Additional Gates . . . 62
A.6 C – Formal Core . . . 63
A.7 X – Context, Views, Goal Types, Provenance, and Quality . . . 68
A.8 U – Recursive Learning Units . . . 75
A.9 P – Optional Acquisition and Time Planning . . . 78
A.10 A – Implicit Mastery via Local Assessment Contracts . . . 79
A.11 D – Document Appendices: Status, Verification Strategy, Sources, and Sign-Off . 83

**B Implementation Status and Boundaries of Claims 85**

**C Verification Strategy for Completion Semantics 89**

**D Glossary and Symbols 90**
D.1 Terms . . . 90
D.2 Symbols . . . 92

**E Normative Source Mapping and Illustrations 94**

**F Building and Verifying the Specification 96**

## List of Figures

1 Goals and two relation kinds . . . 9

2 Direct and indirect containment . . . . . 11
3 Direct prerequisites . . . . . 12
4 Prerequisites inherited from a cluster . . . . . 13
5 Atomic expansion before transitive closure . . . . . 14
6 Effective cycles created by inheritance . . . . . 16
7 Transition diagnostic: inherited redundancy . . . . . 17
8 Available next goals (frontier) . . . . . 22
9 Optimistic (normal) mode and strict mode . . . . . 25
10 Mechanics as a recursive learning unit . . . . . 42
11 Optional calendar-based planning periods . . . . . 45
12 Transitive demand across an explicitly mastered intermediate goal . . . . . 50
13 Local recognition remains globally effective for later learning units . . . . . 52
14 Protected memory goals and external support . . . . . 53
15 Two local assessments sharing a common foundation . . . . . 54
16 Total plan progress and dated period completions . . . . . 55

# 1 Goals, Identity, and Relations

Sections 1 to 5 form the formal core. This section introduces learning goals and the two authored relation kinds. Section 2 derives the relations used in computation, while Section 3 states the graph validity conditions. Section 4 models a learner's state and Section 5 evaluates availability and diagnostics inside a scoped subset of the graph.

## 1.1 The Base Object

The normative core is a finite directed graph

$$\mathcal{G} = (G, C, R), \qquad C, R \subseteq G \times G.$$

$G$ is the set of learning goals. $C$ denotes `contains`: one goal contains another. $R$ denotes `requires`: one goal requires another as a prerequisite. Exactly these two *explicitly authored* relation kinds exist in the skill graph; only they are stored by curriculum authors. Derived relations, curricular placements, source mappings, and evidence belong to separate model layers.

Each pair $(x, y)$ is directed. For any relation $X$ on $G$,

$$X^{-1} = \{(y, x) \mid (x, y) \in X\}$$

denotes the inverse relation and $X^+$ the transitive closure: $(x, y) \in X^+$ holds if and only if a directed $X$-path of at least one edge leads from $x$ to $y$. Zero-length paths do not count. A relation $X$ is *acyclic* if no goal leads back to itself along such a path, that is, $\forall g \in G : (g, g) \notin X^+$. A directed acyclic graph is called a DAG for short. The raw union $C \cup R$ is *not* the relation against which DAG conditions are checked indiscriminately: the two edge kinds carry distinct semantics. Section 3 specifies which relations must be acyclic.

Figure 1 illustrates both relation kinds on an example.

**Arrow Direction Is Part of the Definition**

$k \xrightarrow{\texttt{contains}} c$: cluster $k$ contains goal $c$. $g \xrightarrow{\texttt{requires}} p$: goal $g$ requires prerequisite $p$. A learning-flow view instead depicts $R^{-1}$: from prerequisite to the skill building upon it. That inverse direction must be explicitly labelled.

## 1.2 Attributes and Stable Identity

Every goal carries three normative base attributes. They are total mappings, defined for every goal:

$$\mathrm{Id} : G \to \mathrm{UUID}, \qquad \mathrm{Title} : G \to \Sigma^*, \qquad \mathrm{Weight} : G \to \mathbb{R}_{>0}.$$

Id is injective: distinct goals have distinct IDs. The UUID remains the canonical identity across placements, locale files, and views. Single-letter identifiers in this document are mathematical variables only, not statements about the serialized ID format.

Two further attributes are partial mappings ($\rightharpoonup$) that need not be defined on every goal. An optional $\mathrm{ShortKey} : G \rightharpoonup \Sigma^*$ is unique within a logical skill landscape, that is, across the shared

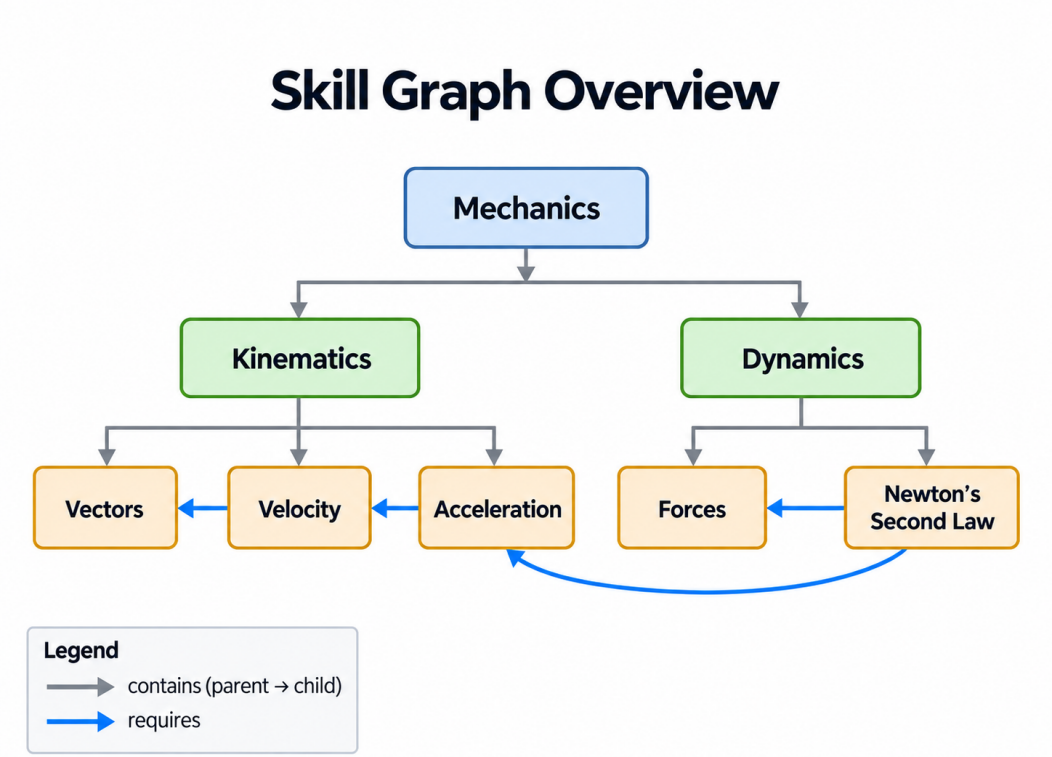



**Figure 1:** Two relations in one skill graph. Grey arrows group children (`contains`); blue arrows point to prerequisites (`requires`). *Velocity* requires *Vectors*; *Newton's Second Law* requires both *Forces* and *Acceleration*. Node colours here do not represent learner states. This Mechanics example recurs across the original illustrations up to Figure 9.

goal inventory of one curriculum. Multiple locale files for the same goal may repeat the same pair of ID and ShortKey. By contrast, the partial mapping PhaseCompat : $G \rightharpoonup P_{\text{compat}}$ captures compatibility or display metadata, often serialized under `phase`; it defines neither canonical identity nor didactic learning order. Table 1 situates other common fields.

**Table 1:** Supplementary goal attributes and their bounded semantics.

| Field | Domain meaning outside the minimal core |
|---|---|
| `description` | Clear, verifiable goal statement, typically "The learner can …". Orientation goals follow their own semantics (Section 8). |
| `area`, `tags` | Subject classification and metadata; a tag never replaces an authoritative decision on goal type or scope. |
| `level` | Difficulty or cognitive demand tier; neither a subject learning unit nor a personalization level. |
| `core` | Core goal versus extension within the given contract; an extension may still be a mandatory target in a selected course profile. |
| `weight` | Share in progress aggregation; not a time duration, not subject-evidence strength, and not a pass probability. |
| `examples` | References to worked examples or tasks; mere presence proves neither task quality nor competence mastery. |
| `resourceLinks` | Existing reference field used for SkillPilot-owned visualizations, official sources, and legacy links. Under the current contract, new optional external study materials belong in separate content packages. |

### 1.3 Atomic Goals and Clusters

The *structural* classification follows solely from $C$:

$$A = \{g \in G \mid \nexists c \in G : (g, c) \in C\}, \qquad K = G \setminus A.$$

A goal without `contains` children is *atomic* (an atom, a leaf); every other goal is a *cluster*. A goal may have multiple parents: the canonical `contains` graph may form a polyhierarchy. Any explicit classification metadata stored alongside nodes must agree with this derived split.

Structural atomicity does not automatically imply *semantic* atomicity. A leaf bundling five independent sub-competencies remains a graph leaf formally, yet fails semantic quality review.

Alongside its structural kind, every atomic goal has a pedagogical *goal type*. The default is the *content goal*; orientation, memorization, and assessments form dedicated pedagogical kinds:

- A **content goal**, the ordinary subject-matter learning goal, describes conceptual understanding or procedural skill.
- An **orientation goal** describes a motivational entry point.
- A **memory goal** describes verified recall of required cards.
- An **assessment goal** describes a graded exam task with its own passing criterion.

Section 8 specifies when a goal of each type counts as achieved. These kinds are neither additional structural node classes nor program units.

## 2 Containment, Prerequisites, and Inheritance

The two authored relation kinds induce the derived relations used to compute availability, progress, and completion. Table 2 summarizes them; the subsections below define each entry in order.

**Table 2:** Authored and derived relations. In the canonical target state $R = R_A = R_{\text{eff}} = R_{\text{at}}$; only the transitive closure $R_{\text{at}}^+$ remains derived.

| Symbol | Name | Construction and use |
|---|---|---|
| $C$ | Containment | Authored `contains` edges. They group goals and determine atoms, clusters, and each goal's atomic basis. |
| $R$ | Authored (direct) requirements | Stored `requires` edges. In the canonical target state they connect atoms only. |
| $R_{\text{eff}}$ | Effective requirements | $R$ together with requirements inherited from `contains` ancestors. |
| $R_{\text{at}}$ | Atomically normalized requirements | Every effective prerequisite is expanded into its atomic basis, leaving pairs of atoms only. |
| $R_{\text{at}}^+$ | Transitive closure | All prerequisites reachable across chains. Basis of demand, frontier, and implicit mastery. |
| $R_A$ | Directly authored atomic requirements | $R \cap (A \times A)$; basis of the didactic route check in Section 3.3. |

### 2.1 Hierarchy and Atomic Basis

For the finite acyclic containment graph, define

$$\begin{aligned}\mathrm{Anc}(g) &= \{k \in G \mid (k, g) \in C^+\},\\ \mathrm{Desc}(g) &= \{d \in G \mid (g, d) \in C^+\},\\ \mathrm{Atoms}(g) &= \begin{cases}\{g\}, & g \in A,\\ \mathrm{Desc}(g) \cap A, & g \in K.\end{cases}\end{aligned}$$

$\mathrm{Anc}(g)$ contains the ancestors of $g$, $\mathrm{Desc}(g)$ its descendants, and $\mathrm{Atoms}(g)$ its *atomic basis*. Every ancestor has children and is therefore a cluster. The short forms Anc and Desc correspond to `Ancestors` and `Descendants` in the original illustrations.

Every cluster possesses at least one atomic leaf: in a finite acyclic graph, every descending path terminates after finitely many steps at a goal without children. Consequently, $\mathrm{Atoms}(g)$ is non-empty for every goal. A node without children is atomic rather than an empty cluster. Across multiple parent paths, each atom is counted once in the set. Figure 2 illustrates direct and indirect containment.

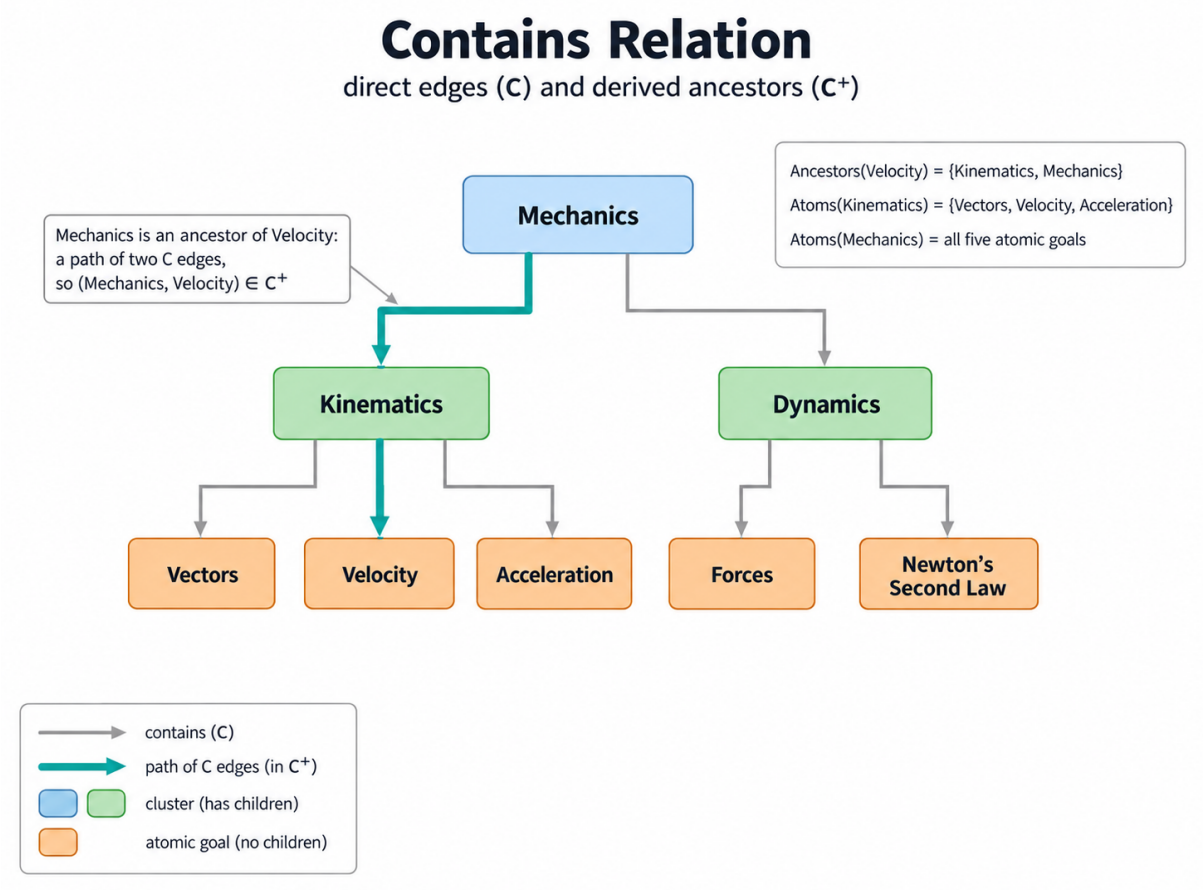



**Figure 2:** Direct and indirect containment. The teal-highlighted path consists of two existing `contains` edges rather than a third relation kind. *Mechanics* is therefore an ancestor of *Velocity*.

### 2.2 Direct Requirements Are Conjunctive

$(g, p) \in R$ means that $g$ directly requires $p$. The edge $(g, p)$ is called a *requirement* of $g$, and the goal $p$ is a *prerequisite* of $g$. Multiple outgoing edges are conjunctive: $g$ requires *all* listed prerequisites, not a choice among them. Alternatives require a dedicated selection contract outside this relation. The relation $R$ must be acyclic. Figure 3 shows four direct requirements.

Under the binding decision of 8 November 2026, the canonical target state satisfies

$$R \subseteq A \times A. \tag{1}$$

This property is called *atomic normal form*: *every* authored requirement has an atomic source goal and an atomic prerequisite goal. Even a universal cluster claim intended to apply to every descendant of a cluster has no permanent exception. Clusters serve hierarchical grouping and derived dependency summaries.

Data that still contain requirements outgoing from or pointing to clusters are called *transition data* in this document. Existing cluster requirements retain their transition reading semantics: a requirement authored on a cluster applies to all descendants through inheritance, and a cluster prerequisite expands into its atomic basis (Section 2.3). Transition compatibility does not satisfy the canonical target norm.

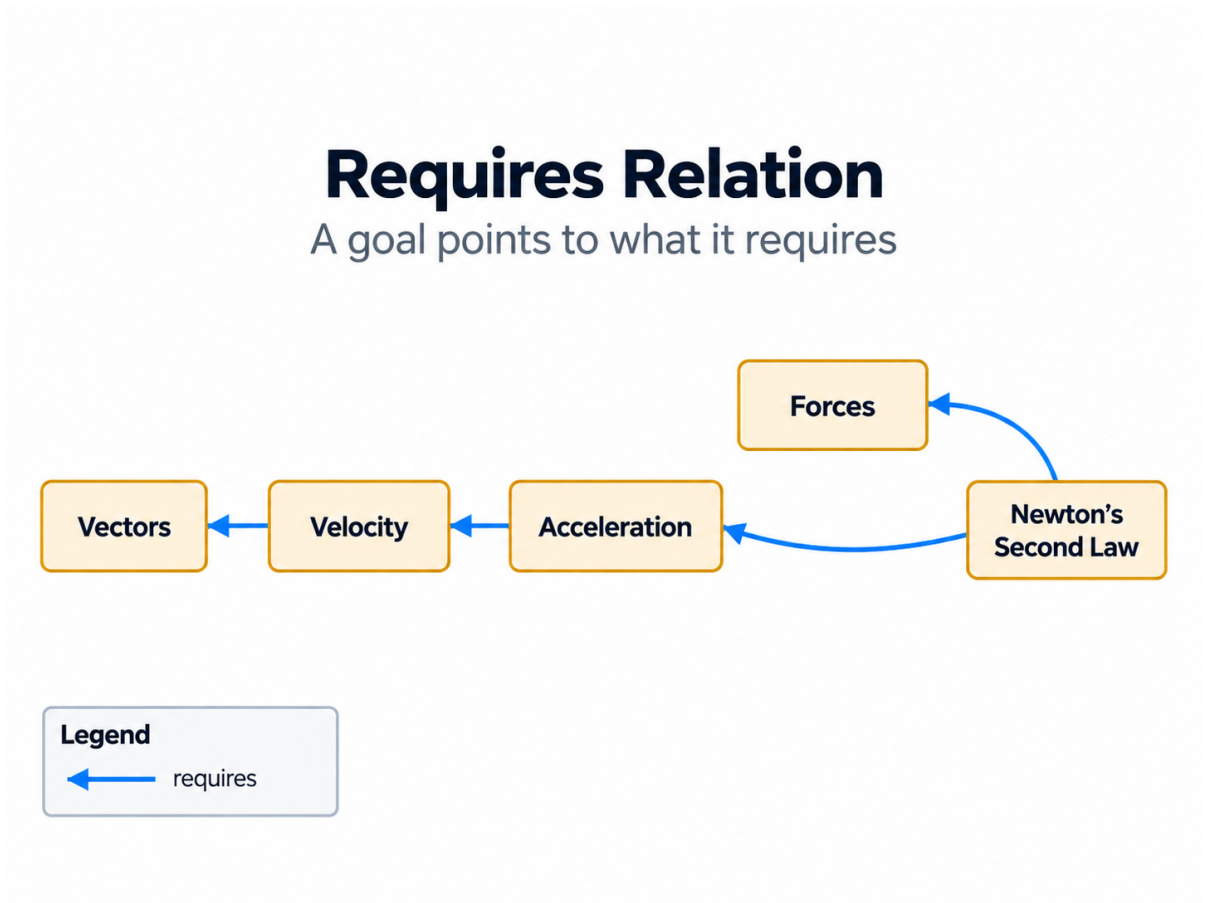



**Figure 3:** Direct prerequisites. The curved arrow is a direct edge just like the straight arrows. The chain from *Acceleration* via *Velocity* to *Vectors* yields transitive reachability, not containment inheritance.

### 2.3 Effective Requirements

So that transition data are also read unambiguously, three ordered steps prepare authored requirements: inherit first, expand to atoms second, and take the transitive closure third. This exact sequence applies wherever this document evaluates prerequisites.

**Step 1: Inherit.** The effective relation also reads legacy cluster requirements. It combines a goal's own direct requirements with those declared on its `contains` ancestors:

$$(g,p) \in R_{\text{eff}} \iff (g,p) \in R \ \vee \ \exists k \in \text{Anc}(g) : (k,p) \in R. \tag{2}$$

Here

$$R_{\text{inherited}} = R_{\text{eff}} \setminus R, \qquad R_{\text{eff}} = R \cup R_{\text{inherited}}, \qquad \text{Pre}_{\text{eff}}(g) = \{p \mid (g,p) \in R_{\text{eff}}\}.$$

Multiple parent paths combine requirements by set union without duplication. *Inheritance* in (2) and *transitive closure* $R^{+}_{\text{eff}}$ are distinct operations: inheritance propagates ancestor requirements downward along `contains` to descendants, whereas transitive closure follows chains of `requires` edges. Figure 4 illustrates an inherited requirement.

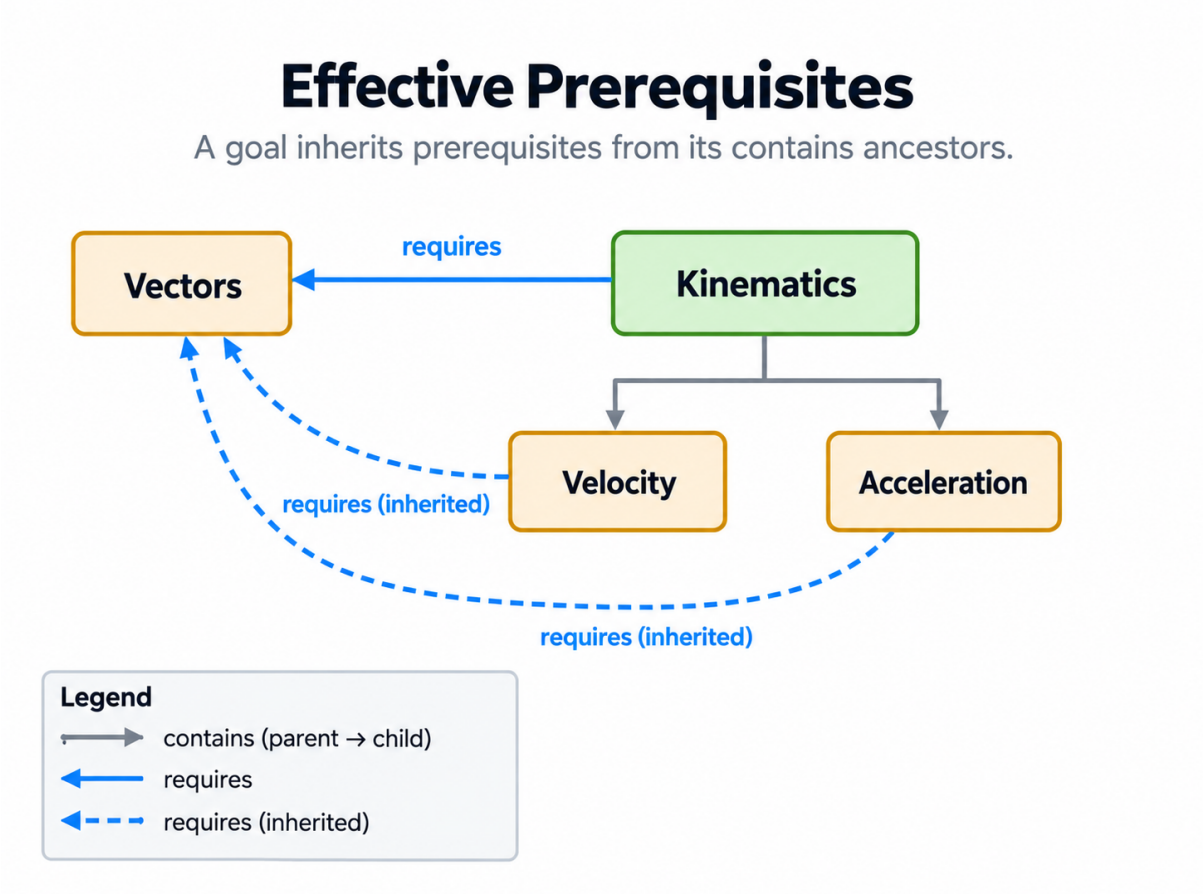



**Figure 4:** Transition-data example: a prerequisite inherited from a cluster. *Vectors* lies outside *Kinematics* in this example; if it were contained inside, it would inherit a self-requirement. Solid arrows are directly authored; dashed arrows are inherited.

**Step 2: Expand to atoms.** For runtime evaluation, prerequisites are normalized to atoms first. Every effective prerequisite is replaced by its atomic basis:

$$R_{\text{at}} = \{(a, b) \in A \times A \mid \exists p \in G : (a, p) \in R_{\text{eff}} \wedge b \in \text{Atoms}(p)\}.$$

**Step 3: Take the transitive closure.** The transitive closure $R_{\text{at}}^{+}$ therefore also captures prerequisite chains continuing beyond the expansion of a required cluster. Mastery of an intermediate goal never truncates this closure. Section 13 likewise retains $R_{\text{at}}$ for this normalized atomic relation while keeping $R$ for authored edges.

Figure 5 illustrates Steps 2 and 3 on a cluster requirement: external foundation c remains reachable via a once cluster K is expanded.

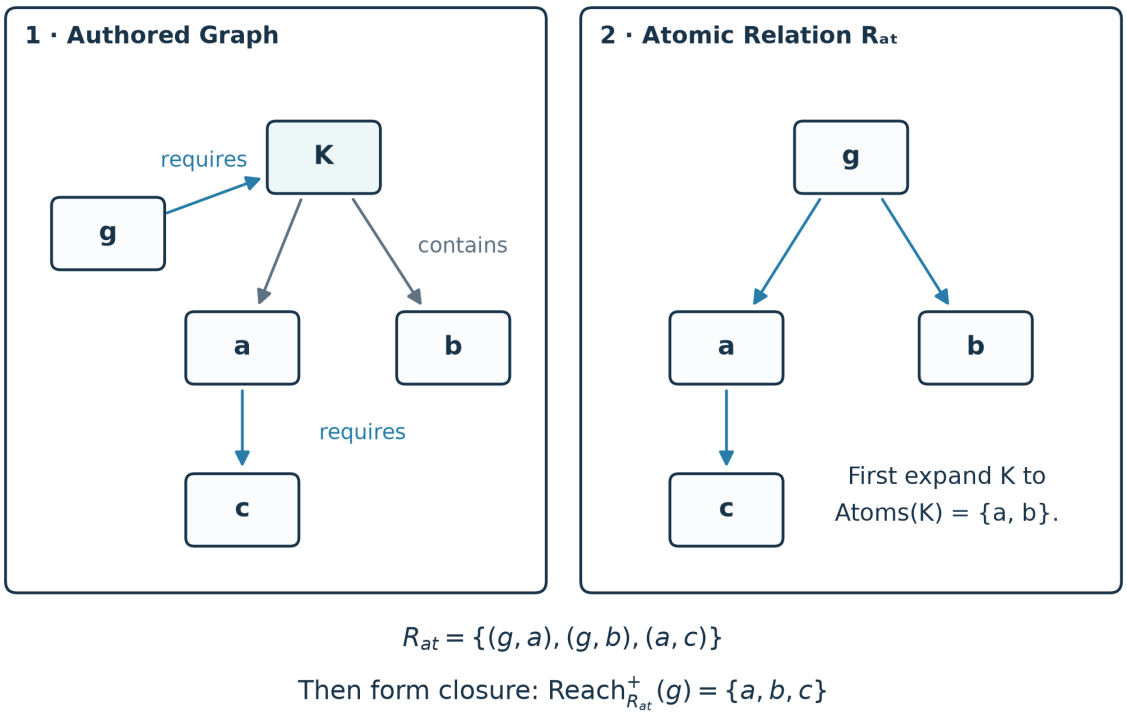



**Figure 5:** From $\mathsf{g} \to \mathsf{K}$ and $\text{Atoms}(\mathsf{K}) = \{\mathsf{a}, \mathsf{b}\}$, atomic expansion yields $(\mathsf{g}, \mathsf{a})$ and $(\mathsf{g}, \mathsf{b})$. Because $\mathsf{a} \to \mathsf{c}$ also holds, forming the transitive closure afterwards includes $(\mathsf{g}, \mathsf{c})$. Example node K denotes one cluster, not the set $K$ of all clusters. The right panel shows the derived atomic relation rather than newly authored edges.

**Atomic demand of a goal.** Several later formulas ask the same question: which atoms does a goal require altogether? For any goal $g$, define

$$\begin{aligned} B(g) &= \bigcup_{p \in \text{Pre}_{\text{eff}}(g)} \text{Atoms}(p), \\ \text{RequiredAtoms}(g) &= B(g) \cup \{b \in A \mid \exists a \in B(g) : (a, b) \in R_{\text{at}}^{+}\}. \end{aligned} \tag{3}$$

RequiredAtoms($g$) is called the *atomic demand* of $g$ (written $D(g)$ in earlier editions). For an atom $g$, RequiredAtoms($g$) $= \{b \in A \mid (g, b) \in R_{\text{at}}^{+}\}$. For a cluster, atomic demand captures its own and inherited prerequisite claims without treating its own `contains` children as additional prerequisites. Atomic demand depends solely on the graph, never on learner state. In Figure 5, RequiredAtoms(g) $= \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$.

In the fully atomic-authored target state, $R_{\text{eff}} = R = R_{\text{at}}$: `contains` ancestors are clusters and carry no outgoing requirements. Transitive chains remain relevant as graph structure.

## 3 Validity, Minimality, and Learnability

### 3.1 Mandatory Conditions

A complete skill graph satisfies the canonical target state if and only if, alongside the attribute and identity requirements of Section 1, atomic normal form (1), acyclicity, and atomic transitive minimality hold:

$$\forall g \in G : \ (g, g) \notin C^{+}, \quad (g, g) \notin R^{+}, \quad (g, g) \notin R_{\text{eff}}^{+}, \quad (g, g) \notin R_{\text{at}}^{+}, \tag{4}$$

$$\forall (a, b) \in R : \quad (a, b) \notin \big(R \setminus \{(a, b)\}\big)^{+}. \tag{5}$$

Condition (4) requires that none of the four relations leads any goal back to itself. Condition (5) requires that no authored requirement is already reachable via the remaining authored edges. All identities and references must resolve; unknown node IDs never count as satisfied prerequisites.

**On acyclicity.** The DAG condition on $R$ already follows from that on $R_{\text{eff}}$ because $R \subseteq R_{\text{eff}}$, yet it remains listed as a clear standalone authoring check. Acyclicity of $C$ remains required independently. Direct acyclicity alone is insufficient because inheritance can create an effective cycle (Figure 6). Likewise, a self-cycle exposed by atomic expansion remains a modeling error even when the unexpanded edges look acyclic.

**On transitive minimality.** In the atomic chain $\mathsf{Q} \to \mathsf{B} \to \mathsf{A}$, an extra direct edge $\mathsf{Q} \to \mathsf{A}$ is redundant and must be removed. The remaining closure still demands $\mathsf{A}$ even when $\mathsf{B}$ is already mastered. Transitive pairs are derived rather than stored as additional direct edges. The binding transitive-minimality rule of 4 November 2026 remains in force; the decision of 8 November 2026 fixes atomic normal form as the canonical target state together with its ordered migration path.

The internal check identifier `G17` denotes complete atomic normal form plus complete transitive minimality across the declared canonical subject inventory, including required cross-subject prerequisites and inheriting ancestors; a partial scan or a single route profile does not suffice.

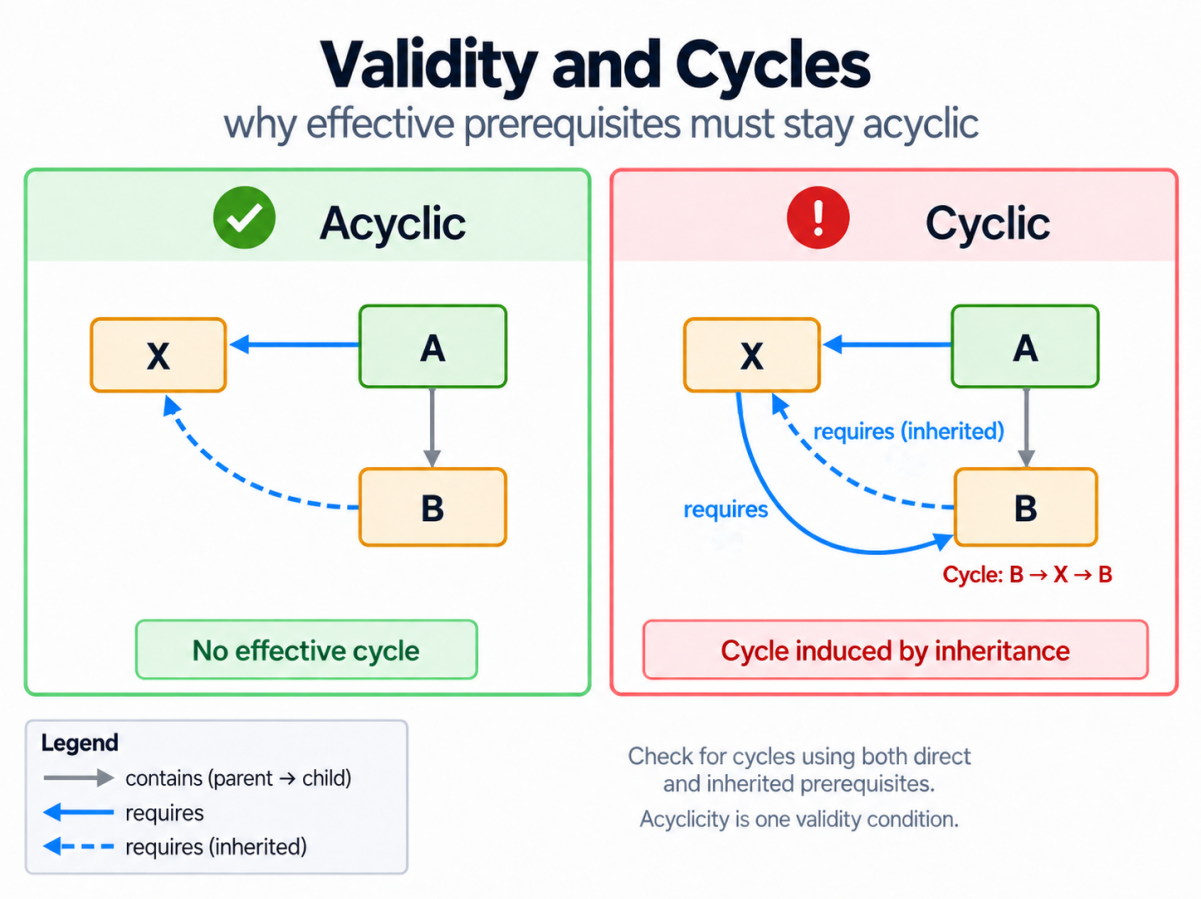



**Figure 6:** Direct acyclicity is not enough. Cluster A contains B and requires X, inducing the inherited edge B → X. Adding X → B creates the effective cycle B → X → B. Example node A here denotes a cluster, not the set $A$ of atomic goals.

**Binding migration order.** Moving from transition data to the canonical target state follows three steps in a fixed order. Keeping $C$ and stable goal identities unchanged:

1. **Establish atomic requirements.** First materialize or retain the complete reviewed atomic prerequisite relationships. Resolve existing meaning via $R_{\text{at}}$ from Section 2.3: inherit through `contains`, expand cluster prerequisites into atoms, and form the transitive closure. Covering full atomic demand does not mean authoring every transitive pair as a direct edge. Atomic edges must never be deleted in favour of their inherited cluster equivalent.
2. **Remove cluster requirements.** Remove any requirement with a non-atomic source or prerequisite goal only after proving that the authored atomic relationships cover its full atomic demand. No such cluster edges remain in the target state, including universal cluster claims.
3. **Reduce transitively.** Only afterwards transitively reduce the resulting atomic relation while preserving its full prerequisite closure.

Across the complete migration, the proof must establish in particular:

$$(R_{\text{at}}^{\text{before}})^{+} = (R_{\text{at}}^{\text{after}})^{+} = R_{\text{after}}^{+}, \qquad R_{\text{after}} \subseteq A \times A.$$

Thus the normalized atomic prerequisite closure stays invariant, and after migration the authored edges generate it directly. In addition, the relevant resolved-scope, applicability, target-set, frontier, completion, and mastery contracts must be preserved. Closure equality alone does not prove those further contracts, which include direct atomic routes and explicitly authored

projection roles. Raw prerequisite reachability between non-atomic goal nodes may change when cluster edges are removed. Migration neither invents subject dependencies or roles automatically nor writes learner outcomes. Contradictory legacy claims require dedicated subject review.

**Diagnostics on transition data are not deletion instructions.** Two legacy checks remain useful as diagnostics, yet neither authorizes deleting an atomic edge while a cluster requirement still supplies the same demand.

The former *local minimality condition* reads

$$(g, p) \in R \Longrightarrow \nexists k \in \mathrm{Anc}(g) : (k, p) \in R. \tag{6}$$

It flags direct duplicates of inherited requirements, but must never eliminate an atomic edge in favour of a cluster edge. Whenever an atomic goal $q$ and its cluster ancestor $k$ both require an atomic goal $p$, $q \to p$ is retained during atomarization; once complete atomic coverage is established, $k \to p$ is removed. In the target state, (6) holds automatically because clusters carry no outgoing requirements.

The *edge-removal test* formerly run under `G17` is likewise a diagnostic on transition data: for each authored edge $e = (g, p)$, it forms $R' = R \setminus \{e\}$, recomputes $R'_{\text{eff}}$ against unchanged $C$, and flags redundancy whenever $(g, p) \in (R'_{\text{eff}})^+$. Figure 7 illustrates both this mathematically sound diagnosis on transition data and why naive deletion fails: the alternative path still relies on a cluster requirement.

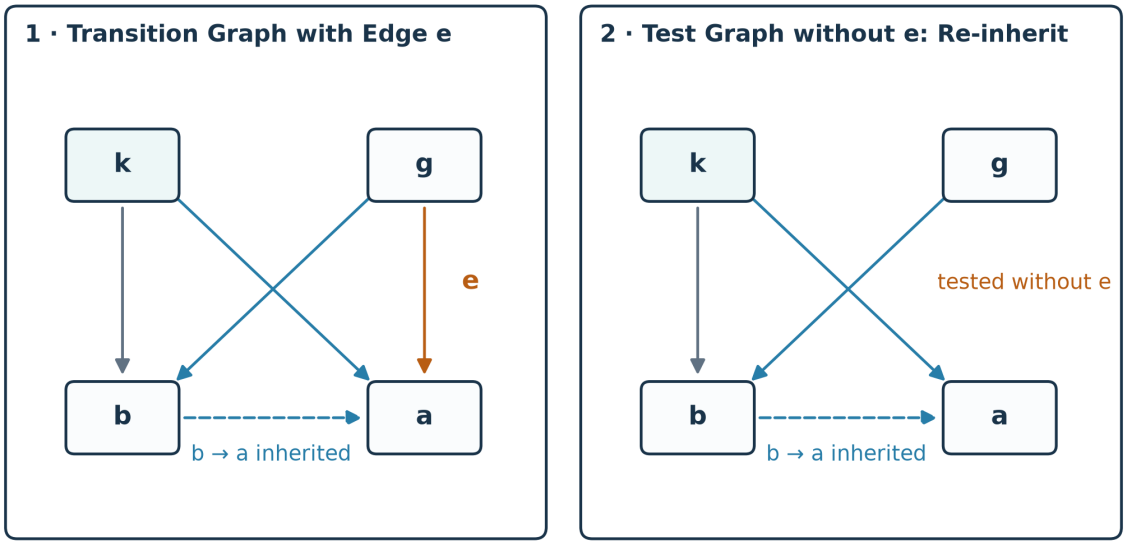



**Figure 7:** Cluster k contains b and requires a, so b inherits b → a. Removing the authored edge e = (g, a) and recomputing inheritance leaves the effective path g → b → a. This is a redundancy diagnostic on transition data, never an instruction to delete e in favour of k → a. First materialize the full atomic prerequisite coverage of k → a, then remove the cluster edge; only afterwards does a purely atomic replacement path decide whether e reduces. Only the inherited edge is dashed.

**What current checks establish.** Since 11 November 2026, `CQR-001` in the status generator runs the removal auditor on every declared root of the complete current canonical Gymnasium catalog, including the required cross-subject goals and their inheriting ancestors. Besides references and the effective DAG, it reports every authored requirement with a non-atomic source or prerequisite goal, every redundant authored requirement, and every contradicting explicit type. Once atomic normal form holds, the removal criterion coincides with (5); a pass therefore establishes `G17` for that root. A violation caps the subject at `M0` and thereby blocks every milestone pass `M1`–`M7`. The generator aborts when the catalog is incomplete or a canonical document deviates from the verified prerequisite-reduction proof registry. CI fails hard on a violation only through the protected maturity floors; other canonical subjects merely report `M0`, and `CQR-001` does not check roots outside the canonical Gymnasium catalog. A pass certifies the resulting state, not the migration order; delivered reductions are bound to the registry, which every run replays (Appendix A).

Two rule families currently enforce parts of the target state: *Curriculum Quality Review* (CQR) and *graph-validation rollout rules* (GVR). Besides `CQR-001`, direct atomic subject-route coverage is checked by `CQR-102`; hard complete route validation for the Sek I and Sek II Mathematics profiles is enforced by `GVR-012`. Within their declared scopes, both already require direct atomic routes and reject inherited cluster paths as substitutes. Rule-code prefixes are listed in Appendix A.2.

### 3.2 Additional Profiles and Counterexamples

A strict tree profile may require at most one parent per node:

$$\forall c \in G: \quad \big|\{k \in G \mid (k,c) \in C\}\big| \leq 1.$$

This is not a base requirement of the polyhierarchy. Similarly, requirements targeting a goal's own ancestors are undesirable and prohibited in named validator profiles. Requirements targeting a goal's own descendants induce an effective self-dependency through inheritance and already fail the base DAG contract.

**A DAG Alone Does Not Prove Learnability**

As a transition-data counterexample, let $G = \{k, a\}$, $C = \{(k, a)\}$, and $R = \{(a, k)\}$: cluster $k$ contains atom $a$, and $a$ requires $k$. Both the direct and effective relations are acyclic and minimal. Yet $\mathrm{Atoms}(k) = \{a\}$: to start $a$, $a$ would already have to be satisfied, so the global frontier from an empty learner state is empty. The cluster requirement already violates atomic normal form; moreover, the atomically normalized relation $R_{\mathrm{at}}$ contains the self-loop $(a, a)$ and violates acyclicity. A completion witness (Section 12.2) exposes the same defect.

Three distinct claims must therefore remain separate:

1. The base skill graph is valid.
2. The selected projection is valid.
3. The target set is completely learnable from a declared entry baseline.

A single passing validator check does not automatically prove all three claims.

### 3.3 Didactic Routes from Orientation to Autonomy

A didactic route leads from motivational entry through subject learning to independent performance. Let $O \subseteq A$ denote orientation (motivation) anchors, $Z \subseteq A$ terminal autonomy goals representing independent capstone performance, and $E_{\text{route}} \subseteq A$ explicitly exempted support goals. Write $R_A = R \cap (A \times A)$ for the direct atomic relation. Without an exemption profile, $E_{\text{route}} = \varnothing$; any exemptions require a documented, machine-readable selection contract. An atom $a \in A \setminus E_{\text{route}}$ is *route-covered* if

$$\begin{aligned} \exists o \in O,\ z \in Z: \quad & (a = o \ \vee \ (o, a) \in (R_A^{-1})^+) \\ & \wedge (a = z \ \vee \ (a, z) \in (R_A^{-1})^+). \end{aligned}$$

In plain terms: the atom lies on at least one learning path starting at an orientation anchor and ending at a terminal autonomy goal. The path follows $R_A^{-1}$, the learning-flow direction from prerequisite to dependent goal.

Writing $O$ and $Z$ (rather than $M$ and $T$ as in earlier editions) avoids collisions with the mastery set $M$ and the target sets $T(U)$ and $T_\pi$. Restricting to $R_A$ prevents a cluster from acting as a hidden intermediate node in an atomic route proof. This formula is a structural connectivity criterion, not a proof of didactic quality.

Concrete route profiles additionally constrain educational stage, admissible goal types, and semantic order: orientation at entry, subject learning, optional support via memory goals where needed, and independent performance at the end. Memory goals do not thereby become mandatory checkpoints everywhere. Profile rules verify distinct claims: complete routes over effective prerequisites (`CQR-101`), direct atomic routes (`CQR-102`), and absence of `requires` on selected clusters (`CQR-103`). In the named Mathematics profiles, `GVR-012` hard-enforces complete direct atomic routes, while `GVR-013` verifies for selected local non-memory atomic goals that their direct local prerequisites are atomic. These scoped profile checks do not constitute a repository-wide proof of atomic normal form across all subjects; that base check is `CQR-001` (Section 3.1).

Higher-level dependency summaries between clusters may be derived from atomic routes. For example, an existential summary $k_2 \to k_1$ means:

$$\exists a \in \text{Atoms}(k_2),\ b \in \text{Atoms}(k_1) : (a, b) \in R_A^+.$$

A percentage coverage summary additionally requires an explicit denominator. Derived summaries never create new authored cluster requirements.

## 4 Learner State, Satisfaction, and Progress

The preceding sections describe the skill graph independently of any learner. This section adds learner state: what is persisted, what counts as effectively mastered, which goals are available next, and how progress is aggregated.

### 4.1 Satisfaction Without Backward Prerequisite Inference

**Explicit outcomes.** Exactly one binary state $m : A \to \{0, 1\}$ is persisted, with $M = \{a \in A \mid m(a) = 1\}$. $M$ is the set of explicit achievements, or explicit mastery for short. An explicit achievement on a content goal demonstrates verified understanding appropriate to that goal. Orientation, memorization, and assessments each apply their own completion contract (Section 8). No partial score creates a third mastery state.

**Preview of implicit mastery.** Sections 11 and 13 introduce learning units and the derivation of implicit mastery; here only their result is needed. For every learning unit $U$ holding a local contract durably valid for this learner, the implicit set $I_U \subseteq A$ is derived. That contract binds the unit's directly owned acquisition targets, their content revision, and the subject-reviewed evidentiary scope of the unit's assessments. On content goals, $I_U$ demonstrates independent application. Concretely, a passed assessment on linear functions can substantiate independent application of $U$'s own content goals even when those goals were not checked off individually. An independent flashcard obligation or sibling child unit may still be open; full completion of the unit is not a prerequisite for this recognition. A derivatively satisfied orientation marker retains strictly the meaning of a completed entry step; it claims neither subject competence nor that a motivational dialogue occurred.

**Effective mastered set.** The effective global mastered set is

$$M_{\text{eff}} = M \cup \bigcup_{U \in \mathcal{U}_{\text{valid}}} I_U.$$

Here $\mathcal{U}_{\text{valid}}$ contains the learner- and revision-bound authoritative local contracts that remain durably applicable, not merely currently visible units or latest content revisions. Both forms of justification carry equal weight in their effective outcome: they satisfy future prerequisites, count toward achieved progress, and remain effective across focus or schedule changes. $I_U$ never writes synthetic rows into explicit mastery storage. Section 13 defines the derivation and the subject-review prerequisites of the contract.

All subsequent formulas use $M_{\text{eff}}$. The original illustrations in Figures 8 and 9 write $M_A$ for the explicit set $M$. Those examples involve no implicit mastery, so $M_{\text{eff}} = M$ there.

**Satisfaction.** A goal $g$, whether an atom or a cluster, is *satisfied* when its entire atomic basis is effectively mastered:

$$\text{Sat}(g, M_{\text{eff}}) \iff \text{Atoms}(g) \neq \varnothing \ \wedge \ \text{Atoms}(g) \subseteq M_{\text{eff}}. \tag{7}$$

Satisfaction propagates forward only. Mastering a dependent content goal by itself never proves its prerequisites; only a valid local contract supports additional implicit recognition. Cluster progress never writes outcomes back to its children.

### 4.2 Global Frontier

The *frontier* is the set of atomic goals available next:

$$\mathrm{Frontier}(M_{\mathrm{eff}}) = \left\{ g \in A \setminus M_{\mathrm{eff}} \;\middle|\; \forall p \in A :\ (g,p) \in R_{\mathrm{at}}^{+} \Rightarrow p \in M_{\mathrm{eff}} \right\}. \tag{8}$$

An atom belongs to the frontier when it is still open itself and its entire atomic demand is effectively mastered: $\mathrm{RequiredAtoms}(g) \subseteq M_{\mathrm{eff}}$. The frontier requires all transitive effective atomic prerequisites. A mastered intermediate goal does not short-circuit this check. Partially worked goals do not count as satisfied. A cluster prerequisite is evaluated through its atomic basis.

For navigation, an analogous cluster availability set may be derived using the cluster's atomic demand $\mathrm{RequiredAtoms}(k)$ from (3), covering its own and inherited requirements expanded to atoms and closed transitively:

$$\mathrm{Frontier}_K(M_{\mathrm{eff}}) = \{k \in K \mid \neg\,\mathrm{Sat}(k, M_{\mathrm{eff}}),\ \mathrm{RequiredAtoms}(k) \subseteq M_{\mathrm{eff}}\}.$$

Cluster availability does not replace the atomic learning frontier. A user interface may additionally enforce binding entry and selection rules such as the orientation entry gate; the mathematical frontier is therefore not automatically identical to every displayed UI menu. Figure 8 shows the frontier for the running example.

### 4.3 Aggregation Without Double Counting

Let $S \subseteq A$ be an explicitly specified set of target atoms and $w(a) > 0$ the progress weight of atom $a$. Weighted total progress is the weight share of effectively mastered atoms:

$$P(S) = \frac{\sum_{a \in S \cap M_{\mathrm{eff}}} w(a)}{\sum_{a \in S} w(a)} \qquad (S \neq \varnothing).$$

Each atom contributes through its effective binary status: all or nothing. An empty target set admits no such quotient; the UI must report "no goals in the selected scope" rather than a fabricated 100%. Across multiple clusters, $S$ is formed as the set union of their target atoms; clusters and children must never be summed together. A progress figure such as 50% denotes the share of achieved goals in the scope, never half-mastery of an individual goal.

Distinct from that aggregate ratio is the segmented completion indicator in the goal tree, which may display explicitly stored and implicitly derived contributions separately. Both count in $P(S)$; a completely filled total bar means $P(S) = 1$. Any separate metric restricted to explicit outcomes must state its narrower numerator.

Under the convention $w(a) = 1$, a cluster's weight equals the count of its unique atomic descendants. That is a sound authoring convention rather than a requirement of uniform weights in every domain. Core and extension shares may be reported separately, while the committed Personal Curriculum determines what belongs to completion. An aggregated ratio is neither a calibrated probability nor a school grade; any grading scheme requires a separate, justified mapping.

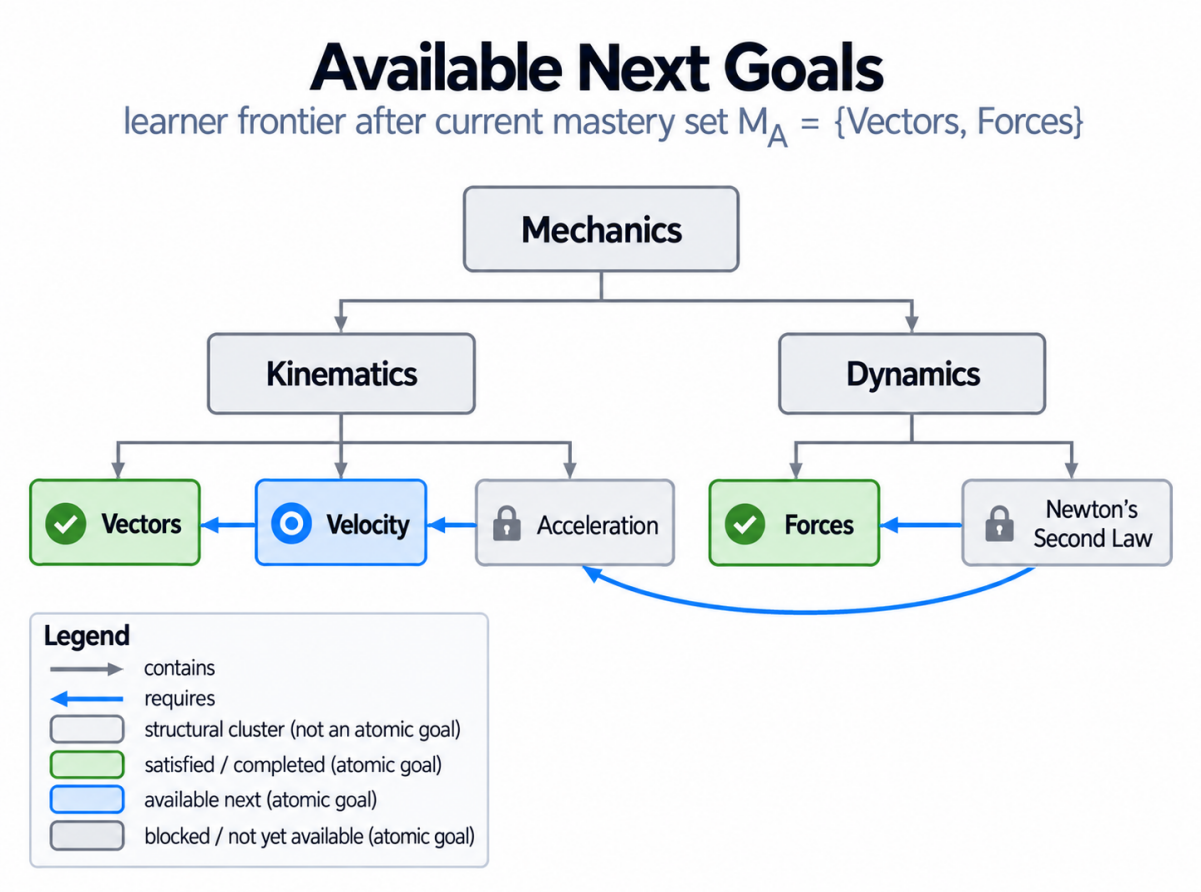



**Figure 8:** With $M_{\text{eff}} = \{\text{Vectors}, \text{Forces}\}$, only *Velocity* is available next. *Acceleration* waits for *Velocity*; *Newton's Second Law* additionally waits for *Acceleration*. Clusters are not directly selectable atomic learning steps. The figure label $M_A$ denotes the explicit set $M$, which coincides here with $M_{\text{eff}}$.

## 5 Filters, Focus, and Global Diagnostics

Up to this point, the entire graph has been evaluated globally. In practice, however, a learner works inside a scoped subset: longer-term within their Personal Curriculum (personalization Level 2) and temporarily within an active focus (Level 3). This section models every such subset as a filter and fixes the evaluation order: derive on the full graph first, restrict afterwards. Sections 6 and 7 introduce the Personal Curriculum, composition views, and projection roles in detail.

### 5.1 Filters as Predicates and Compiled Applicability

At the formal level, a filter is simply a predicate $F : G \to \{0, 1\}$ selecting the goal subset $G_F = \{g \mid F(g) = 1\}$. A filter may be evaluated from dimensional *applicability* metadata (`applicability`), such as jurisdiction, educational stage, or course profile. Let $D$ denote the set of filter dimensions and $V_d$ the value vocabulary for dimension $d$. A formally typed space of partial assignments is

$$\mathcal{V}_D = \bigsqcup_{J \subseteq D} \prod_{d \in J} \mathcal{P}(V_d), \qquad \text{Applicability} : G \rightharpoonup \mathcal{V}_D.$$

An applicability record assigns allowed value sets to a subset of dimensions $J \subseteq D$; the disjoint union records that defined domain $J$. In general metadata filtering, an omitted dimension leaves the goal unrestricted on that dimension, whereas a present dimension carrying an empty value set admits no concrete selection value. Only the query sentinel `ALL` bypasses even that restriction. Treating omitted metadata as unrestricted never permits treating an unresolved personal stage selection as fully resolved.

For an active query selection $Q$, filter evaluation proceeds dimension by dimension:

$$F_Q(g) = 1 \iff \bigwedge_{d \in D_{\text{active}}} \begin{cases} Q(d) = \text{ALL}, \text{ or } g \notin \text{dom}(\text{Applicability}), \text{ or} \\ d \notin \text{dom}(\text{Applicability}(g)), \text{ or} \\ Q(d) \in \text{Applicability}(g)(d). \end{cases}$$

A goal matches the selection when, for every active dimension, one of four cases holds: the query specifies `ALL`; the goal carries no applicability mapping; the dimension is absent from its mapping; or the queried value is explicitly permitted. Each branch is evaluated only where the corresponding partial mapping is defined. `ALL` is strictly a query sentinel and must never be persisted as an authored scope value. Explicit `applicabilityOverrides` document reviewed exceptions for auditability while runtime projection evaluates compiled `applicability` alone. Visibility under a filter does not by itself prove source coverage.

### 5.2 Inherit Globally First, Restrict Afterwards

The induced relations are

$$C_F = C \cap (G_F \times G_F), \quad R_F = R \cap (G_F \times G_F), \quad R_{\text{eff}}|_F = R_{\text{eff}} \cap (G_F \times G_F).$$

Effective prerequisites are computed *before* filtering. Otherwise, a requirement inherited from a filtered-out ancestor could vanish even when both the dependent goal and its prerequisite lie inside the focus. Likewise, structural atomicity is never recomputed from the filtered tree: $A_F = A \cap G_F$. Hiding a cluster's children never turns that cluster into a new atomic goal.

**Complete Prerequisite Chains in Projections.** Level 2 denotes the longer-term scope of a learner's Personal Curriculum, for instance for a specific jurisdiction and course profile (Section 7). A reviewed composition view specifies the canonical goal references for that scope (Section 6.3), while role validation distinguishes actual learning targets `target` from pure prerequisite helpers `prerequisiteOnly` (Section 7.2). Under the Product Owner decision of 9 November 2026, projecting a selected atomic target set $T \subseteq A$ must retain in the underlying prerequisite graph at least

$$\text{Closure}(T) = T \cup \{p \in A \mid \exists t \in T : (t, p) \in R_{\text{at}}^+\}$$

together with the canonical prerequisite relationships required to preserve their full transitive demand. Closure($T$) thus consists of the selected targets and their complete atomic demand (written $U(T)$ in earlier editions). Applicability and reviewed Level 2 composition determine $T$, but must never prune indispensable canonical intermediate goals before prerequisite chains are computed. On transition data, the canonical structure needed for inheritance and atomic expansion must remain available as well.

Consequently, if $a$ requires $b$ and $b$ requires $c$, selecting $a$ and $c$ must retain $b$ in the underlying graph so that $a$'s demand for $c$ is preserved. Losing that path makes the projection invalid; the missing nodes and relationships must be retained rather than worked around by authoring new `requires` edges in the composition view or weakening canonical prerequisites.

Structural completeness does not require displaying every retained node in the visible learner tree. Conversely, silent structural loading alone is insufficient: authored seed references remain hard-validated. For required atoms, a compiled target role backed by a verified canonical prerequisite path and recorded provenance satisfies the role obligation without requiring a manual view reference per atom. Unknown references, cycles, and missing or unreachable required atoms remain hard errors even when already mastered. Unrelated `contains` descendants are not enrolled thereby. Under the explicit Product Owner decision of 10 November 2026, $L = \mathrm{Closure}(S)$ forms the personal learning basis, where $S$ is the union of atomic seed targets from the selected reviewed subject scopes. First, their reviewed course and jurisdiction filters select the eligible seeds $S$; next, closure is computed on the unpruned canonical graph. Additional necessary atoms remain included despite those seed filters, whereas unrelated goals from other courses or jurisdictions are excluded. Every necessary atom belongs to learning, progress, and completion. $L$ is fixed by personal configuration and curriculum/projection revision independently of mastery. Mastered atoms remain satisfied in the denominator without requiring repetition; mastered intermediates never truncate the transitive closure; and an unproven entry assumption never satisfies a learning obligation.

This explicit personal compile rule supersedes the former blanket ban on automatic target addition solely at the Personal Curriculum boundary. Every open necessary atom requires an actually navigable acquisition target in the overall plan. Subject-local `prerequisiteOnly` references remain admissible when the same canonical ID is targeted elsewhere in the plan, counted once. The rule never enrolls siblings, entire foreign units, or whole subjects merely because one atom is needed, yet retains every individually necessary atom in $L$ even if that spans every atom of a subject. It neither alters official curricular assignment or source applicability nor widens focus or creates, copies, transfers, or deletes mastery; goal-type-specific completion rules remain authoritative. Backend and WebGUI implement this closed personal learning basis, while applicability validation, migration proofs, and full native, curriculum, and remote acceptance remain separately evidenced.

### 5.3 Normal Mode and Strict Mode

Both modes select their candidate goals from within the scoped subset; they differ solely in which prerequisites they check: normal mode checks only prerequisites inside the subset, whereas strict mode checks all prerequisites globally.

In normal (*optimistic*) mode, scope-relative satisfaction uses the relative basis

$$\mathrm{Atoms}_F(g) = \mathrm{Atoms}(g) \cap G_F, \qquad \mathrm{Sat}_F(g, M_{\mathrm{eff}}) \iff \mathrm{Atoms}_F(g) \subseteq M_{\mathrm{eff}}.$$

An empty relative basis does not block availability here; that neither certifies global competence nor mutates stored learner state. Inheritance, atomic expansion, and transitive closure are computed globally first, after which normal mode restricts the required atomic endpoints:

$$\mathrm{Frontier}_{\mathrm{opt}}(M_{\mathrm{eff}}, F) = \{g \in A_F \setminus M_{\mathrm{eff}} \mid \forall p \in A_F : \ (g,p) \in R_{\mathrm{at}}^{+} \Rightarrow p \in M_{\mathrm{eff}}\}.$$

In strict (*pessimistic / strict*) mode, candidates remain restricted to the focus while prerequisites are checked across the full graph:

$$\text{Frontier}_{\text{strict}}(M_{\text{eff}}, F) = \{g \in A_F \setminus M_{\text{eff}} \mid \forall p \in A : \ (g,p) \in R_{\text{at}}^{+} \Rightarrow p \in M_{\text{eff}}\}.$$

Hence, for identical inputs,

$$\text{Frontier}_{\text{strict}}(M_{\text{eff}}, F) \subseteq \text{Frontier}_{\text{opt}}(M_{\text{eff}}, F).$$

Given $g \to x \to p$ with $G_F = \{g, p\}$, $p$ remains visibly required as a transitive prerequisite in normal mode, while strict mode additionally checks $x$. Even mastery of $x$ never removes $p$ from the transitive check. The scope rule stands on its own and never converts filtered-out prerequisites into mastery. Figure 9 contrasts both modes side by side.

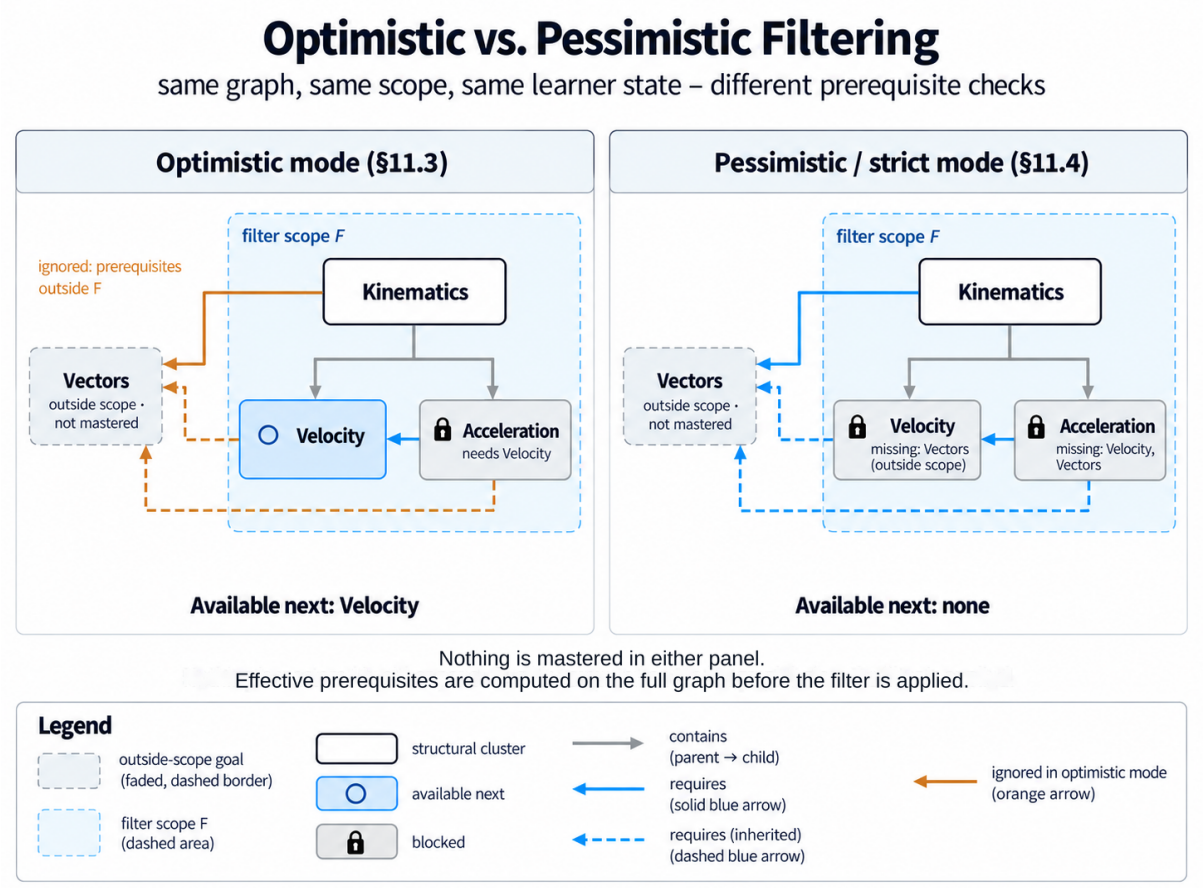



**Figure 9:** Identical focus, different prerequisite check domain. *Vectors* lies outside the scope; $M_{\text{eff}} = \varnothing$. Normal (*optimistic*) mode ignores the orange outside-scope requirements and unlocks *Velocity*; *pessimistic / strict* mode admits neither atomic goal. *Filter scope F* denotes the focus scope. Labels 11.3 and 11.4 inside the illustration refer to historical section numbers of the original draft.

### 5.4 Missing Prerequisites and Missing Atoms

A diagnostic answers two distinct questions: which prerequisite goal is unsatisfied, and which atomic goals are still missing to satisfy it? Both answers evaluate the global relations:

$$\text{Missing}(g, M_{\text{eff}}) = \{p \in G \mid (g,p) \in R_{\text{eff}} \wedge \neg\,\text{Sat}(p, M_{\text{eff}})\},$$
$$\text{MissingAtoms}(g, M_{\text{eff}}) = \text{RequiredAtoms}(g) \setminus M_{\text{eff}}.$$

Missing identifies the unsatisfied effective prerequisite nodes themselves for human explanation. MissingAtoms provides the complete global gap diagnosis: every atom in the goal's atomic demand that is not yet effectively mastered. The partitions $\text{MissingAtoms} \cap G_F$ and $\text{MissingAtoms} \setminus G_F$ separate internal from external atomic gaps. A naive list of directly required clusters can conceal missing external atoms or gaps behind intermediate nodes: for instance, if $g$ requires a cluster $k$ containing only an open atom $b$ while $G_F = \{g, k\}$, then the unsatisfied prerequisite node $k$ lies inside the scope whereas the missing atom $b$ lies outside. Diagnostics neither create new edges nor alter learner outcomes, and they must not be conflated with the blocker list of normal mode.

**Sources.** Graph Definition; Graph Validation Rules; Project Conventions.

# 6 Stable Goals, Contextual Structure

Sections 6 through 10 specify the contracts surrounding the skill graph: curricular structure, personal scope, goal types, time planning, and provenance and quality.

The skill graph answers *what* a learner should be able to do and which domain prerequisites are required for it. An educational program additionally answers *where* those goals are situated. A personal schedule determines *when* they are worked on. These three questions require distinct data: reorganizing a curriculum must neither mint a new competence nor erase previously stored learner outcomes.

The general goal model distinguishes four semantic layers and one presentation layer (Table 3). It sets the direction for incremental migration; not every data structure named here is already implemented across all skill landscapes.

**Table 3:** Five modeling responsibilities over one shared set of stable goals.

| Layer | Meaning | Example and boundary |
|---|---|---|
| Goals | Concrete competencies and subject clusters; `requires` and `contains` live here. | Solving a linear equation; foundations of linear functions. |
| Programs | Years, semesters, modules, phases, and other structural units. | Year 8 is an organizational container, not a competence. |
| Placements | Contextual link from a goal into a program unit. | The same goal is acquired in Year 8 in one program and in Year 9 in another. |
| Competency axes | Domain taxonomies and cross-cutting categories. | K1 “Mathematical reasoning” is a category; “identifying gaps in an argument” can be a concrete goal. |
| Composition view | Reviewed upper structure of the visible tree for a resolved educational scope. | A state-specific view references canonical subtrees without copying their goals. |

## 6.1 Identity Across Different Placements

The identity, attribute, and atomicity rules from Section 1 apply unchanged across every layer. A new placement therefore creates no new competence. A program unit of kind `exam` is not yet an assessment goal with tasks, scoring rubric, and learner outcome. And `phase: Q1` creates neither a parent edge to a Q1 node nor a prerequisite edge between Q1 and Q2.

Legacy trees may still contain year and phase nodes as a compatibility scaffold. In the general goal model, explicit program structure assumes that role while subject goals remain independent. The same content can thus be arranged differently across multiple educational programs without copying its goal ID or stored learner results.

## 6.2 Placements and Projections

A *projection* derives a visible tree or another navigable view from the curriculum data. It is evaluated for a *resolved context* in which dimensions such as jurisdiction, school form, stage, duration model, and course profile are fixed.

In the target model, `programUnits` form an explicit tree or forest of program units, and `goal Placements` attach goals to those units. A `primary` placement supplies the default program anchor; `secondary` indicates additional structural relevance, and `assessed` marks an assessment

reference. Secondary and assessment placements never create extra visible copies of the same goal.

Within a resolved context, exactly one matching primary placement yields a unique anchor. When no primary placement matches, the view must not invent a parent node. Multiple matching primary placements require an explicitly reviewed resolution contract; otherwise the projection is ambiguous. Context fields never alter the goal ID.

The default content view follows explicit `contains` edges alone. The program view follows program units and placements. The competency view follows catalog entries and competency references. Sibling order comes from the declared data order; topological sorting by prerequisites is a separately labeled didactic view.

### 6.3 Reviewed Composition Views

A *composition view* specifies the visible upper structure of a curriculum through structure nodes, labels, child order, and references to canonical subtrees. It contains neither newly authored atomic goals nor new `requires` edges. Wherever such a view is authoritative for a resolved context, a renderer must not override it through guesses based on titles, `phase`, or alternative placements.

For example, a structure node "Mechanics" may carry a `canonicalSubtree` reference to the canonical Kinematics cluster, pulling in its reviewed `contains` substructure. Beside it, a `goalEntry` reference may attach the single canonical atom "Add forces vectorially." Both reference existing canonical goals rather than authoring new ones. A structure node does not automatically become a domain cluster of kind `curricularArea`; that classification requires its own authoritative definition.

The compiled default tree must show each visible goal at most once and under at most one visible parent. A global DAG may have multiple content parents; projecting it to a visible tree therefore requires a deterministic, reviewed resolution rule. Overlapping subtree references are invalid without such a contract. A precedence rule for overlapping projection roles is never permission to duplicate goals in the visible tree.

Compilation should run early, in this order:

1. Resolve the context and select the matching composition view.
2. Expand canonical references and preserve their canonical atomic basis.
3. Retain complete prerequisite chains.
4. Validate projection roles and visible parents.
5. Emit the compiled artifact for the runtime.

The closure condition from Section 5.2 applies even when a view selects only a subset of canonical atoms. That makes structural validation possible before any user interface renders a tree.

**Canonical identity is preserved.** A canonical cluster must never become an independently assessable or masterable atom through a direct reference or collapsed child display. This rule holds regardless of whether a separate `semanticKind` review exists. A collapsed presentation stands for its reviewed canonical atomic basis; prerequisite checks, progress, and completion retain that basis, fixed through `canonicalSubtree` or explicit references to the intended canonical atoms. Changing projections must never copy an existing cluster-ID value down to child goals.

A cluster authoritatively classified as `curricularArea` must not appear via a direct `goalEntry` reference as an opaque, seemingly atomic runtime goal. A `canonicalSubtree` reference preserves

its reviewed descendants; alternatively, explicit view structure names the intended atomic goals. This presentation rule does not alter any projection role.

**Implementation boundary of 9 and 10 November 2026.** The existing canonical resolver already traverses prerequisite chains recursively. In the 9 November 2026 audit, a legacy `goal Entry` path could still strip the children of a canonical cluster and expose it as an atomic runtime node. Since 10 November 2026, the personal learner runtime and the WebGUI resolve canonical atomic seeds and preserve the genuine target descendants of legacy direct cluster entries; other consumers as well as full native and curriculum admission still require their own verification evidence. Compatibility is never permission to author opaque substitute atoms; normal and strict mode retain their focus semantics.

**Sources.** General Goal System and Migration; View Projection and Goal Placement.

## 7 Personal Scope, Focus, and Visible Goals

### 7.1 Four Levels of Personalization

Choosing "School," "Higher Education," or "Languages" is catalog navigation rather than learner scope. Only afterwards do the four personalization levels – Level 1 through Level 4 – begin. They must be kept distinct from domain learning units (Section 11) and from the periods of an optional calendar schedule (Section 12).

1. **Base Curriculum:** An explicitly selectable root curriculum provides the goal universe.
2. **Personal Curriculum:** The committed, longer-lived, backend-owned educational scope. For school curricula, it comprises the state or canonical context, the applicable eight- or nine-year secondary duration model (`G8`/`G9`), an explicitly resolved stage, the selected subjects, and their course profiles.
3. **Learning Focus and Active Goal:** One or more focus roots narrow the temporary working corridor; exactly one atomic goal is the immediate active target. Both focus and active goal are mutable.
4. **Mastery:** The binary outcome "mastered" or "not yet mastered" belongs to the stable atomic goal ID. Outcomes persist across view and focus changes; cluster progress is derived from them.

`GK` and `LK` denote basic and advanced course profiles (*Grundkurs* and *Leistungskurs*), and Skill Pilot also uses them as technical projection labels; wherever official course structures differ, the explicitly declared mapping applies without asserting an identically named official course type. In the Bavarian Mathematics rollout, for instance, `GK` targets the complete compulsory four-hour Mathematics curriculum at elevated level, while `LK` targets those goals plus all five modules of the separate Mathematics *Vertiefungskurs*: using Level 3 focus to work on the three modules actually taught does not remove the other two from the Level 2 target, total progress, or completion, and does not assert that Bavaria offers Mathematics as a selectable *Leistungsfach*. Likewise, under the Product Owner decision of 16 October 2026, the Chemistry rollout includes all authored topics of the applicable scope in the initial target projection while deferring a preceding topic-selection flow – including Schleswig-Holstein's officially optional medication theme in the interim `GK`/`LK` Chemistry target, while preserving its official optional status without claiming a universal curriculum requirement or learner selection. A course profile is a subject attribute and never implies a school stage; missing stage information remains unresolved.

Changing the Personal Curriculum revalidates the current focus and active goal against the new target set without deleting or writing global mastery outcomes. Configuring the Personal Curriculum is owned by SkillPilot's first-party WebGUI surfaces; the language-model coach connected through the concrete OpenAI V1 adapter reads that scope and neither presents nor mutates it.

Under the Product Owner decision of 10 November 2026, the complete personal learning basis is $L = \text{Closure}(S)$ from Section 5.2: the valid seed targets together with all necessary atomic prerequisites on the unpruned canonical skill graph. It remains fixed independently of mastery; existing valid mastery counts as satisfied rather than causing removal or forced repetition. A narrow focus restricts only the temporary working corridor. Necessary goals outside the focus remain authoritative for total progress and completion; completing a focus never completes the entire Personal Curriculum.

## 7.2 Target Role and Prerequisite Support

A reference inside a composition view may explicitly carry one of two *projection roles*:

`target`.
The goal belongs to the selectable learning set, frontier, progress, and completion of this view.

`prerequisiteOnly`.
The goal remains available for prerequisite checks with its global identity and stored state; within this view it is neither selectable nor counted in the local progress denominator.

When omitted, the role defaults to `target`. Subtree references pass their role down to descendants. Overlapping role assignments are resolved by specificity: a direct `goalEntry` reference overrides subtree inheritance; a nearer subtree root overrides a broader one; only at equal specificity does `target` win. This precedence resolves the role rather than permitting duplicate visible occurrences.

Authored subject roles are never guessed from age or phase. By contrast, the explicitly authorized personal closure rule adds necessary atomic acquisition targets to the full plan $L$ without rewriting authored subject roles or inventing new dependencies. A mathematics goal can be a learning target in Mathematics and supporting prerequisite context in Physics while still being counted once in a joint plan. Its Physics role neither erases Mathematics outcomes nor creates a second competence acquisition. However, a genuinely necessary atom cannot remain mere prerequisite support across the entire Personal Curriculum without an acquisition target. Official curriculum assignment and inclusion as necessary preparation remain distinct claims. The backend and WebGUI implement the closed personal learning basis; applicability validation and migration admission remain separate work.

At the personal compilation boundary, the Product Owner decision of 10 November 2026 supersedes the earlier blanket requirement of 7 November to manually reference every reachable prerequisite with an explicit role:

- Authored seed references and subject-local roles remain validated.
- A necessary atom's derived global target role, backed by a checked canonical prerequisite path and recorded provenance, suffices without a manual target or support reference per atom.
- Unknown references, cycles, and missing, unexposed, or unaccounted required atoms remain hard errors regardless of mastery and focus.

This compile-time validity check is distinct from later Level 3 focus selection: a validly resolved prerequisite outside the active focus still does not block normal mode.

Under the Product Owner decision of 11 November 2026, these hard errors belong to validation, curriculum QA, and native acceptance rather than to learner reads. When the canonical graph of a learning context is not strictly compilable, the runtime keeps the reviewed seed scope without the derived closure, claims no learning basis, offers no goal whose prerequisite demand is broken, and logs the failure once per curriculum and error code; learner state, results, and export stay readable. Errors in the seed selection itself remain hard errors. The rule covers the many non-canonical curricula whose root cluster contains and requires its own orientation goal and changes none of their data.

*Transition-data example:* Suppose a Physics goal $g$ requires a Mathematics cluster $k$. Its canonical atomic basis and transitive prerequisites are resolved prior to display filters. A subject-local `prerequisiteOnly` reference to $k$ may remain in Physics, yet every genuinely required atom still receives an acquisition target in the full Personal Curriculum – whether in Mathematics or as derived preparation – without manual per-atom references or enrolling an entire extra subject. Canonical identity, provenance, and mastery remain intact.

### 7.3 Applicability Is Not Placement

`applicability` holds compiled filter metadata for runtime views, such as a state filter. It is derived from inspectable sources, mappings, provenance, and reviewed overrides. It answers in which filtered contexts a goal is visible, whereas a placement answers where the goal sits inside a program structure. Neither datum is evidence of individual mastery.

For state-specific extensions, an explicitly declared inheritance boundary can prevent a broad ancestor source mapping from being reported as source provenance for newly added descendants; direct sources and mappings on the extension branch remain authoritative. Likewise, the visibility of a local practice or assessment endpoint may be derived from its prerequisites under a dedicated contract, establishing applicability without automatically asserting official source-curriculum coverage.

### 7.4 Switching Focus Without Implicit Mastery

A focus consists of the union of the selected focus roots and their visible descendants, intersected with the Personal Curriculum. Normal mode checks transitive effective prerequisites within that focus; explicitly selected strict mode checks them globally. In normal mode, an outside-focus prerequisite neither blocks availability nor becomes mastered by being masked. Section 5.3 gives the formal frontier definitions.

Focus widening follows the unique visible parent path. Each backend-published widening option must form a strict superset and add at least one unmastered target goal. With multiple roots, it replaces only the covered branch and carries independent roots forward. The coach adopts the published option verbatim and waits for learner agreement before changing focus.

An automatic widening proposal is offered only once the current focus is genuinely complete; an empty frontier alone may indicate a blocked, incomplete focus rather than completion. After widening, newly included unmastered prerequisites are checked normally: mastering a dependent goal never writes mastery backward. By contrast, application recognition already derived under a valid local contract remains globally effective, so later consumers – goals requiring that competence – as well as subsequent plan or focus switches never reopen it.

**Sources.** Levels of Personalization; View Projection and Goal Placement.

# 8 Goal Types and the Meaning of an Outcome

For each goal, the skill graph receives only the finished binary outcome established under that goal type's workflow. The graph itself does not judge whether an explanation was convincing, an orientation dialogue closed appropriately, or every required flashcard was recalled reliably; those decisions belong to each goal type's completion contract.

Every learning goal has a binary outcome: mastered or not yet mastered. The goal type defines what "mastered" means. For a content goal, it means explicitly established understanding, while valid assessment derivation additionally attests independent application. For orientation, it marks completion of the motivational entry rather than a subject certificate. For memorization, it denotes passing the goal's own controlled recall check over the required cards. For an assessment goal, it denotes passing the assessment. Differences among completion workflows introduce neither partial mastery levels nor parallel goal outcomes. Table 4 summarizes the four goal types.

**Table 4:** The four goal types and the meaning of their binary outcome.

| Goal type | What "mastered" means | How the outcome is established | Implicitly derivable |
|---|---|---|---|
| Content goal | Understanding is established; under implicit mastery, independent application is attested. | Sufficiently evidenced mastery in the coach-led workflow (Section 8.1) or a valid local contract (Section 13). | yes |
| Orientation goal | Motivational entry is completed; not a subject-matter certificate. | Tailored follow-up, positive summary, and learner agreement to close (Section 8.2). | marker only |
| Memory goal | Completion recall over the required card deck is passed. | Verified recall of all required cards without card hints (Section 8.3). | no |
| Assessment goal | The assessment is passed. | Rubric-based grading of the single complete submission (Section 8.4). | no |

## 8.1 Understanding and Application

A content goal describes conceptual or procedural understanding. Grading must cover the aspects demanded by the goal description. Solving a single task is one piece of evidence rather than automatic mastery of the entire goal: multi-step transfer within one task can suffice, whereas imposing an arbitrary minimum task count would be just as unfounded as crediting every isolated correct answer.

In the coach-led workflow, sufficiently evidenced mastery is persisted immediately, and the coach may report a saved outcome only once persistence is confirmed. Requesting separate learner consent before saving is not domain evidence and is not added as an extra completion gate for ordinary content mastery. Moving on to the next topic remains a separate choice: the coach offers the backend-selected next target and waits for recognizable learner agreement before starting new material.

### 8.2 Orientation as a Distinct Completion Contract

A goal authoritatively classified as `orientation` presents concrete possibilities and honest positive perspectives on the upcoming material. It tests neither prior knowledge nor terminology, calculation skills, recall, transfer, or exam performance. Like every learning goal, its outcome is binary; grading personal motivation on a percentage scale would be misleading.

The workflow proceeds in three steps:

1. **Follow-up.** When the learner picks one of the offered interests, the coach opens a tailored motivational follow-up linking that interest to something concrete the learner can understand, explore, shape, or do, and invites a low-pressure personal reaction, choice, or question. Merely repeating an offered label does not complete orientation.
2. **Summary.** After a meaningful response to that follow-up – or an explicit wish to leave orientation – the coach gives a positive summary together with a separate offer to close.
3. **Closure.** Only a further learner reply agreeing to that closure establishes the completion marker.

This workflow must neither be reinterpreted as a subject exam nor transferred to ordinary content goals. Orientation nodes are prerequisite-free atomic entry anchors; assessment and memory metadata never belong on the same node.

At a genuine zero-progress entry into a projected scope, open orientation goals are the only selectable frontier goals. Learners with established subject progress are not reset to this entry gate.

### 8.3 Memorization: Practice and Verified Recall

A memory goal bundles the required cards of a flashcard deck. Three distinct concepts apply:

- **Card practice.** A spaced-repetition system (SRS) schedules card review intervals over time.
- **Verified recall of a card.** Controlled recall records whether an individual card was answered accurately without card hints; flipping cards or self-rating during practice does not constitute verified recall.
- **Goal completion.** Mastery of the learning goal itself is established by passing its own controlled completion check across the required deck – *completion recall*.

Card-level process lanes and goal completion must remain distinct; they introduce neither a second mastery tier nor an extra stored goal outcome.

**Completion condition.** Let $\mathrm{Cards}(g)$ denote the fully resolved, valid set of required cards for memory goal $g \in A_{\mathrm{mem}}$, and let $\mathrm{RecallPassed}(g)$ denote an explicitly established passing completion-recall check over that required card set. That goal-level completion check is neither a mere tally of current per-card flags nor an extra data field or separate assessment node:

$$\mathrm{Complete}_{\mathrm{memory}}(g) \iff \mathrm{RecallPassed}(g).$$

Missing or unresolved card definitions never count as an empty, automatically completed deck, and clearing a due practice queue never substitutes for completion recall.

**Due dates do not withdraw completion.** Ordinary completed learning goals carry no recurring review due state. For memory goals, a subsequent card due date governs card practice rather than automatically withdrawing established goal completion. Likewise, a later voluntary card check updates that card's process lanes without automatically revoking completed goal mastery. The earlier conjunction "all cards verified and no card due" is superseded by the binding Product Owner clarification.

**Card scheduling under SM-2.** The currently documented SRS contract tracks SM-2 (Super Memo 2) state per card: interval, repetition count, easiness factor, and next review timestamp. Any card without stored state is due. Successful practice extends the interval under the documented rules, whereas insufficient recall resets the repetition sequence; scheduling never replaces verified recall.

**Verified recall workflow.** From an active memory goal, the learner may start verified recall – the *hard recall check* – even while the SRS practice queue is non-empty. First only the card prompt is retrieved, and the learner answers without card hints; only afterwards is the expected answer loaded and the attempt recorded as passed or failed. Passing also updates the card's SRS lane as a strong successful retrieval, whereas failing clears the card's verified status and returns it to practice as due.

**Implementation boundary.** Historical code inspection noted that `LearnerService` and `useSrsMastery` derive a value from current verified-recall flags. That observation neither authorizes withdrawing completed mastery nor proves full conformance with the completion contract above. This documentation clarification alters neither runtime behavior nor learner data; general completion accounting consumes the same binary goal outcome without introducing a separate memory-mastery tier.

### 8.4 Assessment Goals in the Current Workflow

Assessment goals carry `examData` with task content, solution content, and a scoring rubric specifying maximum points, passing threshold, and gradable steps. For validly awarded step points $p_j$, the current contract sets:

$$score = \min\left\{\sum_j p_j,\ maxPoints\right\}, \qquad passed \iff score \geq passingPoints.$$

Step points follow the authored rubric criteria; the total-score formula alone legitimizes neither invented step points nor negative scores nor altered task prompts.

**Access.** Ordinary learning access checks transitive effective prerequisites, including atomic satisfaction of required clusters. Assessment goals additionally grant general *direct assessment access*: missing preparatory mastery does not block an intentional assessment attempt, provided task availability, personal scope, and an authorized learner session hold. Existing direct assessment access must be kept distinct from the new global implicit-mastery derivation, which is **not yet implemented** on `main`.

**Exam-mode workflow.** In Exam Mode, the learner receives the complete, verbatim task prompt and works through it without content hints. The single complete submission is graded strictly against the rubric. On pass, mastery is persisted immediately; on failure, neither an outcome nor a score is persisted and any existing mastery remains untouched. Only confirmed persistence authorizes reporting that the pass is saved.

**Post-grading semantics.** After grading, the learner receives the outcome, score, per-step feedback, and model solution; new material begins only after a subsequent learner choice. Unlimited retries of the same task remain permitted. Seeing the solution may reduce the evidential weight of a later retry without creating a second qualification tier: stored exam mastery remains an explicit pass of that assessment goal. Broader application recognition requires a valid, domain-reviewed local contract; a legacy unreviewed scoring rubric does not satisfy that requirement automatically. Neither an extra per-learner qualification field nor a second partial assessment is introduced.

**Sources.** Node Types and Exam Workflow; Project Rules on Orientation, Evidence, and Continuation; Backend Derivation of Memory Mastery; Frontend Derivation of Memory Mastery.

## 9 Time Planning as a Complementary Overlay

A schedule binds a declared planning scope, a curriculum revision, a calendar, available capacity, and a plan version. Absolute dates never belong on canonical goals, clusters, or placements: the same content can be studied at different times across courses without altering its canonical identity.

Time planning is an execution overlay rather than a fifth personalization level. It may prioritize goals currently available in the active working corridor, yet it creates neither prerequisite edges nor a new frontier nor learner outcomes. Whereas a learning unit expresses domain coherence, a dated schedule must additionally ensure that each prerequisite is scheduled in time before its planned dependents.

Available capacity accounts for actual instructional or study minutes after holidays, cancellations, and other exclusions. Planned activities, protected reserve, and committed subject budgets must fit within that capacity without double counting. Section 12 specifies the formal budget check and its requirement for calibrated effort estimates.

Planned targets and actual attainment share the same named scope. Explicit outcomes together with validly derived application recognition form global cumulative attainment; manual teaching-coverage flags neither establish nor conceal completion, and missing learner results remain visibly unavailable. New plan revisions must never silently reinterpret historical goals, stored outcomes, or past progress statements.

The documented local planning pilot and the target backend-owned architecture must be kept distinct; in particular, the learning-unit definition claims no implemented cross-subject budget validator.

**Sources.** Curriculum Time Axis, Buffer Planning, and Pacing.

# 10 Provenance, Resources, and Defensible Quality Claims

## 10.1 Three Kinds of Links

Curricular provenance, first-party visualization, and optional study resources serve distinct purposes. An official source reference justifies including or phrasing a learning goal; a first-party goal illustration conveys its core idea; a textbook, video, or simulation supports study. None of these links constitutes evidence of individual learner mastery.

The binding content architecture states: resources reference the curriculum; the curriculum never privileges resource providers. New optional external learning resources live in separate content packs pointing to stable goal IDs. Selecting a content pack alters neither Personal Curriculum nor prerequisites, mastery, or completion rules, and incomplete resource coverage must never block ordinary learning.

Official sources, mappings, and structured provenance remain inspectable on the curriculum, while first-party goal visualizations retain their separate canonical reference and quality contract. Legacy external links on canonical goals are transitional material rather than a permanent second authoring architecture, and returning a link alone never proves that a coach read the linked resource.

## 10.2 Source Rationales for Goals and Relations

An inspectable provenance chain links each canonical goal to a source document, a specific source passage, a traceable extraction, and a reviewed mapping decision – whether one-to-one, partial, aggregating, or splitting across multiple goals. A human-readable rationale explains that relationship honestly rather than presenting every mapping as a verbatim curriculum quote.

For `requires` and `contains`, provenance must likewise distinguish relationships explicitly stated in the source from domain-justified modeling decisions. Plausible goal wording never licenses retroactively invented provenance; unresolved sources, superseded editions, and conflicting mappings remain visible findings.

A human-readable rationale renders existing structured evidence rather than forming a second authoritative registry. Structured references and fingerprints allow documents, exports, and runtime tools to emit consistent rationales. MEM (*Metadata for Educational Media*) links digital educational media to curriculum topics and competencies, and FWU denotes the *Institut für Film und Bild in Wissenschaft und Unterricht gGmbH*, whose curriculum ontology is referenced here (see the MEM/FWU roundtrip plan). Cross-checking against those ontology records can corroborate or challenge the classical source route, replacing it only under an explicitly reviewed contract for a declared scope.

## 10.3 Algorithmic Validity and Domain Quality

Quality layers build upon one another without making identical claims. A parser catches malformed JSON; a reference check catches unknown IDs; a cycle check catches forbidden edge loops. None of those checks alone judges whether a competence is well-scoped, an explanation clear, or a task didactically sound.

Each validity and route profile from Section 3 applies to a named validation scope: a local migration gate or a single passing structural check does not automatically certify every skill-graph, projection, and learnability requirement.

For in-depth curriculum QA, SkillPilot tracks five concurrently active evidence areas – D, P, A, M, and V denote area names rather than mathematical symbols:

- **D – Description:** Is the goal wording factually accurate and clear?
- **P – Positive understanding evidence:** Which observable performance demonstrates the intended understanding?
- **A – Atomicity:** Is the competence appropriately scoped?
- **M – Memory:** Is the decision on necessary memorization elements justified and current?
- **V – Visualization:** Is the illustration didactically appropriate and bound to the reviewed image and goal hashes?

Maturity tiers M0 through M7 structure automated curriculum QA and are unrelated to the mastery set $M$ (Appendix A). Tier M7 denotes complete passing automated checks for these five areas and all prerequisite lower tiers across the declared scope. It is neither human release sign-off nor human learner testing nor proof of instructional effectiveness – claims that require dedicated studies and evidence of their own. Even multiple concurring AI reviews never substitute for a separately required human decision.

Every review binds the exact revision of its artifacts: a valid image review for an older asset hash does not certify a replacement image, and editing goal wording or prerequisites can invalidate prior content evidence, whereas a purely technical relocation may qualify under a narrowly justified compatibility proof. Neither case may be masked by wholesale hash refreshing.

### 10.4 Publication and Interchange

Domain edits happen in canonical authoring sources; packages, goal books, illustrations, and validation reports are rebuilt from them while historical published archives remain immutable.

JSON and ontology packages are built from the same canonical revision, with the JSON runtime model serving as the normative reference. Ontology roundtrips must preserve goal identities, relations, projection roles, texts, and domain fields under their semantic-equality contract, while binary assets require byte-identical payloads. An interchange format must never introduce divergent runtime semantics.

External goal references must resolve completely: a Physics package with mathematical prerequisites must ship their definitions rather than empty placeholders. That complete structural *closure* does not by itself decide which goals become selectable `target` items for a learner.

Publication and validation claims always name the artifact, scope, version, and bound source revisions. A passing local build proves neither host rendering nor pedagogical effectiveness, and this compendium grants neither release nor production sign-off.

### 10.5 Licensing and Private Learner Data

Apache-2.0 governs software, technical schemas, developer documentation, and functional coach instructions; CC-BY-4.0 governs SkillPilot's own skill landscapes, goal texts, didactic tasks, flashcards, illustrations, and educational documentation. Classification follows function rather than file extension, without blanket dual licensing across mixed collections.

Third-party sources, dependencies, media, and trademarks retain their own terms; source citation and AI provenance establish neither rights ownership nor content quality. Exports carry applicable licenses, provenance, and modification notices, while private learner outcomes, session records,

and conversations belong neither under the public content license nor inside public source or review packages.

**Sources.** Source and Resource Links; Human-Readable Source Rationales; Graph Validation Rules; Curriculum Quality Maturity and Routes; Learning Goal Book and Evidence Review Pipeline; Dual Curriculum Package Releases; Licensing.

# 11 Learning Units: From Learning Promise to Recursive Completion

The skill graph specifies which goals exist and what they require, yet it does not say which goals together form a coherent domain unit that a learner starts and completes as a recognizable whole. That responsibility belongs to a learning unit. This section defines a unit's scope, its derived completion, and the boundaries of prerequisites between whole units; Section 13 builds implicit mastery on top of it.

**Domain Unit Rather Than Calendar Bucket**

A **learning unit** (German *Lerneinheit*, future technical name `LearningUnit`) binds an understandable learning promise to its own learning goals and, where appropriate, smaller learning units – its *child units*. The structure is recursive: Mechanics may contain Kinematics and Dynamics, and Kinematics may in turn contain smaller units. School years, semesters, and study weeks are possible scheduling anchors rather than synonyms for a learning unit. Personalization Levels 1–4, QA maturity tiers, and the difficulty field `level` keep their existing meanings.

**Purpose: A Secure Foundation Before Building On**

A learning unit establishes solid, independent application of its promise before the planned course builds on it. Its assessments are designed to expose essential misconceptions: an exposed gap leads to focused further learning and a fresh, equivalent assessment, whereas a false completion would send learners onward on an unstable foundation. The ordinary grading contract applies unchanged – there is no hidden pass rule and no second stored outcome; an isolated slip with sound reasoning is judged differently from a missing central understanding. This is a didactic requirement of the unit definition, not a technical access lock (Section 11.5).

## 11.1 Motivation and Distinct Domain Claims

A learning unit answers first: *What will I be able to do afterwards, and why does it matter?* A motivational opening makes that promise tangible before individual tasks or goals are presented. That opening is a didactic role: when modeled as a genuine orientation goal, it follows the orientation completion contract (Section 8.2), whereas a synthetic heading never mints an extra learning goal. Derived unit completion never asserts that a motivational conversation actually took place.

A unit boundary follows a meaningful conceptual transition: which coherent understanding and application must be secure before something new depends on it? An existing assessment, a target quota, a tidy heading, or the density of prerequisite edges is at most an indicator, never the definition. Sparse, transitively minimal chains can form strong units, and shared elementary foundations do not force every later application into one giant unit. Edges are neither added nor removed to make a desired cut fit; a questionable edge is a separate subject-matter finding. A terminal or synthesis unit needs no later consumer to justify it. The design criteria and worked examples apply these criteria to two candidates.

Relative to its child units, a parent unit may make one of three distinct claims:

- **Summarize (aggregate).** The unit merely groups competencies already acquired in its children; in that case it needs no new assessment of its own.

- **Integrate (synthesis).** When the unit additionally promises independent integration across child competencies, its own synthesis assessment may test that integrative claim over competencies already acquired in child units, without inventing artificial new content goals. The role of a synthesis assessment is that extra integration proof rather than re-passing every child.
- **Own domain competencies.** A unit may also introduce genuine new domain competencies of its own, which require their own direct acquisition ownership and valid assessment coverage.

### 11.2 Direct Acquisition, Child Units, and Total Scope

An authoritative, versioned unit $U$ is specified by three sets:

- $\mathrm{Own}(U)$ denotes the directly assigned atomic *acquisition targets* – the unit's own goals. Genuine orientation, assessment, and memory goals may belong to $\mathrm{Own}(U)$; synthetic structure markers never do.
- $\mathrm{Children}(U)$ denotes the mandatory child units – the *required children* – after any authored choices have been resolved.
- $T(U)$ denotes the deduplicated recursive goal set – the *total scope* – given by:

$$T(U) = \mathrm{Own}(U) \cup \bigcup_{V \in \mathrm{Children}(U)} T(V). \tag{9}$$

In Figure 10, Kinematics owns $\{\mathsf{a}, \mathsf{b}, \mathsf{q_K}, \mathsf{k_K}\}$, Dynamics owns $\{\mathsf{d}, \mathsf{q_D}\}$, and Mechanics owns only the synthesis assessment $\{\mathsf{q_S}\}$; the total scope of Mechanics spans all seven goals.

Scope and ownership obey five rules:

- Resolved total scope is non-empty. An empty navigation marker is never a completed learning unit, whereas a pure aggregate may have empty $\mathrm{Own}(U)$ provided its required children carry a non-empty total scope.
- Every acquisition target has exactly one directly owning unit: the smallest domain-responsible authoritative unit (rule R4 in Section 13.3). An ancestor view may display the same stable goals without claiming them again in $\mathrm{Own}(U)$.
- Overlapping references unite as sets, minting neither duplicate goals nor double-counted progress points.
- Direct ownership of $\mathrm{Own}(U)$ together with the domain-reviewed meaning of the unit's own assessments forms the unit's *local contract* (Section 13); switching focus never alters that ownership or contract.
- Every ordinary content goal in $\mathrm{Own}(U)$ requires an own local assessment consumer – an assessment owned by $U$ that requires it directly or transitively (R1) – and at least one such assessment needs reviewed evidence that its tasks and normal grading rubric really test the goal (R2). Section 13.3 lists all validity conditions R1–R5 and where each is enforced. Without them the local contract is invalid; an ad-hoc waiver mask never substitutes for it.

Embedding reuses the existing program and placement layers or reviewed composition views (Section 6). Expanding a UI cluster does not create a learning unit, nor does every `contains` cluster automatically become one; in particular, general cluster satisfaction from Section 4 remains distinct from the explicitly authorized unit hierarchy defined here.

### 11.3 Completion Without Stored Unit Mastery

A learning unit is complete if and only if all of its own goals are effectively mastered and every required child unit is complete:

$$\text{Complete}(U) \iff \big(\text{Own}(U) \subseteq M_{\text{eff}}\big) \wedge \bigwedge_{V \in \text{Children}(U)} \text{Complete}(V). \tag{10}$$

Completion is derived from atomic outcomes rather than persisted as a separate `UnitMastery` record that writes results back to children. Assessment and memory goals remain strictly explicit, so passing a parent assessment can replace neither an open child assessment nor missing verified card recall.

For the three unit forms from Section 11.1, this means the following. A pure aggregate may carry no assessment of its own, deriving completion from its required children and any other own obligations. A synthesis unit may carry an own synthesis assessment without new content goals; passing that synthesis assessment is then additionally required. A pure flashcard collection establishes no claim of independent domain application, and its explicit recall duties remain mandatory; recall obligations belonging to a child unit must never be hoisted to the parent merely to make the child appear complete.

### 11.4 Recognition Before Full Unit Completion

Under the binding target concept – which is **not yet implemented** on `main` – passing an appropriate local assessment can recognize a unit's own content goals before an independent flashcard deck or sibling child unit is finished, provided that no other open own assessment or memory goal of the unit still requires them (Section 13.2). That recognition is globally effective, and a still-open parent synthesis assessment does not withdraw it.

Recognition is therefore computed solely over each unit's *own* goals excluding assessment and memory goals. A parent unit never forms an enlarged recognition scope over all of its descendants; otherwise passing a parent assessment could bypass subordinate child assessments or card-recall duties. A parent assessment may attest additional integration without retroactively expanding what child contracts certify. Section 13 states the exact formula and its task-authoring conditions.

### 11.5 Prerequisites and the Limits of a Unit Sequence

Two kinds of prerequisites must be kept distinct:

- An atomic `requires` edge targeting a competence requires that competence alone rather than completion of its entire owning unit.
- A *whole-unit prerequisite* is an explicitly authorized dependency between complete learning units: it requires *full completion* of the predecessor unit and belongs to the course composition contract outside the base relations $C$ and $R$.

Neither the compiler nor the coach may conflate those two statements.

The unit hierarchy itself must be acyclic. Anyone who further claims an indivisible whole-unit sequence – in which each unit is completed in full before the next begins – must also validate the projected unit-dependency graph. Atomic acyclicity alone does not suffice: for $a \to b \to c$ with $\text{Own}(U) = \{a, c\}$ and $\text{Own}(V) = \{b\}$, unit $U$ depends on $V$ via $a \to b$ while $V$ depends on $U$ via

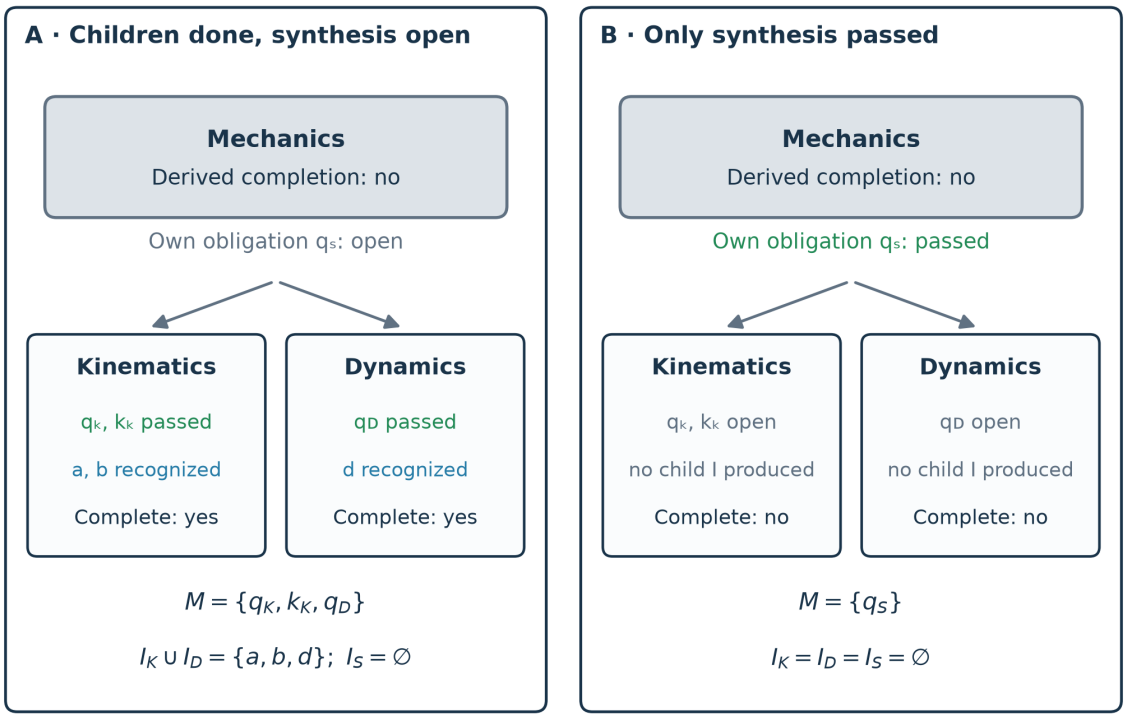



**Figure 10:** Illustrative hierarchy with independent child contracts. Case A (left): both child units are complete and their skills are globally recognized; only the open synthesis assessment $\mathsf{q_S}$ keeps parent completion open. Case B (right): only the parent assessment is passed; it satisfies neither child assessments nor memory recall nor child-unit recognition. Grey arrows denote the unit hierarchy alone; subject `requires` edges are listed separately underneath. Here $\mathsf{q_K}$, $\mathsf{q_D}$, and $\mathsf{q_S}$ are assessment goals, $\mathsf{k_K}$ is a memory goal, and $\mathsf{a}$, $\mathsf{b}$, $\mathsf{d}$ are content goals. "Child $I$" and "Parent $I$" denote each unit's implicit set $I_U$.

$b \to c$. Grouping produces a mutual block cycle, so $U$ and $V$ admit no indivisible whole-unit order; resolving that finding requires re-scoping the unit boundary or explicitly permitting interleaved acquisition rather than inferring silent unit locks from atomic edges.

The atomic normal form $R \subseteq A \times A$ and its ordered migration from Section 3 remain in force: secure complete atomic prerequisites first, remove redundant non-atomic edges second, and transitively reduce the atomic relation last. That rule implies no automatic minimality rule for a separate whole-unit prerequisite relation yet to be specified.

**Access Profile Still Undecided**

A domain acquisition order is not yet a technical access gate. Whether and when a whole-unit prerequisite enforces a hard technical entry lock – a *unit lock* – and how deliberate direct assessment attempts behave under it remains **undecided**. Any such profile requires a separate product decision and host verification; the learning-unit definition itself introduces none.

Only *if* an explicit gated access profile is later chosen could authorized predecessor units $\mathrm{Pred}(U)$ define:

$$\mathrm{Open}(U) = \bigwedge_{V \in \mathrm{Pred}(U)} \mathrm{Cleared}(V), \qquad \mathrm{Cleared}(U) = \mathrm{Open}(U) \wedge \mathrm{Complete}(U).$$

Here Complete records established evidence whereas Cleared expresses conditional course policy; neither opening a unit nor computing clearance writes mastery, and this conditional illustration does not settle whether direct assessment attempts remain open inside a locked unit.

**Sources.** Learning Units; Learning-Unit Progression and Access Policy; Graph Definition; Assessment-Gated Completion and Implicit Mastery.

# 12 Optional Acquisition and Time Planning: Sequence and Budget

This section answers two planning questions: can the goals of a schedule be acquired in the proposed temporal sequence at all, and does the available time suffice? Both checks evaluate a reference plan rather than a learner's stored state.

**Status: Optional Planning Profile**

Learning units carry domain coherence and recursive completion. An optional schedule assigns their acquisition targets to planning periods – such as school years or semesters – and checks available time capacity. That calendar assignment neither defines learning units nor alters base skill-graph validity or stored mastery, nor does it introduce technical unit locks. No production cross-subject budget validator is claimed.

A domain learning unit may be acquired at different times across educational programs. Here a planning period denotes a scheduled interval – figuratively a construction stage: competencies remain the building blocks and retain their domain prerequisites. Nothing here requires completing every subject of one calendar year before any later goal becomes accessible.

## 12.1 Plan Scope, Entry Basis, and Single Initial Acquisition

For a resolved reference plan $\pi$, let

$$\begin{aligned} &T_\pi \subseteq A && \text{planned acquisition targets,}\\ &B_\pi \subseteq A \setminus T_\pi && \text{explicitly assumed entry basis,}\\ &n_\pi \in \mathbb{N}_{>0}, \qquad \ell_\pi : T_\pi \to \{1,\dots,n_\pi\} && \text{planning period.} \end{aligned}$$

The plan thus specifies what is to be acquired ($T_\pi$), what is assumed present at entry ($B_\pi$), and in which of the $n_\pi$ planning periods each target is acquired for the first time ($\ell_\pi$). The period sets and their prefixes are

$$S_i = \{g \in T_\pi \mid \ell_\pi(g) = i\}, \quad S_{<i} = \bigcup_{j<i} S_j, \quad S_{\le i} = \bigcup_{j\le i} S_j.$$

This calendar assignment (Figure 11) references – rather than replaces – the domain acquisition ownership of learning units. Each planned target receives one initial acquisition period in the plan, while another educational program may place it differently. Subsequent review, deepening, or application constitutes additional scheduled activity rather than a second goal identity. $B_\pi$ is a planning assumption that writes no learner mastery; individual gaps inside that entry basis require separate diagnosis and extra study time.

## 12.2 Structural Feasibility Through a Completion Witness

The assumed entry basis sets a planning horizon whose prior acquisition history is not reconstructed. Solely for this hypothetical reference-plan check, define

$$R_{\mathrm{at}}^{\pi} = \{(a,b) \in R_{\mathrm{at}} \mid a \notin B_\pi\}.$$

Let $\mathrm{Frontier}_\pi$ denote the frontier from (8) evaluated over $R_{\mathrm{at}}^{\pi}$ and its full transitive closure. Every planned target and earlier planning period retains its full transitive demand; reconstruction stops

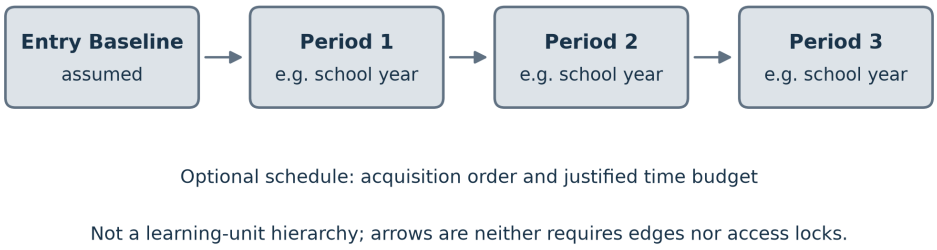



**Figure 11:** Calendar ordering is not a recursive learning-unit hierarchy. Arrows are neither `requires` edges nor technical entry locks.

only at declared entry competencies in $B_\pi$. That planning boundary never licenses stopping at mastered intermediates during real learner evaluation.

A *completion witness* for a planning period is an ordering of its targets in which each goal is available as soon as its turn arrives. A period $S_i$ of $s_i$ goals is structurally feasible when its elements can be enumerated as $a_1, \ldots, a_{s_i}$ such that

$$a_j \in \mathrm{Frontier}_\pi(B_\pi \cup S_{<i} \cup \{a_1, \ldots, a_{j-1}\}) \qquad (1 \le j \le s_i). \tag{11}$$

For instance, if $g$ requires $p$, both lie in the same period, and all of their remaining prerequisites are already satisfied, then $p, g$ is a completion witness whereas $g, p$ is not. Evaluation uses the global check domain, preserving inherited, cluster-expanded, and resolved cross-subject prerequisites; a normal-mode focus filter could mask relevant dependencies. The hypothetical witness uses neither direct-exam shortcuts nor anticipated implicit recognition.

For diagnostics, define the required atomic demand from (3) under the plan relation:

$$\mathrm{RequiredAtoms}_\pi(g) = \{b \in A \mid (g, b) \in (R^\pi_{\mathrm{at}})^+\}. \tag{12}$$

Existence of a completion witness necessarily implies

$$\forall g \in T_\pi : \quad \mathrm{RequiredAtoms}_\pi(g) \subseteq B_\pi \cup S_{\le \ell_\pi(g)}. \tag{13}$$

Every required atom therefore belongs to the entry basis or is acquired no later than the dependent goal's own planning period. A required competence outside $B_\pi \cup T_\pi$ is an unplanned gap; a required target assigned to a later period is a sequence error. Same-period prerequisites remain valid whenever a witness exists. Base skill-graph validity – including cycles after atomic expansion – must already hold prior to this set check; the explicit witness ties period assignment to a concretely realizable acquisition sequence.

A finite decision procedure starts each period from $W^{(0)} = B_\pi \cup S_{<i}$ and iterates

$$W^{(r+1)} = W^{(r)} \cup \big(S_i \cap \mathrm{Frontier}_\pi(W^{(r)})\big).$$

It succeeds if and only if its fixed point contains all of $S_i$. Because satisfying additional prerequisites can never revoke availability of a still-open goal, any batch added simultaneously can also be acquired one goal at a time; conversely, no valid completion witness can fail this

iteration prematurely. The procedure is a *hypothetical* traversal that alters no learner data; on a purely atomic DAG, prerequisite-respecting topological order recovers the familiar special case.

### 12.3 Effort, Reserve, and Budget

Structural feasibility does not yet prove that a school year affords enough time. Every workload claim must state reference learners, entry conditions, target and elective selections, calendar exclusions, estimation method, and uncertainty. Collecting those declared estimation assumptions of plan $\pi$ into $\Theta_\pi$, let $\widehat{t}_h(\Theta_\pi) \geq 0$ denote the estimated duration of an actual activity $h$ under $\Theta_\pi$. All quantities must share one calibrated time unit:

$$\mathrm{Work}_\pi(i;\Theta_\pi) + \mathrm{Reserve}_\pi(i) \leq \mathrm{Budget}_\pi(i), \qquad i = 1, \ldots, n_\pi. \tag{14}$$

Here Work $\geq 0$, Reserve $\geq 0$, and Budget $> 0$. Reserve remains protected and cannot simultaneously count as schedulable capacity; a zero buffer is permissible only when disclosed explicitly. Exam preparation or review already included inside Work must not be double-counted as reserve demand.

A declared activity model may set $\mathrm{Work}_\pi(i;\Theta_\pi) = \sum_{h\in\mathcal{H}_i} \widehat{t}_h(\Theta_\pi)$, where $\mathcal{H}_i$ counts each genuine effort item once: initial acquisition, targeted practice, card recall, assessment, and any other scheduled work. Shared canonical atoms are not counted twice as initial acquisitions across Mathematics and Physics, though dedicated application practice in Physics may legitimately add its own time. Uncalibrated "goal points" or `weight` values are never minute estimates. Table 5 illustrates separate accounting of work and protected reserve.

**Table 5:** Worked illustration: Work = 60 and Reserve = 10 fit inside Budget = 75. The invented figures illustrate the arithmetic only rather than empirically validated durations of any SkillPilot curriculum.

| Illustrative item | Hours | Meaning |
|---|---|---|
| Initial acquisition | 42 | Already deduplicated across shared goals |
| Practice and application | 12 | Additional scheduled activities |
| Assessment and review | 6 | Included inside planned work |
| Protected reserve | 10 | Not available for the work items above |
| Available time | 75 | Net of calendar exclusions |

An aggregate budget alone is insufficient whenever individual subjects, rooms, or timetable slots impose binding caps: under a declared subject allocation, each subject slice must also satisfy $\mathrm{Work}_{i,s} + \mathrm{Reserve}_{i,s} \leq \mathrm{Budget}_{i,s}$ without double-counting shared instructional time, and a dated timetable must schedule prerequisites ahead of their concrete use.

### 12.4 Validity Claims and Completion Boundaries

Under this optional profile, a schedule and acquisition plan is valid when it is both structurally feasible and workload-feasible under its declared assumptions; without calibrated effort estimates, only the structural claim is admissible. Every feasibility claim binds the plan, curriculum revision, resolved scope, and estimation assumptions, without guaranteeing uniform learning speed across individuals.

Completion of a domain learning unit still follows the recursive goal-type and child-unit obligations from Section 11. One calendar period may host several learning units, and one learning unit may

span multiple calendar periods; nothing in this scheduling profile makes the entire preceding year a universal prerequisite for every subsequent goal.

**Sources.** Learning Units; Graph Definition; Curriculum Time Axis, Buffer Planning, and Pacing.

# 13 Implicit Mastery Through Valid Local Assessments

**Binding Concept Revision of 4 November 2026**

An explicit pass on a content goal establishes understanding. Passing a valid local assessment can establish independent application of the underlying competencies. That implicit mastery carries equal global weight: it satisfies subsequent prerequisites and counts toward cumulative attainment. Assessment and memory goals themselves remain strictly explicit.

Consider a learning unit containing the chain Q → B → A: assessment goal Q requires content goal B, and B requires A. As long as Q remains open, the assessment still demands its prerequisites B and A. Once the learner passes Q under a valid local contract of that unit, both B and A count as independently applied – and hence implicitly mastered – even without separate explicit results of their own. The subsections below formalize that set equation and state the domain conditions required before evaluating it.

The Product Owner confirmed this recognition rule on 11 November 2026: an own ordinary goal is recognized once no open own assessment or memory goal of its unit still requires it. If assessments $Q_1$ and $Q_2$ belong to the same unit and both require A, then A stays open after $Q_1$ is passed until $Q_2$ is passed as well (Figure 15); explicit mastery of A remains possible at any time. The rule is sound only inside units that satisfy the validity conditions of Section 13.3.

The stored binary value $m$ remains the sole persisted outcome per goal. This specification derives additional satisfaction without writing mastery backward, adding extra qualification records, introducing manual waiver masks, or requiring second partial exams. On `main`, this concept is **not yet implemented**; the current concept specification is authoritative.

## 13.1 Local Contract, Target Scope, and Protected Goal Types

**Local contract.** A *local contract* belongs to an authoritative learning unit $U$ from Section 11. It binds $U$'s own acquisition targets $\mathrm{Own}(U)$, their canonical identities, the curriculum revision, and the domain-reviewed meaning of $U$'s own assessments; adjacent documents also call it a local recognition or acquisition contract. Total scope $T(U)$ additionally aggregates child-unit targets, yet $T(U)$ is *never* an enlarged substitute recognition scope for the parent unit, and a subsequent parent integration claim never retroactively widens child contracts. Neither a display filter nor a temporary focus nor the currently selected plan roots redefines that ownership.

**Protected goals.** Let $Q \subseteq A$ denote the authoritatively classified assessment goals and $A_{\mathrm{mem}} \subseteq A$ the memory goals, while $K = G \setminus A$ remains the set of structural clusters. A unit's own protected goals are

$$Q_U = \mathrm{Own}(U) \cap Q, \qquad A_{\mathrm{mem},U} = \mathrm{Own}(U) \cap A_{\mathrm{mem}}, \qquad P_U = Q_U \cup A_{\mathrm{mem},U}.$$

$P_U$ protects *all* own assessment and memory goals from implicit completion: passing another assessment substitutes neither for an assessment's own pass nor for verified recall of required flashcards. Orientation remains a distinct motivational goal type rather than a third protected class: an orientation marker may enter $I_U$ as a derivatively closed entry anchor without certifying domain competence or asserting that a motivational dialogue actually took place. Outside support beyond $T(U)$ never enlarges the completion denominator, and competencies outside $\mathrm{Own}(U)$ receive no implicit recognition from $U$'s own contract – child targets are governed by

their respective child contracts. The remaining own goals – content goals and any orientation goals – are called *unprotected* below.

**Underlying relation.** All formulas below evaluate the fully inherited and atomically expanded prerequisite relation $R_{\text{at}}$ from Section 2.3, including hidden and cross-subject support. Every reference must resolve and the normalized relation must be acyclic; full prerequisite closure is evaluated prior to any display restriction.

### 13.2 Transitive Open Demand and Implicit Set

For any subset $X \subseteq A$, let $\text{Reach}^*_{R_{\text{at}}}(X)$ denote $X$ itself together with every atom directly or transitively required by an element of $X$:

$$\text{Reach}^*_{R_{\text{at}}}(X) = X \cup \{b \in A \mid \exists a \in X : (a, b) \in R^+_{\text{at}}\}.$$

Given global explicit mastery $M = \{a \mid m(a) = 1\}$, define

$$N_U = \text{Reach}^*_{R_{\text{at}}}(P_U \setminus M), \tag{15}$$

$$I_U = \text{Own}(U) \setminus (M \cup P_U \cup N_U). \tag{16}$$

Here $N_U$ is *open demand* – the still-open protected goals of $U$ together with their full atomic prerequisite demand – and $I_U$ is the *implicit set*. An own goal belongs to $I_U$ if and only if it is

1. not already explicitly mastered,
2. not protected, and
3. no longer required by any open protected goal of the unit.

Reflexive reachability retains the open roots themselves together with *all* of their transitive prerequisites: traversal **never** stops at an explicitly mastered intermediate goal or at the boundary of $\text{Own}(U)$. Consequently, $I_U$ remains invariant whenever a redundant shortcut edge is added or removed for required transitive minimality, and $I_U \cap M = \varnothing$ as well as $I_U \cap P_U = \varnothing$ hold unconditionally.

Figure 12 works through both starting states of $\mathsf{Q} \to \mathsf{B} \to \mathsf{A}$ with $\text{Own}(U) = \{\mathsf{Q}, \mathsf{B}, \mathsf{A}\}$ and $P_U = \{\mathsf{Q}\}$. In Case A, only $\mathsf{B}$ is explicitly mastered; open assessment $\mathsf{Q}$ continues to demand both $\mathsf{B}$ and $\mathsf{A}$, yielding $N_U = \{\mathsf{Q}, \mathsf{B}, \mathsf{A}\}$ and $I_U = \varnothing$. In Case B, $\mathsf{Q}$ is passed, leaving no open protected goal: $N_U = \varnothing$ and $I_U = \{\mathsf{A}, \mathsf{B}\}$.

**Complexity.** Let $V^{\text{reach}}_U = \text{Reach}^*_{R_{\text{at}}}(P_U \setminus M)$ denote the reachable atomic nodes and $E^{\text{reach}}_U = \{(a, b) \in R_{\text{at}} \mid a \in V^{\text{reach}}_U\}$ their outgoing prerequisite edges. Direct graph traversal visits each such node and edge at most once, computing $I_U$ across $\text{Own}(U)$ in $O(|\text{Own}(U)| + |V^{\text{reach}}_U| + |E^{\text{reach}}_U|)$ steps – and globally in at most $O(|A| + |R_{\text{at}}|)$ – without materializing a quadratic transitive closure.

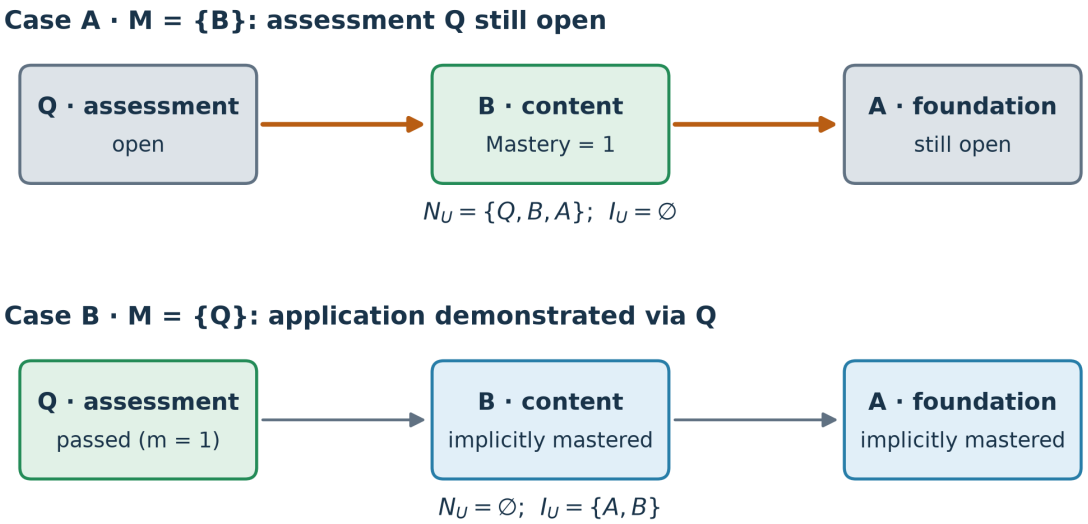



**Figure 12:** Two independent starting states for Q → B → A: with $M = \{B\}$, A remains open; with $M = \{Q\}$, Q carries implicit application of A and B. This is not a temporal sequence: an already explicitly mastered B would stay in $M$ when Q is passed later. A direct edge Q → A would be redundant.

### 13.3 Why the Skill Graph Alone Does Not Prove Application

The set equations above may be evaluated only for a valid domain contract. The Product Owner decision of 11 November 2026 states its conditions as five rules and fixes where each is enforced (Table 6): the recognition rule is sound only if units satisfy them as hard rules.

**Table 6:** Validity conditions of a learning unit and where they are enforced (Product Owner decision of 11 November 2026). No validator enforces any of them on `main` yet.

| Rule | Requirement | Placement |
|---|---|---|
| R1 | Every own ordinary content goal lies on a prerequisite path from at least one own assessment of the unit, directly or transitively; a path that leads only to a memory goal does not count. | Validity condition of the unit definition, in the same class as acyclicity: hard from the first authored unit; a violating unit cannot be merged. |
| R2 | For every own ordinary goal, at least one own assessment that requires it has reviewed evidence that its tasks and normal grading really test the goal. | Milestone `M5` (assessment approval and coverage metadata), with the current per-goal binding at `M7`; without it the curriculum does not reach the milestone. |
| R3 | An assessment claims coverage only for goals it requires directly or transitively: $\mathrm{Cov}(q) \subseteq \{b \in A \mid (q, b) \in R_{\mathrm{at}}^{+}\}$. | As R1. |
| R4 | Every acquisition target has exactly one directly owning unit. | As R1. |
| R5 | A later re-cut of units or later edges never grant additional recognition to existing results; the contract is versioned. | Runtime contract; precondition before implicit recognition is enabled in production. |

**Structural conditions (R1, R3, R4).** Every content goal in $\mathrm{Own}(U)$ must lie on a local prerequisite path from at least one local assessment in $Q_U$: an own assessment of the unit must require it directly or transitively (R1). A path leading only to a memory goal never suffices, because verified card recall does not prove independent application. An isolated terminal content goal without a local assessment consumer is a contract defect rather than zero-work mastery; lacking a downstream consumer never proves competence. No ad-hoc mask of "recognition-eligible" subgoals exists – the formula governs the entire valid local contract.

Let $\mathrm{Cov}(q)$ denote the goals whose independent application an assessment $q$ claims to test according to its reviewed coverage claim. R3 requires $q$ to require each of them directly or transitively. This guarantees that the assessment which actually tests a goal is among the open protected goals the recognition rule waits for: while an own assessment $q$ is open, every own goal in $\mathrm{Cov}(q)$ stays in $N_U$. A missing path is a finding for subject review, not a reason to add a redundant direct edge. R4 makes the contract well defined: a goal with no or with several directly owning units belongs to no valid local contract.

**Content condition (R2).** For every own ordinary goal, at least one own assessment that requires it must carry reviewed evidence that its authored tasks and normal grading rubric genuinely require independent application of that competence (R2), without reinterpreting an orientation marker as an assessable domain competence. Essential competencies must not be wholly omitted while still allowing unrelated points elsewhere to cross the passing threshold. Neither a local `requires` path nor a title, `coveredGoalIds` array, passing threshold, or formal DAG check proves that property: independent domain review inspects tasks, model solutions, rubrics, and claimed application coverage jointly across every passing assessment combination that releases a target under the set formula – checking only the union of all assessments never suffices.

**No second status.** Passing an assessment remains the single ordinary binary pass of that assessment goal. Stronger application meaning resides in the pre-reviewed local contract rather than in an extra per-learner qualification field or a second follow-up exam, and a legacy assessment lacking that review never acquires expanded meaning retroactively from a new diagram or software update.

**Consequence.** Implicit recognition exists only inside units whose contract satisfies R1–R4 with reviewed coverage. Otherwise the unit's own ordinary goals stay open – they can still be mastered explicitly – and the violation keeps the curriculum from reaching the respective milestone (Table 6). A goal that no assessment covers therefore stays open *and* is a rule violation; it is never recognized for free. Section 13.4 treats R5.

### 13.4 Global Effectiveness and Durable Contract Binding

Let $\mathcal{U}_{\mathrm{valid}}$ denote the authoritative local contracts durably bound to this learner at their applicable revisions – rather than arbitrary visited views or all latest content revisions:

$$M_{\mathrm{eff}} = M \cup \bigcup_{U \in \mathcal{U}_{\mathrm{valid}}} I_U.$$

That union is not restricted to currently displayed learning units or the active Personal Curriculum view: contract identity, revision, and evidential meaning must remain evaluable across scope

changes. Derivation always runs directly from explicit outcomes $M$ and valid contracts without feeding previously derived implicit sets back as explicit mastery inputs.

A later learning unit U2 carrying another assessment $q_2$ must never reopen an application competence a already established under U1 (Figure 13); instead, a satisfies $q_2$'s prerequisite. Under a fixed contract set, adding explicit mastery can only enlarge $M_{\text{eff}}$: open protected roots shrink, their transitive demand contracts, and any newly explicitly mastered goal leaving $I_U$ stays inside $M_{\text{eff}}$ via $M$.

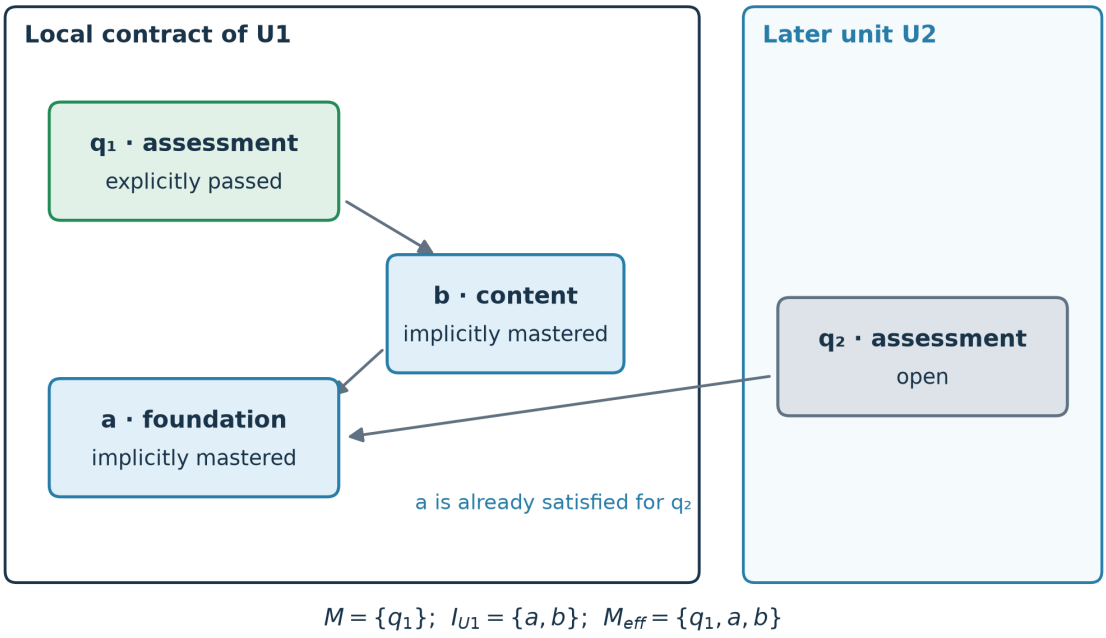



**Figure 13:** U1 recognizes a and b under its own valid contract. The open later assessment $q_2$ does not block that recognition. Switching focus or plans does not erase the still-evaluable contract of U1.

A stable assessment goal ID binds a stable recognition meaning, and equivalent task variants must uphold that same qualified contract. Expanded recognition claims must never silently upgrade legacy exam passes: they require either a new assessment identity or an explicitly reviewed meaning-preserving migration, without adding per-learner qualification records. Whenever historical evidential meaning is unresolved, the explicit pass stands without fabricating broader application recognition, and genuine score or content corrections remain distinct from ordinary downstream consumer or plan switches.

Under rule R5, a later re-cut of units or later edges never grant additional recognition to existing results; the contract is versioned. This runtime contract is a precondition before implicit recognition is enabled in production.

### 13.5 Protected Goals and Full Unit Completion

**Recognition is not yet full completion.** Full unit completion follows recursively from Equation (10): all own goals must be satisfied and every required child unit must be complete. Given global $M_{\text{eff}}$, that condition coincides with $T(U) \subseteq M_{\text{eff}}$ provided every required child obligation is represented inside $T(U)$, and full completion always entails $P_U \subseteq M$ on own protected goals.

Once every own assessment and memory goal of $U$ is passed, $N_U = \varnothing$, allowing $U$'s valid local contract to complete its own unprotected goals. Completion of required child units **never** follows from that fact alone: a parent synthesis assessment may stay open while child content competencies are already globally recognized, and conversely passing the parent synthesis never satisfies an open child assessment or memory obligation.

**Open protected goals in the same contract.** An open own memory goal blocks full unit completion without blocking implicit recognition of another own content goal that it does not transitively require (Figure 14). Whenever an open memory goal does require a content goal directly or transitively, however, that content goal stays in open demand. Likewise, when multiple assessments inside the same local contract share a common foundation, all of them must be passed before that foundation leaves open demand.

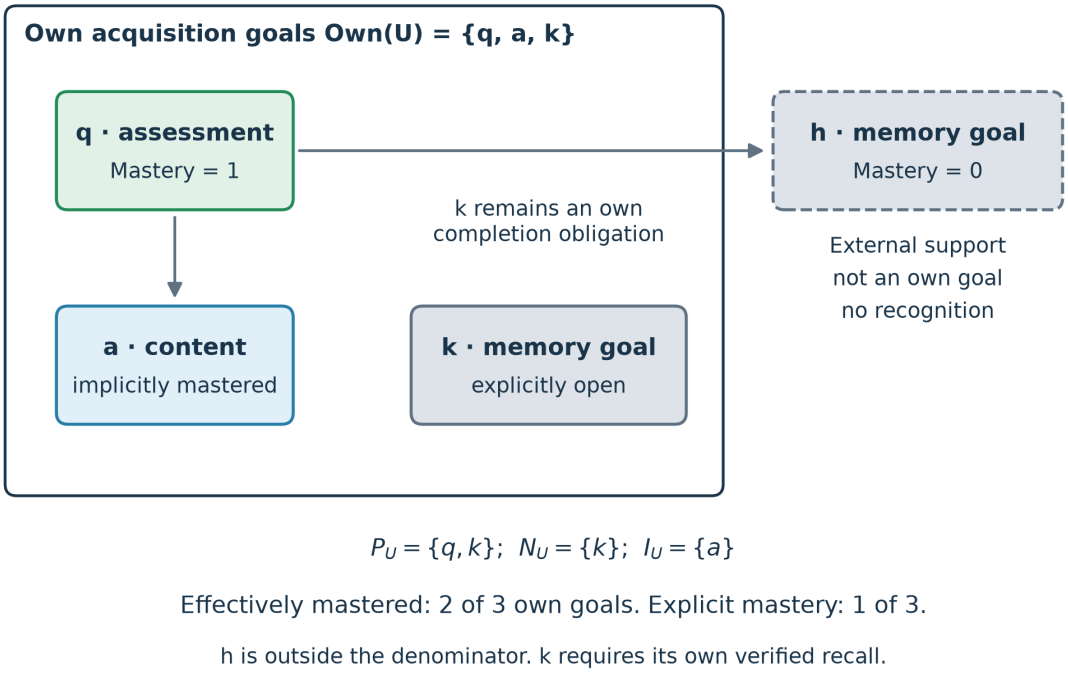



**Figure 14:** Assessment q is explicitly passed and carries application of a. The independent memory goal k remains open and blocks full unit completion. External support h outside Own($U$) receives no recognition.

That rule distinguishes multiple assessments inside the *same* local contract from a subsequent assessment in another learning unit (Figure 13). Figure 15 traces two successive passes at unchanged local scope without further protected obligations.

Comparing that case with Figure 13 presupposes domain-justified, independently valid local contracts: an open assessment in a later unit never revokes recognition already established in an earlier unit, whereas inside a single contract the entire remaining open demand applies. Nothing here licenses splitting or rearranging assessments or units merely to engineer a desired calculation result.

**Assessment access.** Existing direct assessment access remains the baseline of the current learner workflow, subject to task availability, authentication, session ownership, and explicit learner selection. Whether optional gated profiles later impose hard unit locks – and how deliberate

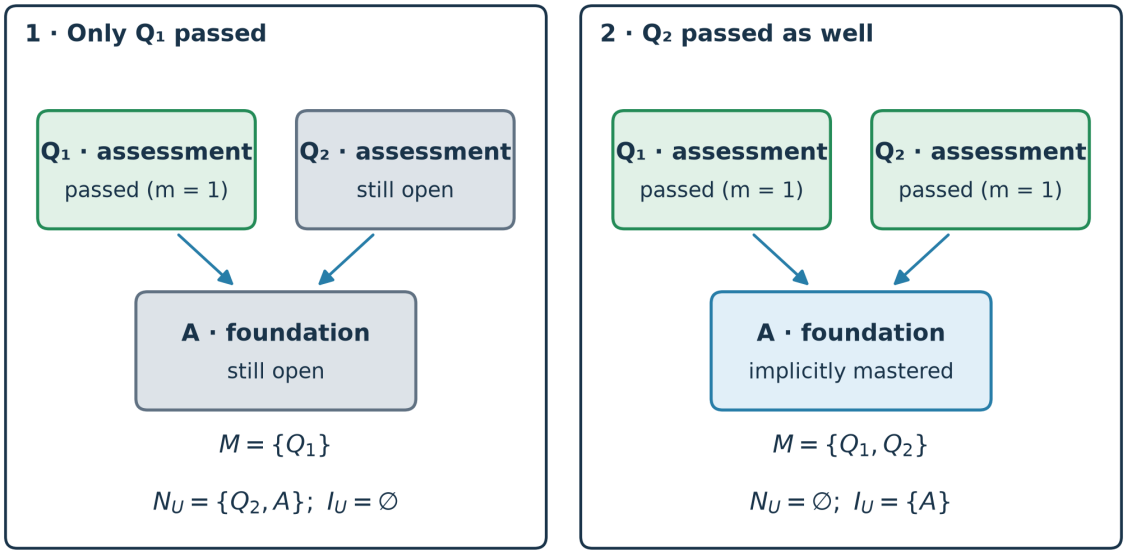



**Figure 15:** $Q_1$ and $Q_2$ both require A in the same valid local contract. After $Q_1$ alone, A stays in open demand because $Q_2$ is still open. Only after both assessments are passed does A become implicitly mastered globally. The explicit pass of $Q_1$ is preserved; there are no further protected goals here.

direct assessment attempts behave under them – remains undecided (Section 11); recursive unit definitions introduce neither a hidden lock nor an unreviewed bypass. Ordinary prerequisite evaluation uses $M_{\text{eff}}$ over the transitive effective relation while preserving normal and strict focus semantics.

### 13.6 Cumulative Attainment and Dated Period Events

For a fixed named plan target set $T_{\text{Plan}}$, cumulative plan attainment counts the effectively mastered goals of the plan:

$$H_{\text{gesamt}} = |T_{\text{Plan}} \cap M_{\text{eff}}|.$$

Forming the union $M_{\text{eff}}$ prior to intersecting with $T_{\text{Plan}}$ prevents double-counting across explicit outcomes, implicit sets, and overlapping contracts; any dedicated explicit-only view must label its distinct counter clearly.

By contrast, an active schedule period counts solely dated explicit completion events under its event contract: newly derived application recognition, prior mastery brought into the course, or a widened plan display never fabricates an extra dated completion event inside the current period (Figure 16).

The unified learning-plan status contract specifies the user-facing surfaces. Static contract definitions may be cached per revision while derived sets remain computed projections rather than a second stored truth, and invalid or unresolved contracts stay visibly unavailable. Automated




**Figure 16:** Total plan progress and period completions use distinct counters. In the example, 7 of 10 planned goals are effectively mastered (4 explicit and 3 implicit). The current period counts only the 2 dated explicit completions against its quota of 3 goals. The three implicit contributions raise cumulative progress without fabricating dated explicit completion events inside the current period.

checks of these formulas replace neither independent content review nor human learner testing or instructional effectiveness studies.

**Sources.** Assessment-Gated Completion and Implicit Mastery; Completion Semantics Clarification of 4 November 2026; Unified Learning-Plan Status Contract.

## A Rules and Milestones

This appendix is the reference register for all 145 skill-graph rules (`GD-*`) in this document. For every rule, it answers three questions: what is required, which validation area owns the obligation, and what the existing checks actually prove.

**How to read a row.** Each `GD-*` row links an obligation to its designated validation area (*target*), existing checks (*actual*), and their evidentiary limits: **ID** (`GD-`, the document-part letter, and a sequential number: `C` for the formal core, `X` for context, `U` for learning units, `P` for planning, `A` for implicit-mastery recognition, and `D` for the document appendices); **Traffic light** (technical enforcement level on the mainline: `HARD` for rules enforced fail-closed across their entire declared domain, `PARTIAL` for rules hard-gated partially or via profiles or maturity floors, `OPEN` for binding obligations without an established hard gate on the core claim, and `OPT/DOC` for optional profiles, undecided policy, or document checks); **Obligation** (concise statement of the rule); **Target** (the validation area assigned to the obligation: a milestone `M0`–`M7` or one of the area codes in Section A.2); **Actual and gap** (what existing checks prove and what remains open); **Section** (the named section where the rule is defined). Rows use concise telegraphic style and recurring shorthand: *Unit* for learning unit, *Exam* or *Assessment* for an exam or assessment goal, *Memory* for a memory goal or memorization, *Normal* and *Strict* for normal mode and strict mode, *Closure* for transitive closure, *Scope* for validation or applicability scope, *Runtime* for runtime execution, *unit-wide lock* for holistic unit gating, and *human trial* for learner evaluation.

**Milestones and rule families.** Milestones `M0`–`M7` structure automated curriculum QA; runtime contracts, concepts, optional profiles, and artifact or document obligations are tracked separately. The two existing rule families are *Curriculum Quality Review* (`CQR`) and *graph-validation rollout rule* (`GVR`), denoting curriculum quality checks and skill-graph validator rollout rules respectively. The legend in Section A.2 defines every code referenced here along with its actual scope.

**An assignment is not a pass.** Target assignment names the validation area responsible for each obligation; the existing gates listed alongside prove only their actual scope. A binding rule without a complete gate is an open verification obligation, not an already passed check. `M0` is a possible incomplete starting state, not a quality seal. Profile-bound route gates apply within their declared QA scopes. By contrast, the global base check for complete atomic normal form and transitive minimality (check code `G17`) must cover the entire declared canonical subject inventory, including relevant cross-subject prerequisites and inheriting ancestors; a QA profile subset or partial scan is insufficient.

**Base before every milestone.** Complete atomic normal form and a full transitive-minimality check are the required target base before any promotion to `M1`–`M7`. A `G17` violation or an incomplete check blocks every milestone pass `M1`–`M7` in the affected subject, even though individual source or review checks may still pass on their own. The atomic target state $R \subseteq A \times A$ is binding: first establish or retain complete atomic prerequisites, next remove the non-atomic requirements rendered redundant by them, and only then transitively reduce the fixed atomic relation. An inherited alternative path alone never justifies deleting an atomic route. `CQR-102` and `GVR-012` remain directly atomic.

**Memory goals.** Flashcard repetitions belong to the learning process. Completing a memory goal requires its own explicit verification record; subsequent review due dates alone do not revoke that completion. Ordinary completed goals receive neither review due dates nor any mastery revocation derived from them. No new assessment entity or schema is introduced by this contract.

**Scope of the register.** The 145 stable `GD-*` IDs cover all six parts of the compendium. Section references use named sections rather than volatile line numbers. Runtime contracts, new recognition and learning-unit semantics, optional planning, and publication or document obligations are never assigned a fabricated `M7` gate; where the Product Owner fixes a target area, such as `M5`/`M7` for the coverage evidence of learning units (R2), the row states that it is not yet enforced. Subject-matter accuracy, human trials, and learning effectiveness remain independent forms of evidence.

## A.1 Traffic-Light Status: What Is Truly Hard-Enforced Today

Many of the 145 `GD-*` rule groups bundle a technically gated structural core together with broader runtime, migration, or subject-matter obligations. The four-state traffic light therefore makes visible—both per individual validator gate and per rule group—where a technically hard (fail-closed) gate already operates today and where scope limits or open obligations remain.

| Traffic light | Meaning for enforcement and CI |
|---|---|
| ○○● HARD | **Fully hard-enforced.** The entire core claim is technically gated fail-closed across its declared domain; no material sub-obligation of the row remains open. Globally enforced in skill-graph, schema, UUID, and composition validators (`validate:graph`, `validate_schemas.py`, `validate_goal_ids_uuid.py`, `validate:composition-views`): JSON/schema, UUID format, resolvable references, self-reference protection, separate DAGs (`contains`, direct/inherited `requires`), `GVR-001/003/008/009/010/014/016`, and duplicate, single-parent, and non-emptiness rules for compiled views (`CPV-001..008`). Fully covered rule groups: **GD-C05**, **GD-C07**, **GD-X07**. |
| ○●○ PARTIAL | **Partially, profile-, or floor-bound hard-enforced.** A named subset fails closed on violation, but applies only to protected subjects/profiles or covers only the structural or artifact share. Maturity floors (`check:curriculum-maturity-floors`) and status-file diff checks block regressions for **ten Gymnasium subjects** (5 at `M7`: Mathematics, Physics, Chemistry, Biology, Economics; 5 at `M6`: German, History, Computer Science, Latin, Politics and Economics); since 11 November 2026, `CQR-001` additionally applies the `G17` check (atomic normal form and transitive minimality) to every canonical Gymnasium root, hard via these floors (**GD-C11, C15–C17, U13**); `GVR-012/013` are hard-enforced in Mathematics, `GVR-006` in the Hesse rollout, and `validate:view-filters` (`APV-102/103`) for canonical Gymnasium views; complemented by the learning-scope closure compiler $L$ (10 November 2026) as well as content-package, license, package, and release checks (**54 groups**). |
| ●○○ OPEN | **Binding, core obligation still without a hard gate.** Normatively binding target, runtime, or concept rule without a cross-cutting fail-closed gate established in this register. Atomic expansion before closure in result derivation (**GD-C13**), open cluster atomization in `goalEntry` (**GD-X09**), cluster satisfaction across all atoms instead of `isCoreForPrereqs` (**GD-C26**), not-yet-implemented global implicit-mastery recognition including the learning-unit validity rules R1–R5 of 11 November 2026 (**GD-A01–A18, C25, C30, X18, X32, X36**), recursive `LearningUnits` including their purpose and boundary criterion (**GD-U01–U12, U15**), and pure runtime, dialogue, and edge-provenance contracts (**60 groups**). |

| Traffic light | Meaning for enforcement and CI |
|---|---|
| OPT/DOC | **No production mandatory gate.** Deliberately optional planning, explicitly undecided lock policy, or pure document/model verification outside curriculum and runtime gates. Optional acquisition and time planning (**GD-P01–P11, X33–X35, X37**), undecided unit-wide access lock (**GD-U14**), and appendix, model-checker, and PDF-build rules (**GD-D01–D12**; build and model scripts check fail-closed locally, while pure `docs/**` changes are exempted in `ci.yml`; **28 groups**). |

**Total register:** 3 `HARD`, 54 `PARTIAL`, 60 `OPEN`, 28 `OPT/DOC` (total: **145 rule groups**).

## A.2 Overview of Validation Areas

| Code | Status | Meaning |
|---|---|---|
| **M0 Base, before M1** | PARTIAL | **Obligation:** Identity, references, structure, and base graph; complete atomic normal form and transitive-minimality check as the target base before every higher milestone pass; validity of every authored learning unit (R1, R3, R4). **Actual limit:** Schema and skill-graph validators check core invariants hard across all loaded skill landscapes; `CQR-002` checks explicit types of canonical Gymnasium roots; since 11 November 2026, `CQR-001` adds the full `G17` check for every root of the complete canonical Gymnasium catalog (a violation caps the subject at `M0`; hard in CI via the ten protected floors). Non-canonical roots have no `G17` gate; no unit validator (R1, R3, R4) exists. |
| `M1` | PARTIAL | **Obligation:** Readable, traceably registered source inventory. **Actual limit:** `CQR-000` hard-gated via the ten protected floors and status diff; does not yet prove subject-accurate mappings. |
| `M2` | PARTIAL | **Obligation:** Bidirectionally source-evidenced views, course-level mapping, and goal-count plausibility. **Actual limit:** `CQR-003/004/005` hard-gated via protected floors, each subject to its actual aggregator status logic. |
| `M3` | PARTIAL | **Obligation:** Additional complete directly atomic orientation-to-autonomy routes within the named scopes. **Actual limit:** `CQR-101/102/103` across twelve profiles; `GVR-012/013` only in Mathematics profiles. Does not fully gate the global target base. |
| `M4` | PARTIAL | **Obligation:** Visible route endpoints and concrete terminal assessment tasks. **Actual limit:** Actually `CQR-104/201/202` via protected floors; catalog target of `104` is `M3`. Does not prove live session grading conformance. |
| `M5` | PARTIAL | **Obligation:** Assessment approval/coverage metadata, current atomicity reviews, composition views, and cleared applicability warning debt; reviewed application coverage of every own ordinary unit goal (R2). **Actual limit:** Actually `CQR-203/301/401/501` plus composition and filter validators; catalog target of `203` is `M4`, and view file counting alone does not prove full domain sign-off. R2 is not checked: no units are authored. |
| `M6` | PARTIAL | **Obligation:** Justified memory-support decisions, traceable cards/decks, and required view visibility. **Actual limit:** `CQR-302` across all ten protected `M6`/`M7` floors plus card-review check; makes no claim about personal due dates or mastery revocation. |

| Code | Status | Meaning |
|---|---|---|
| M7 | PARTIAL | **Obligation:** Simultaneously current D/P/A/M/V evidence for all current curricularAtomic goals, all mandatory checks passed, and zero blockers; current per-goal binding of the R2 coverage evidence. **Actual limit:** CQR-303 on five M7 floors atop all lower stages; implies neither human sign-off, learner trials, learning effectiveness, nor general runtime conformance. None of the six required checks covers R2. |
| **Cross-Stage** | PARTIAL | **Obligation:** Learner state, satisfaction, frontier, Personal Curriculum, roles, focus, goal-type, and persistence contracts; versioned recognition contracts (R5). **Actual limit:** Closure compiler (10 November 2026), seed-scope fallback for non-compilable contexts (11 November 2026), and view/role checks established; remaining runtime and interaction proofs stay separate, with no blanket curriculum M-gate; R5 is a precondition before recognition is enabled. |
| **Concept / Optional** | OPEN<br>OPT/DOC | **Obligation:** New global implicit-mastery recognition and recursive learning units (OPEN), plus optional time/budget planning and undecided unit access (OPT/DOC). **Actual limit:** Derive neither implemented new semantics nor a hard access lock from M7. |
| **Artifact / Document** | PARTIAL<br>OPT/DOC | **Obligation:** Interchange, publication, licensing, data protection (PARTIAL), and document sign-off (OPT/DOC). **Actual limit:** Package, license, and release checks run in CI; a local document build is neither publication nor host acceptance. |

## A.3 Gate Reference and Actual Scope

Stable codes name individual existing checks (maturity and route documentation, graph validation rules). The legend explains every code; the subsequent assignment key maps their actual effect onto the milestones. The legend covers the exact eight GVR and eighteen CQR codes cited in this register; additional validator rules are neither introduced nor claimed as verified here.

| Gate | Status | Actual scope |
|---|---|---|
| GVR-001 | HARD | A goal must not directly require any of its own contains ancestors; local skill landscape (all loaded skill landscapes). |
| GVR-003 | HARD | A goal must not directly require its direct contains parent node; local skill landscape (all loaded skill landscapes). |
| GVR-006 | PARTIAL | A goal must not directly require any of its direct contains children; named Hesse rollout inventory. |
| GVR-009 | HARD | A stored structural node type must match its classification as leaf or cluster; local skill landscape (all loaded skill landscapes). |
| GVR-010 | HARD | An optional shortKey must be unique within a logical skill landscape; language editions of the same goal may share the same ID/shortKey pair. |
| GVR-012 | PARTIAL | Hard complete direct atomic route for every selected subject atom between same-stage orientation and terminal autonomy; canonical Mathematics lower and upper secondary separately. |
| GVR-013 | PARTIAL | Direct local prerequisites of selected non-memory atomic sources must target atomic goals; configured Mathematics profiles only, not a global edge check. |

| Gate | Status | Actual scope |
|---|---|---|
| GVR-014 | HARD | Orientation goals are prerequisite-free atoms without exam, memory, example, or assessment metadata; all loaded skill landscapes. |
| CQR-000 | PARTIAL | Source inventories are readable, linked to official HTTP(S) sources, and have their extracted goals registered. |
| CQR-001 | PARTIAL | Since 11 November 2026: every declared root of the complete current canonical Gymnasium catalog, with its required cross-subject goals and inheriting ancestors, passes the shared reference and effective-DAG audit, atomic `requires` normal form, explicit type consistency, and transitive minimality (`G17`). |
| CQR-002 | PARTIAL | Stored structural node types match the atomic or cluster structure. |
| CQR-003 | PARTIAL | Jurisdiction views have atomic source evidence and cover all registered original source goals completely; checked bidirectionally. |
| CQR-004 | PARTIAL | Upper-secondary source goals match the `GK/LK` mapping of canonical goals; unspecified course levels apply to both `GK` and `LK` unless a reviewed `LK` exception exists. |
| CQR-005 | PARTIAL | Extracted goal counts are plausible against the reviewed `HE/BW` baseline or have explicitly reviewed deviations. |
| CQR-101 | PARTIAL | Complete routes across effective (including inherited) prerequisites connect motivation and terminal autonomy; configured QA scopes. |
| CQR-102 | PARTIAL | Complete routes across directly authored atomic prerequisites connect motivation and terminal autonomy; configured QA scopes. |
| CQR-103 | PARTIAL | Clusters selected within the QA scope carry no `requires`; not a global proof across all authored edges. |
| CQR-104 | PARTIAL | Motivation and terminal autonomy endpoints are visible in the relevant learner-facing composition views; scopes configured accordingly. |
| CQR-201 | PARTIAL | Terminal autonomy goals of the configured scopes carry `examData`. |
| CQR-202 | PARTIAL | Their exam tasks are concrete and usable in assessment mode rather than placeholders. |
| CQR-203 | PARTIAL | Their assessment metadata specifies approval status, covered goals, content areas, and cognitive demand level. |
| CQR-301 | PARTIAL | Configured atomicity reviews are complete, current, and free of open review findings. |
| CQR-302 | PARTIAL | Memory-support decisions, required flashcards, and decks are traceable; required memory goals are visible in the configured views. |
| CQR-303 | PARTIAL | For a non-empty current inventory of `curricularAtomic` goals, `D/P/A/M/V` evidence is jointly present, all mandatory checks pass, and zero blockers exist; automated QA, not human sign-off. |
| CQR-401 | PARTIAL | Composition-view files exist; actual file counting does not prove complete domain sign-off of view contents. |
| CQR-501 | PARTIAL | Active applicability warnings are cleared and accepted warning waivers match the current findings. |

## A.4 Assignment Key

Register rows name validation areas using the eleven codes below. Each entry explains a code together with its existing checks and their evidentiary limits.

**BASE.** CQR-001 (references, effective DAG, atomic normal form, explicit types, and transitive minimality, i.e. G17, over the complete canonical Gymnasium catalog since 11 November 2026) and CQR-002 (explicit structural node types) carry catalog target M0. `validate:graph` runs additional standalone GVR and integrity checks across all loaded skill landscapes. Unit validity (R1, R3, R4) is assigned here but has no validator yet. This does not imply complete verification of every mathematical or runtime rule.

**SOURCE.** CQR-000 → M1; CQR-003/004/005 → catalog target M2. Covers source inventory, view coverage, course-level mapping, and count plausibility; the concrete aggregator status logic is not equally strict across all checks. Subject-matter accuracy does not follow from technical evidence checks alone.

**ROUTE.** CQR-101/102/103 → M3, configured QA scopes only. CQR-102 uses exclusively directly authored atomic edges; CQR-103 forbids `requires` on selected clusters. GVR-012/013 are additional scoped profiles, not a global completeness proof. CQR-104 carries catalog target M3 but blocks only M4 in the actual curriculum promotion chain.

**EXAM.** CQR-201/202/203 carry catalog target M4. 201/202 are actual prerequisites for curriculum M4; 203 takes effect only together with core reviews at M5. Metadata and task-readiness checks prove neither accurate live-session grading nor independent application coverage across every passing score combination.

**VIEW.** CQR-401 and CQR-501 → M5. 401 counts existing view files rather than proving their complete domain sign-off. Standalone `validate:composition-views` and `validate:view-filters` gates check further, individually bounded structural and projection obligations. Never equate these blanketly with all runtime rules.

**REVIEW.** CQR-301 → M5 (current semantic atomicity decisions), CQR-302 → M6 (memory decisions, cards, and visibility), CQR-303 → M7 (complete current D/P/A/M/V intersection with required checks passed and zero blockers). M7 remains automated curriculum QA.

**RT.** Cross-stage runtime and interaction contract; no curriculum M-gate is designated for this overall claim. Individual implementations and tests must be evidenced separately; their mere existence is not a live conformance check.

**CONCEPT.** Normatively specified, but new global implicit-mastery recognition and recursive learning-unit semantics are explicitly not yet implemented in production. No existing curriculum M-gate. Structural or content shares may be mapped to target validation areas for future implementation without fabricating M7 fulfillment.

**OPTIONAL.** Optional planning profile or still-undecided holistic unit-access policy. No curriculum M-gate; do not derive a new hard runtime lock from the specification.

**ARTIFACT.** Standalone publication, interchange, licensing, or data-protection contract; no automatic M0–M7 assignment. Any specific package or release check must be evidenced within its own declared scope.

**DOCUMENT.** Document build, mathematical formula model, or diagram check; neither a curriculum M-gate nor proof of runtime conformance or learning effectiveness.

### A.5 Actual Promotion Chain and Additional Gates

Actual maturity promotion is governed by `deriveScopeMaturity` and `deriveCurriculumMaturity` in the status generator; catalog metadata alone never computes maturity. The actual promotion chain requires `CQR-104/201/202` only at curriculum `M4` and `CQR-203` together with core reviews at `M5`; distinguish these from the catalog targets of `104` (`M3`) and `203` (`M4`).

- **M0 → M1:** `CQR-001/002` and `CQR-000` must be `pass`. Both base graph rules carry catalog target `M0`; since 11 November 2026, `CQR-001` includes the `G17` check, whose violation also caps every scope at `M0`.
- **M1 → M2:** `CQR-003` must be `pass`; `CQR-004` blocks on `fail`, `CQR-005` on `warn`. `not_configured` on either of those two rules does not block independently in the aggregator.
- **M2 → M3:** Route profiles must exist; `CQR-101/102/103` must be `pass` in every configured scope.
- **M3 → M4:** Configured `CQR-104` and `CQR-201/202` must be `pass` in every scope. `CQR-104` carries catalog target `M3`.
- **M4 → M5:** All scoped `CQR-203` checks plus `CQR-301/401/501` must be `pass`. `CQR-203` carries catalog target `M4`; `CQR-401` counts registered view files.
- **M5 → M6:** Additionally `CQR-302` must be `pass`.
- **M6 → M7:** Additionally `CQR-303` must be `pass`: non-empty current `curricularAtomic` denominator, complete `D/P/A/M/V` intersection, all six mandatory checks passed, and zero blockers.
- Missing route profiles cap an otherwise source-ready curriculum at `M2`. `CQR-004` blocks in the aggregator on *fail* and `CQR-005` on *warn*; `not_configured` on either does not block independently there. `CQR-401` counts view files. Source coverage binds existing parseable registered snapshots rather than proving exhaustive extraction of official PDFs on its own.
- Routes are evaluated across twelve explicit profiles for ten curricula: Mathematics and Physics each in lower and upper secondary separately; Chemistry and Biology each in upper secondary; Economics, Politics and Economics, Computer Science, History, German, and Latin each cross-stage. `GVR-012/013` apply only within their Mathematics profiles (`GVR-012` remains an error even in legacy warn mode); `GVR-013` covers direct local atomic sources for non-memory atoms rather than all subjects, cluster sources, or cross-subject edges. `CQR-103` checks only the selected clusters. The atomic target state $R \subseteq A \times A$ is gated by `CQR-001` (canonical Gymnasium catalog only, hard via floors); the ordered migration is evidenced only for reductions bound to the prerequisite-reduction proof registry.
- The transitive-minimality auditor also runs standalone with CLI and contract tests (check mode fails on findings, whereas report mode can emit findings with exit code `0`); CI runs its tests (`test:transitive-minimality`). Since 11 November 2026, its zero-finding result per canonical Gymnasium root, together with atomic normal form, is part of `CQR-001` through the graph-integrity adapter; wherever `CQR-001` passes, the historical raw-edge criterion coincides with atomic minimality. Before emitting status rows, the generator verifies the prerequisite-reduction proof registry against the complete current canonical catalog and aborts on any deviation.
- Commands `quality:curriculum-status`, `quality:curriculum-status:check`, and `check:curriculum-maturity-floors` are registered in package.json. The status producer computes and writes the report; domain warnings or failures lower maturity without changing its exit code on their own. The CI workflow then runs the floor checker against the floor policy and verifies a clean git diff of generated status files. Floor checking trusts the numerically

computed maturity on a single matching subject entry and computes no independent CQR/G17 verdict. Five curricula are protected at M7 and five at M6; experimental inventories without a floor remain reportable at M0. Standalone graph, route, source, filter, and composition checks act as additional CI gates rather than extra CQR rules.

Workflow triggers are manual dispatch, push to main, and pull request to main under path filters; pure documentation changes are generally exempted while generated QA status files are re-included. No single CI or build result on its own proves all runtime, subject-matter, and learning-effectiveness rules.

## A.6 C – Formal Core

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-C01 PARTIAL §1.1 | Finite learning-goal graph $(G, C, R)$; contains and requires are the sole authored goal edges. Inverse and non-empty transitive closure are distinct operations. DAG checks apply to the respective named relations rather than blanketly to $C \cup R$. | BASE (M0 base); semantic definition cross-stage | BASE, GVR-008, and schema enforce finite goal sets, permitted edge fields, and separate DAGs; mathematical operation semantics is not a standalone CQR pass. |
| GD-C02 PARTIAL §1.1 | contains: parent goal → child; requires: dependent goal → prerequisite. Any learning-flow diagram drawn in the reverse direction must explicitly declare the inversion. | BASE (M0 base) / presentation cross-stage | BASE/ROUTE consume the edge direction; diagram labeling is not a CQR check. |
| GD-C03 PARTIAL §1.2 | Canonical UUID identity is injective and persists across language, placement, and view; local formula letters do not define an alternative storage format. | BASE (M0 base); identity preservation additionally RT/ ARTIFACT | BASE plus UUID and view validators enforce ID format and uniqueness, but do not on their own prove all version and migration guarantees. |
| GD-C04 PARTIAL §1.2 | Base attributes are total: ID, title, and strictly positive weight. description must be intelligible and verifiable; examples are illustrative references, neither quality proof nor competency evidence. | BASE (M0 base) for form/data domain; M7 for current content evidence | Schema enforces required fields and REVIEW (on M7 floors) enforces current content evidence; strictly positive weight and prose quality are not globally gated across all inventories. |
| GD-C05 HARD §1.2 | An optional shortKey is unique per logical skill landscape; localized language files for the same goal may reuse the same ID/shortKey pair. | BASE (M0 base) | Hard-enforced via GVR-010 in validate:graph across all loaded skill landscapes, including the localization boundary. |
| GD-C06 PARTIAL §1.2 | phase is compatibility/display metadata, neither identity nor sequencing. area/tags never replace authoritative goal-type or scope decisions; level denotes difficulty; core does not on its own determine the selected target scope; weight denotes progress share rather than duration or evidentiary strength; new optional materials belong in content packages. | Cross-stage model contract; M2/M5 for respective mapping/view evidence; material contract ARTIFACT | Content-package validator and GVR-008 enforce material separation; no single gate checks every field's semantic boundary. For sources and materials see GD-X38–GD-X43. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-C07 ○○● HARD §1.3 | Structurally atomic iff a goal has no `contains` children; otherwise a cluster. Polyhierarchy is permitted; explicit structural types must match. Pedagogical goal kinds are neither new structural types nor program units. | `BASE` (M0 base) | Structural type, leaf/cluster classification, polyhierarchy, and separation from program units are hard-enforced globally via `GVR-009`, `CQR-002`, and schema; semantic atomicity is governed separately by `GD-C08`. |
| GD-C08 ○●○ PARTIAL §1.3 | A structural leaf is not automatically semantically atomic; bundling multiple independent sub-competencies can fail content review. | M5, deeper current binding M7 (`REVIEW`) | Hard-protected via `CQR-301` and `CQR-303` across the ten M6/M7 floors; unprotected inventories remain below, and structural leaf status alone is insufficient. |
| GD-C09 ○●○ PARTIAL §2.1 | `Anc`/`Desc` follow non-empty `contains` paths; Atoms$(a) = \{a\}$ for atoms and atomic descendants otherwise. In a finite DAG every cluster has leaf descendants; set semantics counts multi-paths once. Childless means atom, never empty cluster. | `BASE` (M0 base); `RT` for evaluation | `BASE` proves structural invariants, not every runtime aggregation path. |
| GD-C10 ○●○ PARTIAL §2.2 | Multiple prerequisites apply conjunctively. Alternatives require an explicit selection contract; $R$ is acyclic. | `BASE` (M0 base); selection contract separate | `BASE` checks DAG acyclicity; there is no tacit OR semantics. |
| GD-C11 ○●○ PARTIAL §2.2; §2.3 | In the target state, all domain prerequisites are authored strictly between atomic learning goals: $R \subseteq A \times A$. Cluster inheritance remains a transitional compatibility mechanism and never substitutes for complete atomic modeling. Never delete an atomic route in favor of an inherited cluster representation. | `BASE` (M0 target base, before M1): necessary prerequisite for the full `G17` check; never deferred to M3 | Since 11 November 2026, `CQR-001` reports every authored requirement with a non-atomic source or prerequisite for every canonical Gymnasium root and its required closure; a finding caps the subject at `M0`. CI fails on it only via the protected floors; other canonical subjects report `M0`, and non-canonical roots are not checked. `ROUTE` additionally checks scoped profiles. |
| GD-C12 ○●○ PARTIAL §2.3 | Effective direct requirements equal a goal's own requirements united with those inherited from all `contains` ancestors; multiple parents yield set union. Inheritance and transitive closure are distinct operations. | `BASE` (M0 base for valid transition inventory); `RT` | `validate:graph` checks effective cycles; that alone does not prove all runtime evaluations. |
| GD-C13 ●○○ OPEN §2.3 | Expand effective requirements to atoms first, then form the transitive closure; further prerequisites of expanded atoms remain included. Mastering an intermediate goal never truncates the closure. | `BASE` (M0 target base for normalized validity); `RT`/ `CONCEPT` for result derivation | Not blanketly covered by `CQR-001` or historical `G17`/completion tests. |
| GD-C14 ○●○ PARTIAL §3.1 | `contains`, direct `requires`, and effective `requires` are acyclic; self- or back-dependencies induced via atomic expansion are likewise invalid. | `BASE` (M0 base) | `BASE` and effective DAG checks hold only for the relations they actually evaluate; full atomic expansion must be verified separately. |
| GD-C15 ○●○ PARTIAL §3.1 | Requiring the same target both directly and via ancestor inheritance is locally redundant. During migration, removing this duplication must never roll back the complete atomic representation. | `BASE` (M0 target base, before M1, jointly with atomic normal form) | `CQR-001` flags redundant authored requirements, including duplicates of inherited ones, and non-atomic requirements in canonical Gymnasium roots (hard via protected floors); delivered removals are bound to the replayed prerequisite-reduction proof registry. Non-canonical roots are not checked. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-C16 PARTIAL §3.1 | Transitive minimality is binding. First establish or retain complete atomic prerequisites, next remove redundant non-atomic requirements, and only then transitively reduce the atomic relation. Testing raw authored edges against an inherited alternative path alone never justifies deleting an atomic edge. | BASE (M0 target base, before M1): full G17 check on the atomic target state; any violation or incomplete check blocks every subject milestone pass M1–M7 | Since 11 November 2026, CQR-001 runs the full G17 check (atomic normal form and transitive minimality, including required cross-subject goals and inheriting ancestors) for every root of the complete canonical Gymnasium catalog; a violation caps the subject at M0 and so blocks M1–M7, and an incomplete catalog aborts the status run. Hard in CI via the protected floors only; other canonical subjects report M0, and non-canonical roots are not checked. A pass certifies the resulting state; the migration order is evidenced only for reductions bound to the proof registry. CQR-102/GVR-012 remain directly atomic. |
| GD-C17 PARTIAL §3.1 | Transitively reducing a fixed atomic relation preserves its closure. When modifying a mixed inventory, recompute inheritance and expansion; preserve atomic demand and all relevant scope, applicability, and frontier contracts. Raw reachability between non-atomic cluster nodes is not a general preservation invariant. | Cross-stage migration contract; fulfilling it underpins the atomic G17 target state as BASE (M0 target base before M1) | Registered prerequisite reductions of the canonical Gymnasium catalog are replayed with their witnesses against digest-bound baseline and current documents in every status run, which aborts on failure; this covers the closure share of delivered reductions. It does not prove preservation of the further scope, applicability, and frontier contracts in general, nor anything for non-canonical roots; a raw-versus-effective reachability comparison on its own does not prove a valid migration. |
| GD-C18 PARTIAL §3.1 | Identities and references must be resolvable; unknown goals are never treated as satisfied prerequisites. | BASE (M0 base) + RT | BASE checks static references; fail-closed runtime semantics is verified separately. |
| GD-C19 PARTIAL §3.2 | Single-parent tree structure is an optional profile rather than a global DAG requirement. Forbidding own ancestors is profile-bound; requiring own descendants can already induce effective self-dependency. | BASE (M0 base for actual cycles); optional profile separate | GVR-001/003/006 operate within declared scopes; there is no universal single-parent tree mandate. |
| GD-C20 PARTIAL §3.2 | Skill-graph validity, projection validity, and full learnability from a declared entry state are distinct claims. Atomic self-dependency and an empty initial frontier can refute learnability. | BASE (M0 base) + VIEW; learnability profile separate | A single green validator pass never proves all three claims simultaneously. |
| GD-C21 PARTIAL §3.3 | Every non-exempt atom lies on a direct atomic route Orientation → Atom → Terminal Autonomy. Exemptions require a documented machine-readable selection contract; clusters do not serve as route bridges. | M3 (ROUTE) | ROUTE; GVR-012 applies only within its declared profile. The formula is a structural criterion, not a proof of pedagogical quality. |
| GD-C22 PARTIAL §3.3 | Route profiles define stage, goal kinds, ordering, and validation scope. Memory support is added only where needed, not as a universally mandatory intermediate step. | M3 (ROUTE); memory content decision M6 (REVIEW) | CQR-101/102/103 and GVR-012/013 apply within scoped profiles; CQR-302 is checked separately. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-C23 OPEN §3.3 | Cluster overviews may be derived from atomic reachability. An existential summary edge is never a universal prerequisite; percentage coverage requires a declared denominator; author no new cluster `requires`. | Cross-stage presentation; prohibition of new authored cluster edges follows from `BASE` (M0 target base before M1) | No standalone `CQR` pass covers overview display semantics. |
| GD-C24 OPEN §4.1 | Exactly one stored binary state per atom; partial scoring creates no third mastery tier. Ordinary completions evidence understanding, whereas special goal types use their own completion contracts. | `RT`, cross-stage | No curriculum `M`-gate; see goal-type rules. |
| GD-C25 OPEN §4.1 | $M_{\text{eff}}$ equals explicit mastery united with derivations from durably valid local contracts. It acts globally on prerequisites and actual progress across focus switches without writing synthetic mastery records. Orientation denotes only a completed entry. | `CONCEPT`, no M-gate | New global implicit-mastery recognition is explicitly not yet implemented; M7 does not prove it. |
| GD-C26 OPEN §4.1 | $\text{Sat}(g)$ holds iff $g$ has a non-empty atomic basis contained completely in $M_{\text{eff}}$. Mastering a dependent content goal alone never proves its prerequisites, and cluster completion never writes mastery back to children. | `RT/CONCEPT` | Appendix B documents the historical `core`-biased cluster aggregation divergence; live conformance must be verified separately. |
| GD-C27 OPEN §4.2 | The global frontier consists of unmastered atoms whose full transitive effective atomic prerequisites are satisfied; traversal never stops at mastered intermediate goals, and partial work never counts as satisfaction. | `RT/CONCEPT` | No curriculum `M`-gate covers this overall runtime claim. |
| GD-C28 OPEN §4.2 | A separate cluster frontier uses own and inherited atomic bases together with transitive closure and never replaces the atomic learning frontier. The UI may apply further explicit selection rules. | `RT` | Never equate the mathematical frontier with every UI menu listing. |
| GD-C29 PARTIAL §4.3 | Weighted progress counts unique target atoms rather than clusters and children simultaneously. Weights must be positive; an empty target set never receives a fabricated 100% value. Uniform weights are a domain convention rather than a requirement; the Personal Curriculum fixes the target denominator. Progress share is neither fractional per-goal mastery nor a grade or probability. | `RT`; attribute base `BASE` (M0) | Schema enforces numeric `weight` and the closure compiler (10 November 2026) fixes the personal denominator $L$; no global gate checks strictly positive weights and every display formula. |
| GD-C30 OPEN §4.3 | Segmented progress views may distinguish explicit and implicit contributions, while total progress counts their union. Any counter restricted to explicit mastery must be labeled as such. | `RT/CONCEPT` | Joint explicit/implicit progress accounting is proven neither by M7 nor by current display components. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-C31 PARTIAL §5.1 | A filter is a predicate; applicability is a partial dimensional map. An omitted dimension is unrestricted, an explicit empty value set excludes concrete values, and ALL ignores a dimension solely as a query value and is never persisted. An unresolved personal stage stays unresolved. | VIEW/RT; M2/M5 only for respective source/ warning-debt shares | Filter validator and CQR-501 check compiled applicability in canonical scopes; not a blanket proof of every runtime filter case. |
| GD-C32 PARTIAL §5.1 | Applicability overrides justify exceptions; runtime evaluates compiled applicability. Visibility alone never proves official source coverage. | M2 (SOURCE) for source evidence; M5 (VIEW) for warning debt; RT | SOURCE/VIEW are checked separately; an accepted override waiver never substitutes for CQR-003 source coverage. |
| GD-C33 PARTIAL §5.2 | Preserve the full transitive atomic closure prior to view or focus restriction; global inheritance and atomic expansion always precede filtering. Under the 10 November 2026 Product Owner decision, this closure together with the selected seed targets forms the fixed personal learning scope $L$. Hidden children never turn a cluster into an atom. | VIEW/RT | Verify full closure and identity preservation separately; existing opaque goalEntry cluster atomization remains an open implementation gap. |
| GD-C34 OPEN §5.3 | Normal mode restricts globally computed transitive atomic requirements to focus endpoints; an empty relative prerequisite basis does not block. Strict mode checks all global prerequisites for candidates inside the focus. The strict frontier is a subset of the normal frontier. Out-of-focus prerequisites are never marked mastered. | RT/CONCEPT | No curriculum M-gate. Global closure computation prior to filtering must be verified separately. |
| GD-C35 OPEN §5.3 | Given $g \to x \to p$ and focus $\{g, p\}$, $p$ remains required in normal mode while strict mode additionally requires $x$. Mastering $x$ never removes the transitive requirement on $p$. | RT/CONCEPT | Distinct mediated-transitivity case, not proven by direct-neighbor checks alone. |
| GD-C36 OPEN §5.4 | Global diagnostics reports all missing transitive atoms and distinguishes in-focus from out-of-focus blockers. Listing direct clusters is insufficient; diagnostics creates neither edges nor mastery and is distinct from normal-mode blockers. | RT | No curriculum M-gate. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|

## A.7 X – Context, Views, Goal Types, Provenance, and Quality

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X01 PARTIAL §6 | Goals, program units, placements, competency axes, and composition views serve distinct responsibilities. Reordering creates no new competency and deletes no learner results; broad taxonomy categories are not concrete assessable skills. | Cross-stage model contract; M5 target (`VIEW`) for delivered views | Schema and composition validator separate the layers structurally; complete migration across all legacy inventories is not claimed. |
| GD-X02 PARTIAL §6.1 | Goal identity persists across new placements. Program-unit type `exam` is not an exam goal; `phase` induces neither `contains` nor `requires` edges. Historical year trees are compatibility scaffolding rather than durable domain identity. | `BASE` (M0 base for identity); `VIEW/RT` for projection | `BASE/VIEW` cover only structural sub-obligations. |
| GD-X03 PARTIAL §6.2 | `programUnits` form a tree or forest; `goalPlacements` distinguish `primary`, `secondary`, and `assessed` relations. Secondary relevance or assessment links never create duplicate visible goal nodes. | M5 target / `VIEW` | Schema and composition validator enforce structure and duplicate-freeness of compiled views; `CQR-401` on its own only counts files. |
| GD-X04 PARTIAL §6.2 | Within a resolved context, a goal has at most one matching primary anchor; an absent placement never permits inventing a parent node. Multiple matching primary placements require reviewed resolution. Context never alters goal IDs. | M5 target / `VIEW` | Placement and projection checks run separately; they imply no automatic domain review. |
| GD-X05 OPEN §6.2 | Content view follows `contains`, program view follows program units and placements, and competency view follows catalog/competency references. Child order stems from declared data order; topological learning order is a distinct view. | `VIEW/RT` | No blanket `M`-gate covers all view-rendering semantics. |
| GD-X06 PARTIAL §6.3 | Composition views specify tree structure, child order, and canonical references—never newly authored atoms or `requires`. Renderers must not override an authoritative view with title, phase, or placement heuristics. | M5 target (`VIEW`); `RT` | Composition validator enforces reference-based views free of authored atoms or `requires`; the UI renderer contract remains separate. |
| GD-X07 HARD §6.3 | The compiled default learner tree displays each goal at most once and under at most one parent node. Canonical polyhierarchy requires deterministically validated resolution; role resolution alone never legitimizes duplicates. | M5 target / `VIEW` | Hard-enforced across all composition views via `CPV-004/005/006` in the composition validator; imposes no global single-parent constraint on canonical goals. |
| GD-X08 PARTIAL §6.3 | Compile early: select context/view, resolve references and full prerequisite closure, verify explicit roles and visible parents, and validate the compiled tree prior to UI rendering. | M5 target / `VIEW` | Composition, role, and filter checks verify static compilation of canonical views; `CQR-401` only counts files, and cross-subject `APV-102` alignment remains open. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X09 OPEN §6.3 | No canonical cluster may be turned into an independently completable surrogate atom via `goalEntry` or display compaction; this rule holds regardless of any semantic-kind review. The canonical atomic basis and required prerequisite chains remain intact, with neither new view `requires` nor mastery propagation to children. | M5 target / `VIEW` | `CPV-009` blocks only `curricular Area` nodes in views; existing runtime cluster atomization is not yet remediated, and M7 does not prove it. |
| GD-X10 OPEN §7.1 | Catalog navigation precedes personalization. Level 1 is the base curriculum, Level 2 the backend-owned durable personal scope, Level 3 the temporary focus with one active atom, and Level 4 the binary result bound to stable goal IDs. View or focus changes preserve results; cluster state is derived. | `RT`, no curriculum `M`-gate | Curriculum QA maturity implies no blanket runtime implementation claim. |
| GD-X11 OPEN §7.1 | `LK` is a per-subject course attribute rather than an implicit stage; an absent stage remains unresolved. Changing Level 2 revalidates focus and the active goal without granting or deleting global results. Configuration occurs in designated first-party SkillPilot surfaces, never autonomously inside the OpenAI V1 coach. | `RT` | Provider and authorization boundaries are not an M7 gate. |
| GD-X12 PARTIAL §7.1 | $L$ comprises seed targets together with all necessary atomic prerequisites, independent of current mastery; mastered goals stay satisfied inside the denominator. Focus narrows active work only, never total progress or completion. Open necessary atoms require an acquisition target in the full Personal Curriculum; each canonical ID counts once. | `VIEW/RT`; M2/M5 only for concrete view shares | 10 November 2026 Product Owner decision; implemented in backend and WebGUI. Full native, curriculum, and remote CI acceptance remain separate; `CQR` maturity is no substitute. |
| GD-X13 PARTIAL §7.2 | Authored subject roles and their specificity precedence hold: omitted role defaults to `target`, direct `goal Entry` overrides subtree inheritance, closer root overrides broader root, and ties resolve to `target`. Roles are never inferred from age or phase. The explicit personal closure rule adds necessary atoms; a subject-local `prerequisiteOnly` role is valid when an acquisition target exists elsewhere in $L$. | `VIEW/RT` | `CPV-008`, role unit tests, and the closure compiler check `projection Role` and specificity; no single `CQR` gate proves every cross-view overlap case. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X14 PARTIAL §7.2 | Authored seed references remain hard-validated. A necessary atom's compiled target role backed by a verified canonical path and provenance suffices without manual per-atom authoring. Unknown references, cycles, and missing or unreachable necessary atoms are errors of validation, curriculum QA, and native acceptance; under the 11 November 2026 decision, a learner read of a context whose canonical graph is not strictly compilable keeps the reviewed seed scope without derived closure, claims no learning basis, offers no goal with broken demand, and logs once per curriculum and error code. Seed-selection errors stay hard. Normal mode continues to ignore valid out-of-focus requirements. | Cross-stage projection/RT contract | CPV-002 and validate:graph enforce reference and cycle errors; the personal closure (10 November 2026) and the seed-scope fallback (11 November 2026, b511654efb, with regression tests) are implemented in the backend; applicability and migration acceptance remain separate. |
| GD-X15 PARTIAL §7.3 | Applicability describes evidenced visibility and placement describes structural location; neither implies mastery. Declared mapping-inheritance boundaries shield supplementary branches from false broad provenance while retaining direct sources. Local endpoint visibility may follow from requires under an explicit contract without proving official source coverage. | M2 (SOURCE) for source evidence; M5 (VIEW) for warning debt/ views; RT | SOURCE/VIEW; inheritance boundaries and derived applicability never automatically satisfy CQR-003. |
| GD-X16 OPEN §7.4 | Focus is the union of selected visible roots and their descendants within the Personal Curriculum. Normal mode checks internal transitive effective requirements while strict mode checks global ones; ignoring out-of-focus blockers never grants mastery. | RT | Refers to GD-C33–GD-C35; no additional curriculum M-gate. |
| GD-X17 OPEN §7.4 | Focus expansion follows the unique visible parent path, forms a strict superset containing at least one open goal, replaces only subsumed multi-root branches while preserving independent roots, and requires the coach to adopt the exact backend option and await learner confirmation. | RT | Standalone backend and interaction conformance; not an M-gate. |
| GD-X18 OPEN §7.4 | Only genuine focus completion justifies expansion; an empty frontier may indicate a prerequisite blockade. Newly exposed prerequisites are evaluated normally; mere successor mastery never propagates backward, whereas validly recognized implicit application persists globally and is never reopened. | RT/CONCEPT | Global recognition remains conceptual; distinguishing true completion from blocked empty frontiers is verified separately. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X19<br>PARTIAL<br>§8 | The skill graph receives a binary result resolved according to goal type and never grades answers, dialogues, or flashcards itself. Ordinary understanding must cover all required competency aspects; one right exercise is not automatically full mastery, nor is there an arbitrary minimum task count. | RT; positive content evidence M7 (REVIEW) | CQR-303/P checks authored performance profiles, not live learner success per session. |
| GD-X20<br>OPEN<br>§8.1 | Sufficiently evidenced mastery is persisted immediately; success may be announced only after confirmed persistence. Extra learner consent to save is not a completion condition. Moving to a new topic requires a backend-selected proposal and recognizable learner agreement. | RT | Persistence and coach-interaction contract; not an M-gate. |
| GD-X21<br>PARTIAL<br>§8.2 | Orientation conveys honest, motivating possibilities and tests neither prior knowledge nor subject competency, recall, or transfer; there are no motivation percentages. Orientation nodes are pure prerequisite-free atomic entries without exam or memory metadata. | BASE (M0 base)/ M3 (ROUTE) for structure; RT for dialogue | GVR-014 enforces structure and route profiles check embedding; dialogue quality is not proven thereby. |
| GD-X22<br>OPEN<br>§8.2 | Selecting an interest opens exploration rather than immediate completion: concrete connection, personal response, then—after a meaningful reply or explicit wish to move on—a positive summary with a separate completion offer, completed only upon further confirmation. Never apply this flow to content goals. | RT | No curriculum M-gate covers the conversational turn sequence. |
| GD-X23<br>OPEN<br>§8.2 | Upon a genuine fresh start without subject progress, open orientation goals are the sole selectable frontier goals; existing subject progress never reverts a learner back into an entry lock. | RT | GVR route and orientation structure checks do not prove personalized frontier entry logic. |
| GD-X24<br>PARTIAL<br>§8.3 | A memory goal bundles its required flashcards. SRS practice and controlled verified recall are distinct process states; card flipping or self-rating is never proof of verified recall. Exactly one binary goal result exists, with no extra mastery tiers. | RT; M6 (REVIEW) only for memory content justification | CQR-302 does not inspect individual learner SRS or recall results. |
| GD-X25<br>OPEN<br>§8.3 | Repetitions belong to flashcard learning; completing a memory goal requires its own explicit verification record. Subsequent review due dates alone do not revoke that completion. Ordinary completed goals receive neither review due dates nor any mastery revocation derived from them; voluntary practice stays separate. | RT, binding Product Owner decision; no curriculum M-gate | Individual practice scheduling and durable goal completion are separated. Historical code descriptions are dated audit findings rather than competing norms. The contract introduces neither an extra assessment entity nor a new schema; specification prose is not proof of runtime implementation. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X26 PARTIAL §8.3 | Every required flashcard deck must resolve completely and validly; missing cards must never be treated as an empty auto-completed deck. | M6 (REVIEW) for authored deck integrity; RT for individual completion | Memory-card review checks and CQR-302 enforce authored deck integrity in CI and on M6/M7 floors; individual runtime completion stays separate. |
| GD-X27 OPEN §8.3 | The specified SRS process tracks interval, repetition count, ease factor, and next due date: unscheduled cards are due immediately, successful practice extends intervals, and insufficient recall resets the streak. Scheduling never substitutes for verified recall. | RT | Documented runtime process; M6 proves neither its runtime conformance nor any mastery revocation. |
| GD-X28 OPEN §8.3 | Controlled recall may begin on the active memory goal even while SRS practice items remain queued: prompt first, independent learner response second, expected answer and recall evaluation third. Passing updates the practice trace; failure affects the card process. | RT | Card scheduling and goal completion are separate claims. Subsequent due dates alone never revoke a completed memory goal; no mastery revocation is introduced here. |
| GD-X29 PARTIAL §8.4 | An exam goal carries task, reference solution, and grading rubric; valid step points yield $score = \min(\sum_j p_j, maxPoints)$, passing at $passingPoints$. The formula legitimizes neither fabricated or negative step points nor ad-hoc task alteration. | M4/M5 (EXAM) for authored examData; RT for grading | Schema, GVR-016 (tables globally), and EXAM (in scopes/floors) validate examData; live session grading is not an M-gate. |
| GD-X30 OPEN §8.4 | Normal learning progression checks transitive effective prerequisites. A deliberately chosen direct exam attempt is not blocked by missing preparatory mastery, though a valid task, scope, and authorized session remain required. Direct attempt access itself is not global implicit recognition. | RT | Not covered by M7; any future holistic unit-lock profile remains undecided. |
| GD-X31 OPEN §8.4 | An exam presents the complete unaltered task without substantive hints and grades the full submission against the rubric. A pass is persisted immediately and awaits confirmation; a failed attempt persists neither result nor score and leaves prior mastery intact. | RT | Static metadata checks do not prove live session conformance. |
| GD-X32 OPEN §8.4 | Post-exam feedback reports outcome, score, rationale, and solution, with fresh tasks on retakes. Unlimited retakes remain available; prior exposure to a solution creates no extra mastery tier. Broader implicit-application recognition requires a reviewed local content contract, neither a second personal status flag nor a follow-up exam. | RT/CONCEPT; independent content review separate | EXAM/CQR-303 do not fully establish the new implicit-recognition contract. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X33 ○○○ OPT/DOC §9 | A schedule binds target scope, curriculum revision, calendar, capacity, and plan version. Absolute calendar dates never belong on canonical goals, clusters, or placements; rescheduling never alters goal identity. | OPTIONAL/RT | No curriculum M-gate. |
| GD-X34 ○○○ OPT/DOC §9 | Time planning is supplementary execution scheduling rather than a fifth personalization level. It prioritizes available goals without creating `requires`, frontier entries, or results; planned acquisition must precede planned downstream use. | OPTIONAL/RT | Claims no new runtime access validator. |
| GD-X35 ○○○ OPT/DOC §9 | Use net available minutes after calendar outages; fit learning activities, reserve buffer, and binding subject budgets without double counting. | OPTIONAL | Detailed formulas in GD-P07–GD-P09; no M-gate. |
| GD-X36 ●○○ OPEN §9 | Planned target and actual progress share the exact same named scope. Explicit and validly derived implicit completions form total actual progress; manual classroom-coverage markers neither replace nor mask learner completion. Open gaps remain visible, and revisions never rewrite historical events. | RT/CONCEPT | Joint global actual-progress accounting is not proven by M7. |
| GD-X37 ○○○ OPT/DOC §9 | Keep the local planning pilot distinct from the target backend-owned architecture; the learning-unit definition claims no implemented cross-subject budget validator. | OPTIONAL/ DOCUMENT | Local pilot and target architecture must be documented as distinct implementation states. |
| GD-X38 ○●○ PARTIAL §10.1 | Official sources justify a goal, first-party visualizations explain it, and supplementary materials support acquisition. None of the three is personal competency evidence. | M1/M2 (SOURCE) sources; M7 (REVIEW) image QA; RT boundary | SOURCE, image QA/REVIEW, and the content-package validator enforce separated reference types in their scopes; none proves personal competency. |
| GD-X39 ○●○ PARTIAL §10.1 | Materials reference the curriculum rather than embedding provider lock-in inside the graph. New optional external materials live in separate content packages bound to stable goal IDs; package selection alters neither scope nor `requires`, mastery, or completion, and material gaps never block learning. | ARTIFACT/RT | Content-package and link-migration checks enforce separated packages with stable goal IDs in CI; no M-gate covers the entire runtime contract. |
| GD-X40 ○●○ PARTIAL §10.1 | Sources, mappings, and provenance stay with the curriculum, and first-party visualizations retain their canonical quality contract. Legacy links never establish a permanent second authoring architecture, and returning a URL never proves its content was read. | M1/M2/M7 (SOURCE/REVIEW) respective evidence; ARTIFACT | GVR-008 forbids legacy goal links globally while SOURCE and image QA check evidence in their scopes; coach content fetching is not gated. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X41 PARTIAL §10.2 | Maintain a traceable provenance chain: goal → official source → concrete section → extraction → reviewed mapping decision. Justify 1:1, partial, aggregating, or split mappings explicitly rather than claiming verbatim identity everywhere. | M1/M2 (SOURCE) | SOURCE; technical completeness alone does not substitute for domain source review. |
| GD-X42 OPEN §10.2 | For requires and contains, distinguish explicit official-source statements from pedagogical modeling decisions. Never fabricate provenance; keep unresolved, stale, or conflicting evidence visible. | M2 target (SOURCE) for provenance; M3 (ROUTE) for sequencing review | No current CQR gate automatically verifies every individual edge rationale. |
| GD-X43 PARTIAL §10.2 | Human-readable source rationales are derived views of structured evidence, never a second authoritative register. Keep references and fingerprints consistent across documentation, exports, and runtime; MEM/FWU alignment replaces the official source route only under an explicitly reviewed scope contract. | M1/M2 (SOURCE) + ARTIFACT/RT | Source-rationale and package-conformance checks verify consistency in covered scopes; not a blanket substitute across all interchange formats. |
| GD-X44 PARTIAL §10.3 | Parser, reference, cycle, projection, route, and content checks carry distinct evidentiary claims and each requires a declared scope. No single migration gate or structural pass certifies them all. | Cross-stage (BASE/ROUTE/ VIEW/REVIEW) | BASE/ROUTE/VIEW/REVIEW are explicitly separated. |
| GD-X45 PARTIAL §10.3 | Five current content-evidence dimensions: D domain-clear description, P observable understanding performance, A appropriate atomic sizing, M justified memory-support decision, and V domain-accurate hash-bound visualization. | M5 A, M6 M, M7 simultaneous D/P/ A/M/V completion (REVIEW) | REVIEW enforces A (M5), M (M6), and the simultaneous D/P/A/M/V intersection (M7) without averaging on protected M6/M7 floors; unprotected inventories remain below. |
| GD-X46 PARTIAL §10.3 | M7 is complete automated curriculum-QA closure atop all lower maturity levels for the declared scope. It implies neither human sign-off, learner trials, nor learning effectiveness; multi-model AI agreement never substitutes for mandatory human decisions. | M7 boundary (REVIEW); human evidence outside M0–M7 | CQR-303 enforces automated M7 closure across the five M7 floors; human sign-off, learner trials, and learning effectiveness lie outside M0–M7. |
| GD-X47 PARTIAL §10.3 | Reviews bind the exact artifact revision; an old image hash never certifies a new image. Content or prerequisite edits can invalidate evidence, technical moves require narrowly justified compatibility, and blanket hash refreshing is forbidden. | M5/M6/M7 per review dimension (REVIEW) | REVIEW checks current digest bindings; never infer substantive review validity from a mechanical hash update. |
| GD-X48 PARTIAL §10.4 | Author domain changes in canonical sources and regenerate downstream artifacts; keep published historical archives byte-immutable. | ARTIFACT | Plugin-freeze, release, goal-book, and status-diff checks guard canonical regeneration and archive immutability in their CI paths; they grant no publication authorization. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-X49 PARTIAL §10.4 | Build JSON and ontology packages from the identical canonical source state; the JSON runtime remains normative. Roundtrips preserve identities, relations, roles, prose, and domain fields per equality contract (semantic equality for texts, byte identity for binary assets) without divergent runtime semantics. | ARTIFACT | Package-contract and conformance checks verify JSON/ontology parity in CI (fast mode in routine CI); full standalone roundtrip verification remains a dedicated release check. |
| GD-X50 PARTIAL §10.4 | Close external goal references completely—including cross-subject prerequisites—without empty placeholders. Complete package closure does not by itself define Personal Curriculum targets. | BASE (M0 base for references) + ARTIFACT/VIEW | validate:graph, standalone-builder tests, and the closure compiler check reference closure in their scopes; package closure and Level 2 targets stay separate. |
| GD-X51 PARTIAL §10.4 | Every verification or release claim names its exact artifact, scope, version, and source digest binding. A local build proves neither host display nor learning effectiveness, and this compendium grants neither release nor production sign-off. | ARTIFACT/ DOCUMENT | Release, submission, and goal-book checkers bind artifact, version, and source hashes in CI; prose claims, live host display, and production approval lie outside. |
| GD-X52 PARTIAL §10.5 | Function determines license: software, technical schemas, developer docs, and functional coach instructions carry Apache-2.0; first-party skill landscapes, content, and didactic media carry CC-BY-4.0. There is no blanket dual licensing. Third-party rights stay intact; provenance or AI generation proves neither ownership nor quality; exports carry license, attribution, and modification notices. | ARTIFACT | First-party license and package checks enforce Apache-2.0/CC-BY-4.0 declarations on covered repository artifacts in CI; third-party rights clearance is not an M-gate. |
| GD-X53 PARTIAL §10.5 | Private learner results, sessions, and conversations fall neither under the public content license nor into public source or review packages. | ARTIFACT/RT | Submission-ZIP audits and package-contract checks reject extraneous or private files in CI; end-to-end runtime privacy is not an M7 claim. |

## A.8 U – Recursive Learning Units

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-U01 OPEN §11 | A LearningUnit recursively binds a learning promise, own atomic goals, and optional child units. Calendar periods are not synonyms; personalization levels, QA maturity, and difficulty retain independent meanings. | CONCEPT, no M-gate | Not claimed as an implemented runtime feature. |
| GD-U02 OPEN §11.1 | Unit motivation explains purpose ahead of individual tasks. Only a genuine orientation goal uses the orientation completion contract; a synthetic heading is not a goal, and derived unit completion never claims an orientation dialogue actually took place. | CONCEPT/RT | Unit display derives no synthetic orientation mastery. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-U03 ●○○ OPEN §11.1 | A pure aggregate unit requires no extra exam. Added integrative synthesis claims may require a synthesis exam, and genuinely new competencies require own acquisition assignment plus valid exam coverage; pure synthesis forces neither artificial ordinary goals nor re-passing child units. | CONCEPT; content review independent | EXAM/M7 do not prove a complete learning-unit contract. |
| GD-U04 ●○○ OPEN §11.2 | Own($U$) denotes own atomic goals and Children($U$) mandatory child units after choice resolution. $T(U) = \text{Own}(U) \cup \bigcup_{V \in \text{Children}(U)} T(V)$ is non-empty. Structural markers are not goals; an aggregate unit may have empty Own($U$) when its child scope is non-empty. | CONCEPT | Future structural contract; no existing CQR gate today. |
| GD-U05 ●○○ OPEN §11.2 | R4: every acquisition target has exactly one directly owning unit, the smallest authoritative owning unit. Viewing a parent never turns child goals into parent-owned goals; overlapping references unite as sets, and focus switches alter neither ownership nor local recognition contracts. | BASE (validity condition of every authored unit definition, same class as acyclicity): hard from the first authored unit; a violating unit cannot be merged | No unit definitions and no unit validator exist on main; the placement requires a fail-closed check from the first authored unit. Multi-view display creates no duplicate acquisition or progress counter. |
| GD-U06 ●○○ OPEN §11.2 | R1: every own ordinary content goal lies on a prerequisite path from at least one own assessment of the unit, directly or transitively; a path that leads only to a memory goal does not count. The reviewed application coverage of such an assessment is R2 (GD-A08). Without both the local contract is invalid; arbitrary manual waiver masks never substitute for it. | BASE (validity condition of every authored unit definition, same class as acyclicity): hard from the first authored unit; a violating unit cannot be merged; content share R2 see GD-A08 | No unit definitions and no unit validator exist on main; ROUTE/ EXAM check related properties rather than this unit condition; no fabricated M7 gate. |
| GD-U07 ●○○ OPEN §11.2 | Authoritative program/placement or reviewed composition structure defines a learning unit. Expanding an arbitrary contains cluster never turns it into a learning unit; general cluster satisfaction stays distinct. | CONCEPT/VIEW | Clusters are never automatically classified as learning units at runtime. |
| GD-U08 ●○○ OPEN §11.3 | Complete($U$) holds iff all own goals lie in $M_{\text{eff}}$ and all mandatory child units are complete. Neither extra unit-mastery state is stored nor is mastery written backward onto children. | CONCEPT | Document-model verification is neither a runtime nor an M-gate. |
| GD-U09 ●○○ OPEN §11.3 | Exam and memory goals remain explicit; passing a parent exam replaces neither a child exam nor verified recall. Synthesis assessment is an additional own obligation, card decks alone never prove application, and child recall obligations must never be hoisted onto parents to fake child completion. | CONCEPT/RT | Never infer runtime fulfillment from M6 card review or M7 content status. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-U10 ●○○ OPEN §11.4 | Own ordinary competencies can be implicitly recognized ahead of independent memory, child, or parent completion. Recognize only own goals of the local contract—never via a second parent contract spanning all descendants. Open parent synthesis never revokes valid child recognition. | CONCEPT | Global runtime effect is not yet implemented; see GD-A10–GD-A14. |
| GD-U11 ●○○ OPEN §11.5 | An explicitly authorized whole-unit prerequisite denotes full unit completion and lives outside $C$ and $R$. An atomic prerequisite requires only that competency rather than the entire foreign unit; neither compiler nor coach may conflate the two. | CONCEPT; access policy OPTIONAL | An atomic edge induces no whole-unit hard lock. |
| GD-U12 ●○○ OPEN §11.5 | The unit hierarchy is acyclic. Any claimed undivided unit-completion sequence must additionally be checked for block cycles; an atomic DAG alone does not suffice. Cyclic block grouping requires regrouping or finer interleaved sequencing rather than inducing an implicit unit-wide lock. | CONCEPT / conditional structural contract | Finite document model verifies examples; no production unit validator is claimed. |
| GD-U13 ○●○ PARTIAL §11.5 | Transitive minimality remains binding for authored **requires** in accordance with the atomic target norm and ordered migration. No extra minimality rule is posited for a not-yet-decided holistic unit-access relation. | BASE (M0 target base before M1 per GD-C11/GD-C16); unit relation OPTIONAL | The authored-**requires** share is enforced as in GD-C11/GD-C16 (CQR-001, hard via protected floors, canonical Gymnasium catalog only); no unit relation exists to check. This shared minimality reference never authorizes deleting atomic edges in favor of remaining inherited cluster requirements. |
| GD-U14 ○○○ OPT/DOC §11.5 | Both a hard unit-access profile and direct exam access under it remain undecided. The illustrative equations $\mathrm{Open}(U) = \bigwedge_{V \in \mathrm{Pred}(U)} \mathrm{Cleared}(V)$ and $\mathrm{Cleared} = \mathrm{Open} \wedge \mathrm{Complete}$ apply only if an access profile is later authorized. Complete denotes evidentiary state and Cleared course policy; neither grants mastery, and the formula does not settle direct-exam exceptions. | OPTIONAL, no curriculum M-gate | Claim neither authorization nor implementation of a technical unit lock. |
| GD-U15 ●○○ OPEN §11; §11.1 | A unit establishes secure independent application of its promise before planned learning builds on it; its assessments expose essential misconceptions, and a gap leads to focused learning and a fresh equivalent assessment. Boundaries follow meaningful conceptual transitions, not existing assessments, target quotas, headings, or edge density; edges are never changed to fit a cut. The ordinary grading contract applies: no hidden pass rule, no second stored outcome. | CONCEPT; content review (REVIEW) separate; no access lock | Didactic design and review criterion; no validator or M-gate checks unit boundaries or purpose, and no unit is authored on **main**. |

## A.9 P – Optional Acquisition and Time Planning

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-P01 ○○○ OPT/DOC §12 | An optional schedule assigns acquisition targets to calendar levels; it neither defines learning units, alters core skill-graph validity or mastery, nor introduces holistic unit locks or blanket full-year completion gates. | OPTIONAL | Claims no new production budget validator. |
| GD-P02 ○○○ OPT/DOC §12.1 | A plan specifies atomic acquisition targets $T$, a disjoint declared entry basis $B$, a positive number of levels, and a unique first-acquisition level $\ell$ per target defining level sets and prefixes. Later practice or application is scheduled activity rather than a duplicate goal ID. $B$ writes no learner mastery; personal entry gaps require diagnosis and time. | OPTIONAL | No curriculum M-gate. |
| GD-P03 ○○○ OPT/DOC §12.2 | The hypothetical plan relation prunes outgoing requirements solely at the declared entry basis $B$, preserving full transitive closure across all planned levels. This planning convention never permits truncating real runtime prerequisite traversal at mastered intermediates. | OPTIONAL | Purely hypothetical planning construct, not a runtime mastery contract. |
| GD-P04 ○○○ OPT/DOC §12.2 | A completion witness orders every level's targets so each enters the global frontier given $B$, prior levels, and preceding witness atoms. Inherited, cluster, and cross-subject requirements remain intact, with neither normal-mode hiding, direct exam jumps, nor anticipated implicit recognition. | OPTIONAL | No curriculum M-gate; not proven by current UI navigation paths. |
| GD-P05 ○○○ OPT/DOC §12.2 | Required atoms equal the full closure under the plan relation. Necessarily, every target's prerequisite demand lies in $B$ or at latest its own level: outside $B \cup T$ is unplanned, a higher level is an ordering error, and same-level dependencies are permitted. Graph and expansion validity precede closure checking. | OPTIONAL | Structural feasibility check only; does not by itself prove time feasibility. |
| GD-P06 ○○○ OPT/DOC §12.2 | The finite fixed-point iteration starts from $B$ plus prior levels and repeatedly adds currently frontier-eligible level targets, succeeding iff the entire level is reached. Eligibility is monotone, each parallel wave is serializable, the test mutates no learner data, and a purely atomic DAG reduces to standard topological sorting. | OPTIONAL/ DOCUMENT | Algorithmic specification, not claimed as a production validator. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-P07 ○○○ OPT/DOC §12.3 | Any time-feasibility claim states reference learners, entry state, target/ elective selection, calendar, estimation method, and uncertainty in a shared calibrated time unit: Work $\geq 0$, Reserve $\geq 0$, Budget $> 0$, and Work + Reserve $\leq$ Budget. Keep reserve protected (or disclose zero explicitly) without double-counting exam or review effort. | OPTIONAL | Without calibrated estimates there is no time-validity claim; no M-gate. |
| GD-P08 ○○○ OPT/DOC §12.3 | The activity model counts actual effort once across acquisition, practice, recall, assessment, and supplementary work. A shared canonical atom is not acquired twice across subjects, though subject-specific application may still consume time. `weight` and uncalibrated goal counts are never minutes. | OPTIONAL | Underlying estimation model and empirical data must be evidenced separately. |
| GD-P09 ○○○ OPT/DOC §12.3 | Worked numerical examples are not empirical curriculum durations. Binding subject, room, or date sub-budgets must hold simultaneously without double-counting shared time, and concrete prerequisites must be scheduled ahead of their planned application. | OPTIONAL | Satisfying a total budget alone is insufficient when sub-constraints apply. |
| GD-P10 ○○○ OPT/DOC §12.4 | Full validity of an optional plan requires both structural and time feasibility under bound assumptions. Without calibrated time data only structural feasibility may be claimed; every claim binds plan, revision, scope, and estimation assumptions without guaranteeing uniform individual learning speed. | OPTIONAL | Neither an M7 gate nor proof of learning effectiveness. |
| GD-P11 ○○○ OPT/DOC §12.4 | Unit completion follows own and child obligations; learning units and calendar periods may cross-cut one another. A schedule profile never creates a blanket full-year prerequisite for later goals. | OPTIONAL/ CONCEPT | Access policy remains separate. |

## A.10 A – Implicit Mastery via Local Assessment Contracts

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-A01 ●○○ OPEN §13 | Explicit ordinary completion evidences understanding and a valid local exam evidences independent application; both act globally. Exactly one binary state is stored—without backward writes, extra qualification tiers, manual waiver profiles, or second sub-exams. Exam and memory goals themselves always remain explicit. | CONCEPT, no M-gate | New semantics is explicitly not yet implemented; never reinterpret legacy tests as proving it. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-A02 ●○○ OPEN §13.1 | Each local contract binds authoritative own unit goals, canonical IDs, curriculum revision, and the reviewed evidentiary scope of own exams. Total child scope never forms an inflated parent recognition contract, and display, focus, or schedule roots never alter unit ownership. | CONCEPT | Future contract and revision check; neither CQR-401 nor M7 substitutes for it. |
| GD-A03 ●○○ OPEN §13.1 | Determine exam and memory goals authoritatively; $P_U = Q_U \cup A_{\text{mem},U}$ protects all own goals of both types from implicit completion. Orientation is not a protected demand root, though a derived unit marker claims neither competency nor dialogue. External support neither inflates the unit denominator nor permits local recognition of foreign Own goals. | CONCEPT | Memory content review alone does not establish this recognition logic. |
| GD-A04 ●○○ OPEN §13.1 | Operate on the fully inherited and atomically expanded relation—including hidden and cross-subject prerequisite support—with all references resolvable and the normalized graph acyclic. Compute full closure prior to display filtering. | BASE (M0 target base for inputs); CONCEPT for recognition use | BASE alone does not prove the recognition normalizer. |
| GD-A05 ●○○ OPEN §13.2 | $N_U = \text{Reach}^*_{R_{\text{at}}}(P_U \setminus M)$ comprises all open protected roots and every transitive prerequisite without stopping at mastered intermediates. $I_U = \text{Own}(U) \setminus (M \cup P_U \cup N_U)$ is disjoint from $M$ and $P_U$. Redundant edges with identical closure leave $I_U$ unchanged. | CONCEPT; formula check DOCUMENT | Legacy contextual waiver logic does not implement or prove this equation. |
| GD-A06 ●○○ OPEN §13.2 | The demand traversal visits reachable atoms $V_U^{\text{reach}}$ and their outgoing $R_{\text{at}}$ edges $E_U^{\text{reach}}$ at most once; computing $I_U$ adds one pass over $\text{Own}(U)$, yielding $O(\lvert\text{Own}(U)\rvert + \lvert V_U^{\text{reach}}\rvert + \lvert E_U^{\text{reach}}\rvert)$ without materializing a quadratic closure. | CONCEPT / algorithmic contract | Verified in the finite document model, not a production benchmark. |
| GD-A07 ●○○ OPEN §13.3 | The implicit-set equation is evaluated only for units whose contract satisfies R1–R4 (GD-U06, GD-A08, GD-A18, GD-U05) with reviewed coverage. Otherwise the unit's own ordinary goals stay open and the violation keeps the curriculum from reaching the respective milestone; a goal that no assessment covers stays open and is a rule violation, never recognized for free. Absence of a downstream consumer proves nothing, and arbitrary waiver masks are forbidden. | CONCEPT for the recognition gate; violations count against BASE (R1, R3, R4) and M5/ M7 (R2) | Recognition is not implemented and no unit validator exists on main; ROUTE is structurally related but not equivalent to a local unit contract. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-A08 OPEN §13.3 | R2: for every own ordinary goal, at least one own assessment that requires it has reviewed evidence that its tasks and normal grading really test the goal, so that no essential skill can be fully compensated away by unrelated rubric points. Neither `requires` paths, titles, `coveredGoalIds`, passing thresholds, nor DAG validity prove this; independent review must verify all tasks, solutions, rubrics, and every passing score combination rather than just their union. | M5 (`EXAM`/`REVIEW`: assessment approval and coverage metadata), current per-goal binding M7 (`REVIEW`); without it the curriculum does not reach the milestone | `CQR-203` checks approval status and coverage metadata of terminal assessments in configured scopes, and `CQR-303` current per-goal `D/P/A/M/V` evidence; neither reviews unit-local coverage, and none of the six required M7 checks is a coverage check. Not enforced today. |
| GD-A09 OPEN §13.3 | Passing an exam remains a single binary learner outcome. Its broader evidentiary reach resides in the pre-reviewed exam contract rather than a second personal status flag; legacy unreviewed exams are never retroactively upgraded by diagrams or software. | `CONCEPT/RT` | Stronger evidentiary meaning resides in reviewed exam content; no second stored learner status follows. |
| GD-A10 OPEN §13.4 | The valid contract set $\mathcal{U}_{\text{valid}}$ is bound durably per learner and revision—neither restricted to the active view/plan nor expanded across all latest revisions. $M_{\text{eff}} = M \cup \bigcup_{U \in \mathcal{U}_{\text{valid}}} I_U$ stays available across focus or scope changes and derives strictly from explicit results $M$ without feeding prior implicit sets back into demand. | `RT/CONCEPT`: versioned contract binding (R5, `GD-A12`) is a precondition before implicit recognition is enabled in production | Not yet claimed in production; no curriculum `M`-gate. |
| GD-A11 OPEN §13.4 | Subsequent exams or units never reopen an already recognized goal. Under fixed contracts, additional explicit mastery can only grow $M_{\text{eff}}$, and moving a goal from $I_U$ into $M$ preserves satisfaction monotonically. | `CONCEPT`; monotonicity proof `DOCUMENT` | Finite model checking $\neq$ production runtime conformance. |
| GD-A12 OPEN §13.4 | R5: a later re-cut of units or later edges never grant additional recognition to existing results; the contract is versioned. A stable exam ID carries stable recognition semantics, and task variants satisfy the same contract. Changed claims require a new assessment identity or reviewed migration; unverified historical exams retain explicit pass status without fabricating broader implicit application. Separate genuine error correction from mere plan or consumer changes. | `RT` (runtime contract): precondition before implicit recognition is enabled in production; `ARTIFACT` for contract versions | Not implemented; no recognition runtime exists yet to enable. No extra personal qualification record; no silent historical upgrade. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-A13 ●○○ OPEN §13.5 | Full unit completion is recursive—equivalent under complete goal representation to $T(U) \subseteq M_{\text{eff}}$, with own $P_U \subseteq M$. Passing own exam and memory goals empties $N_U$ and can recognize own ordinary goals, never child units. Open parent synthesis never revokes child recognition, and passing a parent exam never replaces child exam or memory goals. | CONCEPT | Specification equation, not an existing M7 feature. |
| GD-A14 ●○○ OPEN §13.5 | An open own memory goal blocks full unit completion, yet withholds implicit recognition from another own goal only if it transitively requires that goal. A shared foundation stays demanded as long as any local open protected consumer requires it; a later open consumer in another unit contract never revokes existing recognition. | CONCEPT | Never conflate local contract demand with downstream units. |
| GD-A15 ●○○ OPEN §13.5 | Direct exam attempt access remains the baseline given a valid task, auth, session, and explicit selection. Optional unit-wide locking stays undecided; the learning-unit concept creates neither a lock nor an auth bypass. Normal learning progression evaluates $M_{\text{eff}}$ across full effective transitivity under separate normal/ strict focus rules. | RT/CONCEPT/ OPTIONAL | Neither an M-gate nor an access-lock policy decision is introduced here. |
| GD-A16 ●○○ OPEN §13.6 | Cumulative actual progress counts $\lvert T_{\text{Plan}} \cap M_{\text{eff}} \rvert$, uniting explicit, implicit, and multi-contract contributions prior to counting. Any explicit-only view must state its distinct numerator/ denominator. Current-period reporting counts genuinely dated explicit completion events rather than newly derived implicit items, prior work, or later scope expansion. | RT/CONCEPT | M7 checks no personal plan or period accounting. |
| GD-A17 ●○○ OPEN §13.6 | Static contract structures may be cached per revision, whereas derived implicit completions never form a second persisted source of truth. Invalid or unverified contracts stay visible as explicit limitations. Automated formula checks replace neither independent content review nor human trials and learning-effectiveness evidence. | CONCEPT/ DOCUMENT; evidentiary boundary cross-stage | No curriculum M-gate covers the full persistence and disclosure contract. |
| GD-A18 ●○○ OPEN §13.3 | R3: an assessment claims coverage only for goals it requires directly or transitively: $\text{Cov}(q) \subseteq \{b \in A \mid (q, b) \in R_{\text{at}}^{+}\}$. This guarantees that the assessment which actually tests a goal is among the open protected goals the recognition rule waits for. A missing path is a subject-review finding, never a reason for a redundant direct edge. | BASE (validity condition of every authored unit definition, same class as acyclicity): hard from the first authored unit; a violating unit cannot be merged | No unit definitions and no unit validator exist on main. Related: CQR-104 binds every coveredGoal Ids entry to a visible prerequisite path from the assessment, but only for terminal assessments of canonical Economics; CQR-203 only checks that coverage metadata exists. |

## A.11 D – Document Appendices: Status, Verification Strategy, Sources, and Sign-Off

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-D01 ○○○ OPT/DOC Appendix B | Bind every implementation-status statement to a named audited commit. Historical commit `616324e857` and build manifests make no claim about live `HEAD`. A document build proves neither runtime conformance nor deployment, and global implicit recognition remains marked unimplemented. | `DOCUMENT` | Dated implementation audit findings are not a live conformance check. |
| GD-D02 ○○○ OPT/DOC Appendix B | List historical implementation divergences individually: completion traversal truncation, new protected set and local coverage requirements, contextual waivers vs. durable global contracts, incomplete validity proofs, and `core`-preferring cluster aggregation vs. all atoms. Never present mere test existence as remediation. | `DOCUMENT`/ `CONCEPT` | Dated audit findings, not automatically a live fail/pass list. |
| GD-D03 ○○○ OPT/DOC Appendix B | The memory goal type carries its own explicit completion record; review due dates never revoke completed mastery, and ordinary completed goals receive no review due dates. Learning units remain an unimplemented concept; unit-wide locks and direct exam access under them stay open, and time planning stays separate. | `DOCUMENT`/ `CONCEPT`/ `OPTIONAL` | Keep the binding memory-goal decision distinct from dated implementation findings; explicit card-goal completion, SRS scheduling, and undecided unit-access policy are separate contracts. |
| GD-D04 ○○○ OPT/DOC Appendix C | The mathematical model checker is a reference verification script that reads no learner data and mutates no runtime state. Exhaustive finite DAG, mastery, and protected-set checks hold under assumed complete local assessment coverage; the cited 5,222 cases are a dated document finding. | `DOCUMENT` | Neither a curriculum `M`-gate nor end-to-end production verification. |
| GD-D05 ○○○ OPT/DOC Appendix C | Reduction/closure checks and inheritance counterexamples prove their exact algebraic claims; testing raw edges against inheritance does not prove the new atomic migration priority. Reports must keep the binding target norm distinct from the bounded model test. | `DOCUMENT`; `G17` target base (`BASE`) before M1 stays separate, while M3 (`ROUTE`) covers additional routes | The bounded model test proves its stated algebraic claims, not an implemented gate for the new atomic migration order. |
| GD-D06 ○○○ OPT/DOC Appendix C | Monotonicity, protection of exam and memory goals, recursive unit cases, block-cycle detection, downstream contracts/external support, and error cases are distinct mathematical checks. The 128/948 finite test cases do not constitute full curriculum or runtime conformance, and block-cycle checking creates no lock policy. | `DOCUMENT` | Historical model counts stand strictly as dated figures unless a fresh run is executed. |

| ID / Status | Rule (concise summary) | Area | Enforcement and boundary |
|---|---|---|---|
| GD-D07 ○○○ OPT/DOC Appendix C | Diagram verification checks displayed sets and, under `--check`, deterministic PNG bytes. It proves neither task quality nor grading reliability, identity verification, or learning effectiveness; independent content review and runtime, security, concurrency, and UI tests remain separate. | DOCUMENT | Proves neither M7, live host rendering, nor human-trial outcomes. |
| GD-D08 ○○○ OPT/DOC Appendix D | Keep symbol notation consistent throughout: $K$ denotes structural clusters, $A_{\text{mem}}$ and $A_{\text{mem},U}$ memory goals, Cards($g$) their flashcards, $R_{\text{at}}$ the atomic relation, $M/M_{\text{eff}}$ explicit/ effective mastery, and Own $/T$ direct/ total unit scope. $m = 0$ does not preclude implicit recognition, and glossary definitions introduce no extra outcome tiers. | DOCUMENT | Notation consistency check across GD-C/U/A, introducing no extra substantive rule. |
| GD-D09 ○○○ OPT/DOC Appendix E | Chapter source attributions record normative ownership; cross-referencing a concept never claims production implementation of optional profiles. Never silently import norms from unread external files into this register. | DOCUMENT | Source attribution register, not an M-gate. |
| GD-D10 ○○○ OPT/DOC Appendix E | Attribute static original illustrations and generated deterministic PNGs accurately; equations and legends remain normative while figures create no extra edges. Technical prose and build tooling carry `Apache-2.0`, first-party didactic figures `CC-BY-4.0` where rights hold, with clear provenance and revision metadata. | DOCUMENT/ ARTIFACT | Figure byte and license boundaries are independent of curriculum M7. |
| GD-D11 ○○○ OPT/DOC Appendix F | The build pipeline binds chapters and PNG assets and documents prerequisites and exact commands. Multi-pass LuaLaTeX resolves cross-references, and the manifest records SHA-256 digests of inputs and PDFs alongside dates and tool versions; differing TeX distributions do not guarantee byte-identical PDFs. | DOCUMENT | Document-build conformance, neither a runtime nor a content pass. |
| GD-D12 ○○○ OPT/DOC Appendix F | Document sign-off requires an error-free LaTeX build, resolved cross-references, no unchecked overfull horizontal or vertical boxes, and visual PDF inspection. Mathematical model checking complements sign-off without replacing runtime, domain, or empirical validation. | DOCUMENT | Document sign-off criteria outside curriculum M0–M7. |

**Completeness and Evidentiary Boundary.** This register comprises 36 core (C), 53 context (X), 15 learning-unit (U), 11 planning (P), 18 implicit-recognition (A), and 12 document (D) rule groups, totaling **145 rule groups**. Distinct sub-obligations are preserved within each row while repeated explanations and examples map onto those same obligations. Target assignment claims neither complete enforcement nor a live CI, runtime, release, or learning-effectiveness pass.

**Sources.** Status generator (rule catalog, `deriveScopeMaturity`, `deriveCurriculumMaturity`); skill-graph validator; transitive-minimality auditor; graph-integrity adapter; prerequisite-reduction proof registry; floor checker; floor policy; composition validator; filter validator; schema validator; UUID validator; content-package validator; package-contract check; first-party license check; command registry; CI workflow; maturity and route documentation; graph validation rules; Markdown matrix.

## B Implementation Status and Boundaries of Claims

**Status of this edition.** This appendix records the repository states against which the specification's claims were audited and documents where the inspected implementation diverges from the normative model (`GD-D01`). This unified Single-Source edition consolidates the 4 November concept correction, the 8 November atomic normal-form mandate and milestone mapping, the 9 November projection contract clarification, the 10 November closed Personal Curriculum learning scope, and the 11 November 2026 decisions on non-compilable learning contexts and on learning-unit validity across Web and PDF. Since 11 November 2026, `CQR-001` also enforces `G17` for the canonical Gymnasium catalog. Both backend and WebGUI now evaluate the fixed atomic prerequisite closure; the closure compiler PersonalCurriculumLearningScope.java reuses canonical normalization. Local regression tests cover learnability, planning, focus, reference errors, and cross-subject prerequisite delivery. Full native, curriculum, and remote-CI acceptance as well as alignment of applicability and migration validators remain separate steps. Global application recognition ($M_{\text{eff}}$) is **not yet implemented**. Readable documentation evidence or a green document build proves neither runtime conformance nor production deployment.

### Binding Product Owner decisions.

- **4 November 2026:** Implicit mastery through passed assessments of a valid local contract demonstrates independent application and carries global effect ($M_{\text{eff}}$); transitive minimality remains mandatory. Assessment and memory goals remain protected: for memory goals, verified completion recall alone establishes goal completion, and subsequent card due state never revokes it (Sections 3, 8, and 13; `GD-D03`).
- **7 November 2026:** Every reachable prerequisite of a visible goal requires an explicit projection role (`target` or `prerequisiteOnly`) in subject composition views; at the Personal Curriculum compile boundary, the 10 November 2026 closure rule supersedes blanket manual per-atom view entries with derived target roles (Section 7.2).
- **8 November 2026:** Atomic normal form $R \subseteq A \times A$ as the binding canonical target state without cluster exceptions, mandatory ordered migration sequence, and mapping of all 143 rule groups to validation areas and milestones (Sections 2 and 3, Appendix A).
- **9 November 2026:** Projections preserve full prerequisite closure; a canonical cluster never becomes an atomic runtime goal through a composition view (Sections 5 and 6.3).
- **10 November 2026:** The Level 2 Personal Curriculum learning scope is $L = \text{Closure}(S)$ on the unpruned canonical skill graph, fixed independently of mastery, assigning derived target roles to all necessary prerequisite atoms (Sections 5 and 7).
- **11 November 2026 (learning context):** Hard closure errors belong to validation, curriculum QA, and native acceptance, not to learner reads. When the canonical graph of a learning context is not strictly compilable, the runtime keeps the reviewed seed scope without derived closure, claims no learning basis, offers no goal with broken demand, and logs the failure once per curriculum and error code; seed-selection errors stay hard (Section 7.2).
- **11 November 2026 (learning units):** The recognition rule of 4 November stays: an own ordinary goal is recognized once no open own assessment or memory goal of its unit still requires it (if assessments $\mathsf{Q}_1$ and $\mathsf{Q}_2$ of the same unit both require $\mathsf{A}$, $\mathsf{A}$ stays open until both are passed). The alternative recorded on the archived branch of pull request #83 as a "6

November 2026 target decision" – immediate recognition from the qualified coverage list of a single passed assessment, including proof by a parent or foreign unit – was reviewed and is **not adopted**. Recognition is sound only inside units that satisfy rules R1–R5: R1, R3, and R4 as hard validity conditions from the first authored unit, R2 at M5 with the current per-goal binding at M7, and R5 as a runtime precondition before recognition is enabled in production; a unit's purpose is a secure foundation before planned learning builds on it (Sections 11 and 13.3).

**Audited repository states.** The technical findings in this appendix and in the rule registry are anchored to five explicitly named repository states; none of them substitutes for a fresh full-system acceptance run on a later `HEAD`:

- `616324e857` – Historical implementation audit underlying the runtime divergence rows in the comparison table below; not a fresh full runtime acceptance run.
- `3b71ee9370` – Baseline gate-reach audit in Appendix A against `main`.
- `68db5bf499` – Dedicated projection audit of 9 November 2026 regarding direct `goalEntry` references to canonical clusters.
- `cacf323919` – `main` integration of the Personal Curriculum prerequisite-closure compiler $L = \mathrm{Closure}(S)$ (10 November 2026 decision) and baseline for the traffic-light (*Ampel*) reconciliation and this Single-Source edition.
- `44c5b06b77` – `main` state of this edition (11 November 2026) with the standard `G17` milestone gate (`15f74ef564`, pull request #119), the repaired protected subject data (pull request #121), and the seed-scope fallback (`b511654efb`); basis of the 11 November rule reconciliation. The runtime findings in the table below were not re-audited at this state.

**Dedicated projection finding of 9 November 2026.** At audited state `68db5bf499`, `Completion CanonicalGraphResolver` loads prerequisite chains recursively. However, view projection in `LearnerService` and the resolver still treat direct `goalEntry` references to canonical clusters as childless runtime atomic goals. Historical evidence for the package-backed runtime catalog, offerings, and composition views (work package DPK-006c-c) documents that legacy implementation rather than proving preservation of the canonical atomic basis now explicitly required. The code fix and native verification remain open. This targeted read audit does not refresh the remaining historical runtime findings in the table below.

| Area | Binding specification here | Audited repository status |
|---|---|---|
| Completion traversal (transitivity) | Full transitive open demand $N_U = \mathrm{Reach}^{*}_{R_{\mathrm{at}}}(P_U \setminus M)$ without stopping at mastered intermediate goals. | `CompletionEvaluator.evaluate AutomaticGraph` stops at mastered intermediate goals: in Q → B → A with only B mastered, A is released there, whereas the new formula keeps A needed for open assessment Q. The legacy profile path in the same service is a different algorithm and likewise does not implement the new contract. |
| Protected goals and local coverage | The protected set $P_U = Q_U \cup A_{\mathrm{mem},U}$ protects all own assessment and memory goals; every own content goal requires local assessment coverage within $\mathrm{Own}(U)$, without turning pure aggregates or parent syntheses into artificial contracts over descendants. | The existing automatic path starts solely from open assessment goals; joint protection of selected memory goals and own-unit assessment coverage are not yet implemented in runtime evaluation. The unit validity rules R1–R5 of 11 November 2026 have no validator yet. |

| Area | Binding specification here | Audited repository status |
|---|---|---|
| Global recognition ($M_{\text{eff}}$) | Person- and revision-bound global union $M_{\text{eff}} = M \cup \bigcup_{U \in \mathcal{U}_{\text{valid}}} I_U$ acts durably across downstream frontiers, focus switches, and cumulative progress without writing synthetic `Mastery` records. | **Not yet implemented.** Context-local legacy exemption is not the durable contract-bound set $M_{\text{eff}}$; downstream frontier, cumulative progress, contract revision binding, and historical meaning must be implemented and verified together. |
| Atomic normal form and minimality (G17) | Complete atomic normal form $R \subseteq A \times A$, ordered migration (materialize atomic prerequisites, remove cluster edges, reduce transitively), and transitive minimality as mandatory baseline before maturity level M1. | Since `15f74ef564` (on `main` with pull requests #119 and #121, 11 November 2026), `CQR-001` runs the `G17` check (atomic normal form, transitive minimality, explicit types, including required cross-subject goals) for every root of the complete canonical Gymnasium catalog and caps a failing subject at M0; hard in CI via the ten protected floors. In the status generated on 11 November 2026, the ten protected subjects plus Italian, Polish, Russian, and Czech pass; Chinese, English, French, Greek, Music, Spanish, and the Gymnasium overview root fail and report M0. Non-canonical roots have no `G17` gate; the migration order is evidenced only through the prerequisite-reduction proof registry (Appendix A). |
| Cluster satisfaction (Sat) | The normative core requires all atomic descendants $\text{Atoms}(g) \subseteq M_{\text{eff}}$. | Audited backend paths filter recursively via private helper `isCore ForPrereqs`: children with no tags or tag `GK` take precedence over other children when present; otherwise all children count. No boolean `core` field is read there. That differs from normative full-basis aggregation. |
| Memorization and card due state | Verified completion recall (`Recall Passed`) establishes memory-goal completion; subsequent SRS card due dates never revoke completed mastery, and ordinary goals carry no review-due expiry (`GD-D03`). | Legacy due-date target formulas are superseded; historical derivation from recall flags in `useSrsMastery.ts` alone does not prove full runtime conformance across all flows. |
| Recursive learning units (`Learning Unit`) | Recursive target and child hierarchy with unique direct ownership $\text{Own}(U)$, derived completion $\text{Complete}(U)$, and separate optional calendar/budget planning; technical unit locks and direct assessment entry under them remain undecided. | Binding domain-structure specification; **not yet a claimed runtime feature** or productive unit validator. R1, R3, and R4 must be checked fail-closed from the first authored unit (decision of 11 November 2026). |
| Direct cluster references in views | A composition view never turns a canonical cluster into an atom; $\text{Atoms}(g)$ is preserved (`GD-X09`). | At audited state `68db5bf499`, `LearnerService` and `Completion CanonicalGraphResolver` still treat direct `goalEntry` references to canonical clusters as childless runtime atoms; fix and native verification remain open. |

| Area | Binding specification here | Audited repository status |
|---|---|---|
| Personal Curriculum scope ($L = \text{Closure}(S)$) | Valid seed targets $S$ are closed on the unpruned canonical graph ($L = \text{Closure}(S)$), fixed independently of mastery, and assigned derived target roles for necessary prerequisite atoms. | Implemented in backend (`PersonalCurriculumLearningScope.java`) and WebGUI (`cacf323919`) with local regressions; full native, curriculum, and remote-CI acceptance plus alignment of applicability/ migration validators remain separate. Since `b511654efb` (11 November 2026), a learning context whose canonical graph is not strictly compilable keeps its reviewed seed scope without derived closure and claims no learning basis. |
| Focus and filter modes (normal vs. strict) | Level 3 focus narrows candidate goals while keeping Level 2 denominator $L$ fixed; normal mode checks prerequisites inside focus, strict mode checks globally; focus widening is offered only after actual focus completion. | Implemented in learner runtime and WebGUI over explicit $M$; does not yet evaluate $M_{\text{eff}}$ and remains subject to the `isCoreForPrereqs` and `goalEntry` boundaries above. |
| Didactic routes ($O_k \to a \to Z_k$) | Every non-exempt atomic goal lies on a direct atomic prerequisite path from an orientation anchor to a terminal autonomy goal within its route profile. | Evaluated in twelve configured QA-scope profiles across ten Gymnasium subjects (`CQR-101`–`CQR-104`) and hard-gated via `GVR-012` in Mathematics; curricula without route profiles cap at M2. |
| Assessment tasks and application coverage | Releasing assessments must require independent application of every recognized essential competence under every passing score combination (`GD-A08`). | `CQR-201`–`CQR-203` enforce `examData` presence, non-placeholder readiness, and metadata on protected floors, without proving live session grading or non-compensability of every passing rubric combination. Only for terminal assessments of canonical Economics, `CQR-104` binds every `coveredGoalIds` entry to a visible prerequisite path (related to R3). |
| Optional acquisition and time planning | Optional calendar periods verify structural completion witnesses over $B_\pi$ and $\text{Work} + \text{Reserve} \leq \text{Budget}$ only under explicitly declared estimation assumptions $\Theta_\pi$. | Optional planning profile (`GD-P01`–`GD-P11`); no productive curriculum or runtime budget gate is claimed, and `weight` never denotes minutes. |
| Cumulative progress vs. period completions | Cumulative plan progress counts $\lvert T_{\text{Plan}} \cap M_{\text{eff}} \rvert$ (with an optional clearly labelled explicit-only plan counter $\lvert T_{\text{Plan}} \cap M \rvert$), while active periods count only reliably dated explicit completions. | Current Cockpit and velocity views count explicit completion events; segmented explicit/implicit accounting awaits implementation of $M_{\text{eff}}$. |
| Maturity levels M0–M7 and quality floors | Automated phase-1 curriculum QA M0–M7 remains strictly separated from human sign-off, learner trials, and runtime conformance. | CI protects ten Gymnasium subjects via checkCurriculumMaturityFloors.ts (5 at M7, 5 at M6) plus status-file diffing; unfloored experimental landscapes remain reportable at M0, and `docs/**`-only changes are exempted in `ci.yml`. |

**Inspectable code anchors.** PersonalCurriculumLearningScope.java; CompletionEvaluator.java; LearnerService.java; useSrsMastery.ts; validateGraph.ts; Mastery.java. The linked `main` branch continues to evolve. The generated build manifest binds the exact input files read during each document build via SHA-256.

**Sources.** AGENTS.md, §2; completion-semantics-clarification-2026-11-04.md; dual-curriculum-package-implementation-status.md; PersonalCurriculumLearningScope.java; CompletionEvaluator.java.

## C Verification Strategy for Completion Semantics

The accompanying script check-assessment-model.py is a mathematical reference check written with the Python standard library alone. It verifies the formulas from Section 13 across small finite models, reads no learner data, and mutates no runtime state (`GD-D04`).

- **Finite model checking.** Every topologically ordered DAG of up to four vertices under every assignment of explicit mastery and protected sets where local content goals have complete local assessment coverage: 5 222 valid cases. Graph traversal and static transitive closure yield the identical implicit set $I_U$.
- **Minimality.** Transitive reduction is minimal and preserves closure. Redundant edges of the normalized relation do not alter recognition. The older inheritance counterexample diagnoses transitional data; this bounded model test does not by itself verify the new atomic migration priority (`GD-D05`).
- **Global monotonicity.** Under fixed valid contracts, additional explicit mastery never reopens effectively mastered work. Assessment and memory goals are never implicitly passed (`GD-D06`).
- **Recursive units.** 128 state assignments of the Mechanics example verify completion Complete($U$), own recognition $I_U$, and explicit child obligations. A pure aggregate requires no own assessment; a pure parent synthesis requires no new content goals.
- **Unit block grouping.** 948 grouping cases compare the unit-block DAG against a complete block sequence. One counterexample exhibits an acyclic atomic DAG whose coarse block summary forms a cycle; no technical entry-lock policy is derived from it.
- **Concrete counterexamples.** No stopping barrier at mastered intermediate goals; local recognition preserved despite an open later assessment; durable union across contracts, future prerequisites, intersection with planned target sets, and external support outside Own($U$).
- **Invalid inputs.** Cycles, unknown references, and uncovered local content goals fail closed as errors rather than defaulting to automatic mastery.

Diagram generation in generate-completion-diagrams.py verifies the concrete sets displayed in each figure and compares freshly rendered PNG bytes in `--check` mode (`GD-D07`). These model and diagram checks verify neither real task quality, grading reliability, identity verification, nor pedagogical effectiveness. Required independent subject-matter content review remains a separate step; runtime integration and its security, concurrency, and UI tests remain separate engineering work.

**Sources.** check-assessment-model.py; generate-completion-diagrams.py.

# D Glossary and Symbols

## D.1 Terms

The first table defines the terms in the order in which this specification introduces them and gives the corresponding German terms. The second table lists the mathematical symbols. None of these definitions introduces an additional stored outcome kind: only the binary outcome $m$ is persisted per atomic goal.

| Term | Meaning |
|---|---|
| goal (*Lernziel*, *Ziel*) | Node of the skill graph with a stable UUID identity. |
| atomic goal, atom (*atomares Ziel*, *Atom*) | Goal without `contains` children. Only atoms carry a stored learner outcome. |
| cluster goal, cluster (*Cluster*) | Goal with `contains` children; structures and aggregates content rather than being mastered directly. |
| containment (*Containment*) | The hierarchy formed by authored `contains` edges. |
| requirement, prerequisite (*Anforderung*, *Voraussetzung*) | The edge $(g, p) \in R$ is a requirement of $g$; goal $p$ is a prerequisite of $g$. |
| authored, derived (*verfasst*, *abgeleitet*) | Only the stored edges in $C$ and $R$ are authored. Everything else is computed from them. |
| inherited, effective requirement (*geerbte*, *effektive Anforderung*) | An inherited requirement is one that a goal acquires from a `contains` ancestor. Effective requirements combine a goal's own and inherited requirements. |
| atomic basis (*atomare Basis*) | For an atom, the goal itself; for a cluster, its atomic descendants. |
| atomic demand (*atomarer Bedarf*) | All atoms that a goal directly or transitively requires, independent of learner state. |
| atomic normal form (*atomare Normalform*) | The property $R \subseteq A \times A$: all authored requirements connect atomic goals. |
| target state, transitional data (*Zielzustand*, *Übergangsbestand*) | Target state is the intended state of canonical data: atomic normal form and transitive minimality. Transitional data denotes inventories that still carry cluster requirements. |
| transitive minimality (*transitive Minimalität*) | No authored requirement is already reachable through the remaining requirements. |
| motivation anchor, terminal autonomy goal (*Orientierungsanker*, *terminales Autonomieziel*) | Starting and ending endpoints of a didactic route. |
| explicit mastery, Mastery (*Mastery*) | The stored explicit outcome of an atomic goal: mastered or not yet mastered. |
| implicit mastery (*implizite Beherrschung*) | Mastery derived from passed assessments of a valid local recognition contract; for content goals it demonstrates independent application. |
| effective mastery, effectively mastered (*effektiv beherrscht*) | Explicitly mastered or implicitly mastered. Both carry equivalent global effect. |
| satisfaction (*Erfüllung*) | A goal or cluster is satisfied when its atomic basis is non-empty and effectively mastered. |
| frontier (*Frontier*) | The next available atomic goals: themselves not yet mastered, with all required atoms satisfied. |
| filter, focus (*Filter*, *Fokus*) | A filter selects a subset of the skill graph. Focus is a learner's temporary working scope (Level 3). |

| Term | Meaning |
|---|---|
| normal (*optimistic*) mode, strict (*pessimistic*) mode (*Normalmodus*, *strikter Modus*) | Normal (*optimistic*) mode checks prerequisites inside the selected scope; strict mode checks all prerequisites globally. |
| level of personalization (*Personalisierungsstufe*, *Level*) | Four narrowing levels: base curriculum, Personal Curriculum, focus with one active goal, and global mastery. |
| Personal Curriculum (*persönlicher Lehrplan*) | Committed personal goal selection including its necessary atomic prerequisites at Level 2; fixed independently of mastery under a given configuration. Existing valid recognition counts as satisfied. |
| projection, composition view (*Projektion*, *Kompositionssicht*) | A projection derives the visible tree for a resolved scope. A composition view explicitly defines that scope's reviewed upper tree structure. |
| projection role (*Projektionsrolle*) | `target`: selectable learning target of the view. `prerequisiteOnly`: subject-local prerequisite support; necessary atoms still receive an acquisition target in the full Personal Curriculum. |
| applicability (`applicability`, *Anwendbarkeit*) | Compiled dimensional metadata specifying in which scopes a goal is visible. |
| program unit, goal placement (`programUnits`, `goalPlacements`) | Organizational container such as a school year or semester; entry attaching an unchanged goal to such a unit. |
| goal type, node type (*Zieltyp*) | Pedagogical kind of an atomic goal: content goal, orientation goal, memory goal, or assessment goal. |
| content goal, ordinary subject goal (*Inhaltsziel*) | An ordinary subject-matter learning goal; labelled "ordinary" in figures. |
| verified recall, completion recall (*verifizierter Abruf*, *Abschlussabruf*) | Controlled unaided recall of a card without prompt hints. Verified completion recall across all required cards completes a memory goal. |
| direct assessment access (*direkter Prüfungszugang*) | A deliberately chosen assessment attempt is not blocked by unmastered preparation. |
| learning unit, child unit, required child (*Lerneinheit*, *Kindeinheit*, *Pflichtkind*) | Authorized educational unit with a stated learning outcome; its subunits; the mandatory subunits among them. |
| aggregate, synthesis (*Aggregat*, *Synthese*) | An aggregate bundles child units without adding its own assessment. A synthesis unit additionally assesses their integration. |
| local recognition contract (*lokaler Vertrag*) | Binds a learning unit's own goals, its curriculum revision, and the subject-reviewed application meaning of its assessments; also called an acquisition contract. |
| unit validity rules R1–R5 (*Gültigkeitsregeln*) | The five conditions of 11 November 2026 without which implicit recognition is unsound: an assessment path for every own ordinary goal (R1), reviewed coverage evidence (R2), coverage only within the assessment's prerequisites (R3), exactly one directly owning unit (R4), and a versioned contract (R5). |
| protected goal (*geschütztes Ziel*) | Assessment or memory goal; never implicitly passed. |
| open demand (*offene Nachfrage*) | A learning unit's still-open protected goals together with their full transitive atomic demand. |
| consumer (*Verbraucher*) | A goal that requires another goal, viewed from that prerequisite. |

| Term | Meaning |
|---|---|
| whole-unit prerequisite, entry lock (*Einheitsvoraussetzung*, *Einheitssperre*) | Explicitly authorized prerequisite between whole learning units; a hard technical access lock derived from it. Whether such locks exist remains undecided. |
| planning period, entry basis, completion witness (*Planungsstufe*, *Eintrittsbasis*, *Abschlusszeuge*) | Calendar interval of an optional schedule; competencies assumed present at entry; ordering that proves a period's targets can be acquired step by step. |
| planned target, actual count, cumulative progress (*Plan-Soll*, *IST*, *Gesamt-IST*) | Planned target is the scheduled progress of a named scope; actual count is achieved progress. Cumulative actual progress (*Gesamt-IST*) counts a plan's effectively mastered goals. |
| profile (*Profil*) | Named set of additional constraints that applies only where explicitly selected, such as a route, validator, or planning profile. |
| scope, QA scope (*Scope*, *QA-Scope*) | The named validation scope of a rule or route profile. |
| gate (*Gate*) | Automated check whose failure blocks a milestone or build. |
| maturity level, milestone (*Reifegrad*, *Meilenstein*) | Level M0 to M7 of automated curriculum QA; Appendix A refers to them as milestones. M7 is not human effectiveness proof. |

## D.2 Symbols

The following table summarizes all mathematical symbols across the specification (graph core in Sections 1 to 3, learner state and filters in Sections 4 and 5, Personal Curriculum in Section 7, goal types in Section 8, learning units and implicit mastery in Sections 11 and 13, planning in Section 12).

**Note on unified notation.** Earlier standalone English Markdown drafts used $D(g)$ for atomic demand, $U(T)$ for the prerequisite closure of a subset, $M_A$ for the explicit mastery set, $M_k$ and $T_k$ (or $M$ and $T$) for orientation anchors and terminal autonomy goals of a route profile, and $K$ for memory goals. The unified Single-Source specification standardizes these symbols across both Web and PDF to RequiredAtoms($g$), Closure($T$), $M$, $O_k$ and $Z_k$ (or $O$ and $Z$), and $A_{\text{mem}}$, keeping $K = G \setminus A$ unambiguous for structural clusters and $M$ unambiguous for explicit mastery. The unchanged static original illustration `06-available-next-goals.png` (Figure 8) still shows the historical label $M_A$, as noted in its caption.

| Symbol | Meaning |
|---|---|
| $G$, $A$, $K$ | Goals, structural atoms, clusters; $K = G \setminus A$. |
| $C$, $R$ | Authored edges: containment (`contains`) and direct requirements (`requires`). |
| $X^+$, $X^{-1}$ | Non-empty transitive closure and inverse relation of a relation $X$. |
| Anc($g$), Desc($g$) | Ancestors and descendants along `contains`. |
| Atoms($g$) | Atomic basis; the singleton $\{g\}$ if $g$ is atomic, otherwise its atomic descendants. |
| $R_{\text{eff}}$, $R_{\text{inherited}}$ | Effective requirements and their inherited subset; $\text{Pre}_{\text{eff}}(g)$ denotes the effective prerequisites of $g$. |
| $R_{\text{at}}$ | Fully atomically expanded effective requirement relation, also used in the completion formulas. |
| $R_A$ | Directly authored atomic requirements $R \cap (A \times A)$. |

| Symbol | Meaning |
|---|---|
| $B(g)$, RequiredAtoms$(g)$ | Atomic basis of the effective prerequisites of $g$ and full atomic demand (written $D(g)$ in earlier standalone English drafts). |
| $O$, $Z$, $E_{\text{route}}$ ($O_k$, $Z_k$) | Orientation (motivation) anchors, terminal autonomy goals, and exempt support goals (or per route profile $k$; written $M$ and $T$ or $M_k$ and $T_k$ in earlier standalone drafts). |
| $m$, $M$ | Stored binary state per atomic goal and the set of its explicit successes. $m(a) = 0$ does not rule out implicit mastery. Earlier standalone drafts and static Figures 8 and 9 write $M_A$. |
| $M_{\text{eff}}$ | Global union of explicit mastery and validly derived implicit mastery. |
| $\text{Sat}(g, M_{\text{eff}})$ | Satisfaction of an atomic goal or cluster. |
| Frontier, Frontier$_K$ | Next available atomic goals; derived cluster availability. |
| $P(S)$, $w(a)$ | Weighted progress across a goal set $S$; progress weight of an atomic goal $a$. |
| $F$, $G_F$, $A_F$ | Filter predicate, selected goals, selected atomic goals. |
| Applicability, $D$, $V_d$ | Applicability mapping, set of filter dimensions, and value vocabulary of dimension $d$. In Section 5, $Q$ denotes the active query selection. |
| Closure$(T)$ | Selected atomic goals $T$ together with their full atomic prerequisite demand (written $U(T)$ in earlier standalone drafts). |
| $S$, $L$ | Union of valid atomic seed targets across the selected reviewed subject scopes and the resulting closed Personal Curriculum learning scope $L = \text{Closure}(S)$ at Level 2. |
| Frontier$_{\text{opt}}$, Frontier$_{\text{strict}}$ | Frontier in normal (*optimistic*) mode and in strict mode. |
| Missing, MissingAtoms | Unsatisfied effective prerequisite nodes; missing atomic goals of the demand. |
| Cards$(g)$ | Fully resolved valid set of required cards of memory goal $g$. |
| RecallPassed$(g)$ | Explicitly verified successful completion recall across Cards$(g)$. |
| *score*, *passed* | Score and pass result of an assessment goal; *maxPoints* and *passingPoints* come from the scoring rubric. |
| $U$, Own$(U)$, $T(U)$ | Learning unit, directly owned atomic acquisition targets, and total recursive scope including child units; Children$(U)$ denotes its required child units. |
| Complete$(U)$ | Recursively derived unit completion; never an additionally persisted unit-mastery state. |
| $Q$, $A_{\text{mem}}$ | In Section 13, the sets of atomic assessment goals and atomic memory goals (written $K$ for memory goals in earlier standalone drafts). |
| $Q_U$, $A_{\text{mem},U}$, $P_U$ | Directly owned assessment, memory, and combined protected goal sets; $A_{\text{mem},U} = \text{Own}(U) \cap A_{\text{mem}}$. |
| $\text{Reach}^*_{R_{\text{at}}}(X)$ | The set $X$ together with all atomic goals its elements directly or transitively require. |
| $N_U$ | Open demand: full transitive demand of the unit's still-open own protected goals, without stopping at mastered intermediate goals. |
| $I_U$ | Implicit set: directly owned unprotected goals outside $M$ and $N_U$. For content goals, it witnesses locally derived independent application. Not an additional stored outcome kind. |
| Cov$(q)$ | Reviewed coverage claim of an assessment $q$: the goals whose independent application its tasks and normal grading test; under rule R3 never larger than $\{b \in A \mid (q, b) \in R^+_{\text{at}}\}$. Grants no recognition by itself. |
| $\mathcal{U}_{\text{valid}}$ | Person- and revision-bound, durably applicable authoritative local recognition contracts; not merely currently visible units or their latest revisions. |
| $T_{\text{Plan}}$, $H_{\text{gesamt}}$ | Planned target goal set and count of its effectively mastered goals. |

| Symbol | Meaning |
|---|---|
| $T_\pi$, $B_\pi$, $\ell_\pi$ | Planned acquisition targets, entry basis, and optional calendar-bound planning periods; $n_\pi$ is the number of periods and $S_i$ is the target set of period $i$. |
| $R^\pi_{\text{at}}$, $\text{Frontier}_\pi$, $\text{RequiredAtoms}_\pi$ | Plan requirement relation excluding outgoing requirements of the entry basis; corresponding frontier and atomic demand. |
| $W^{(r)}$ | Hypothetical mastery state of the witness verification algorithm after $r$ steps. |
| $\Theta_\pi$, $\widehat{t}_h(\Theta_\pi)$ | Declared estimation assumptions of the plan and the non-negative duration of activity $h$ estimated under them; $\mathcal{H}_i$ denotes the activities of period $i$. |
| Work, Reserve, Budget | Scheduled workload, protected reserve, and available time budget of a planning period. |

## E Normative Source Mapping and Illustrations

The bilingual chapters under `docs/concept/skill-graph/source/` form the sole authored Single-Source baseline for both the MkDocs web pages and the PDF compendia (source/README.md) and connect them to the specialized domain contracts in the repository (`GD-D09`). Cross-references identify domain ownership rather than claiming that all optional or conceptual parts are already deployed in production.

| Part | Normative sources | Role |
|---|---|---|
| Formal core and filters (§§1–5) | `source/chapters/00-frontmatter.{de,en}.md`, `source/chapters/01-core.{de,en}.md` (spec-manifest.json). | Canonical single-source definition of goals, `contains`, `requires`, inheritance, atomic expansion, validity, frontier, and filter modes for Web and PDF. |
| Layers and migration (§6) | `source/chapters/02-context.{de,en}.md`; General Goal System and Migration; Canonical Gymnasium Rollout. | Separation of goal, program, placement, competency-axis, and composition-view layers plus rollout policy for canonical skill landscapes. |
| Views and personal selection (§7) | `source/chapters/02-context.{de,en}.md`; View Projection and Goal Placement; Levels of Personalization. | Projection contract, `projectionRole`, personalization Levels 1–4, closed learning scope $L = \text{Closure}(S)$, and focus switching. |
| Goal types and mastery evidence (§8) | `source/chapters/02-context.{de,en}.md`; Node Types; AGENTS.md. | Pedagogical goal types (content, orientation, memory, and assessment goals), verified recall, and binary outcome semantics. |

| Part | Normative sources | Role |
|---|---|---|
| Recursive learning units (§11) | `source/chapters/03-learning-units.{de,en}.md` (source/README.md); Learning-Unit Design Criteria and Examples. | Canonical single-source definition of recursive `LearningUnits`, unique direct ownership Own($U$), derived completion Complete($U$), and undecided unit-entry policy; design criteria and worked candidates. |
| Optional time and budget planning (§§9, 12) | `source/chapters/04-planning.{de,en}.md`; Curriculum Time Axis and Pacing; Unified Learning Plan Status. | Optional calendar periods, entry basis $B_\pi$, structural completion witness, and workload budget under declared estimation assumptions $\Theta_\pi$. |
| Implicit mastery and local contracts (§13) | `source/chapters/05-assessment.{de,en}.md`; Completion Semantics Clarification (2026-11-04); Exam Preparation Planning. | Canonical single-source specification of local recognition contracts, transitive open demand $N_U$, implicit set $I_U$, and global set $M_{\text{eff}}$ for Web and PDF. |
| Provenance and learning materials (§10) | Source and Resource Links; Content Integration; Human-Readable Source Rationales. | Strict separation between official curriculum provenance (`sourceRef`), optional external content packages, and human-readable rationales. |
| Visualizations and goal books (§10) | Atomic Goal Visualizations; Learning-Goal Book and Evidence-Review Pipeline. | Governance of atomic goal visualizations and the D/P/A/M/V evidence and review pipeline up to maturity level M7. |
| Validation and curriculum QA (§§3, 10, Appendix A) | Graph Validation Rules; Curriculum Quality Maturity and Routes; Semantic Atomicity Review. | Structural `GVR-*` rules, content-quality `CQR-*` rules, promotion chain M0–M7, protected maturity floors, and semantic atomicity review. |
| Package interchange and licensing (§10) | Dual Curriculum Package Releases; MEM/FWU Roundtrip Plan; LICENSING.md. | Dual JSON and ontology release packages, roundtrip interchange boundaries, and licensing/privacy governance (`Apache-2.0` / `CC-BY-4.0`). |
| Rule registry, glossary, and build (Appendices A–F) | `source/rules-and-gates.json`, glossary-and-symbols.json, implementation-status.json, figures.json, Glossary. | 145 `GD-*` rules with traffic-light status, glossary and symbol reference, dated implementation audit, figure attribution, and single-source build contract. |

The seven static English-labelled didactic graph illustrations (`01-skill-graph-overview.png` through `07-filter-modes.png`: Figures 1, 2, 3, 4, 6, 8, and 9) originate from `docs/concept/skill-graph/assets/graph-definition/` (*SkillPilot: Graph Definition*, edition of 29 October 2026) and are embedded unchanged with bilingual captions from figures.json. The ten additional bilingual diagrams (`00-title-skill-graph.{de,en}.png` for the title page and `08-transitive-demand.{de,en}.png` through `16-recursive-learning-unit.{de,en}.png` covering transitive

demand, local recognition, protected support, plan period accounting, calendar planning, atomic expansion, inherited redundancy, shared local assessments, and recursive learning units) are rendered deterministically with Matplotlib as PNG bitmaps by generate-completion-diagrams.py. The formulas and captions are authoritative; the illustrations assert no additional edges or requirements (GD-D10). Technical prose, mathematical specifications, structured source datasets, and build scripts are licensed under Apache-2.0; own didactic illustrations are licensed under CC-BY-4.0 where applicable rights exist. Full license texts and boundaries: LICENSE, LICENSES/CC-BY-4.0.txt, and LICENSING.md. Attribution for the didactic illustrations: *SkillPilot, enpasos – Enterprise Patterns & Solutions GmbH, Graph Definition* (static original illustrations 01–07: edition of 29 October 2026; bilingual generated diagrams 00, 08–16 and single-source integration: this edition, CC-BY-4.0).

## F Building and Verifying the Specification

All bilingual web pages and PDF editions are compiled from the sole authored source under docs/concept/skill-graph/source/ (source/README.md, spec-manifest.json). The compiler docs/concept/skill-graph/build_spec.py verifies 1:1 structural parity between German (DE) and English (EN), translates chapters and structured JSON datasets into MkDocs Markdown and LuaLaTeX sources, and embeds both the static original illustrations (01–07) and the deterministically generated bilingual bitmap PNGs (00, 08–16) from docs/concept/skill-graph/assets/graph-definition/. From the repository root:

- python3 docs/concept/skill-graph/build_spec.py --web-and-tex – uses only the Python standard library to verify DE/EN AST parity and generate all bilingual MkDocs web pages plus intermediate LaTeX files under tmp/skill-graph-document/.
- python3 docs/concept/skill-graph/latex/check-assessment-model.py – runs the finite mathematical reference checks for Section 13.
- python3 docs/concept/skill-graph/latex/generate-completion-diagrams.py --check – verifies deterministic bilingual PNG diagram rendering without modifying files.
- python3 docs/concept/skill-graph/build_spec.py --all – runs the parity check, formula model verification, bilingual PNG rendering, Web Markdown emission, and multi-pass LuaLaTeX builds for both the German (skillpilot-graph-definition.pdf, skillpilot-graph-definition.de.pdf) and English (skillpilot-graph-definition.en.pdf) PDFs together with .build.json manifests (GD-D11).

Python 3 alone suffices for --web-and-tex and the formula verifier; full --all runs additionally require Matplotlib and LuaLaTeX with German (ngerman) and English (english) Babel support plus preamble packages. Multi-pass LaTeX compilation resolves cross-references, and generated .build.json manifests bind input files, SOURCE_DATE_EPOCH, tool versions, and output PDF digests via SHA-256. Byte-for-byte PDF identity across different TeX Live versions is not guaranteed. All generated Web Markdown pages, generated diagram PNGs (00, 08–16), intermediate .tex files, PDFs, and build manifests are .gitignored build artifacts and must never be edited by hand or committed to Git.

Complete document verification requires an error-free build, resolved cross-references, inspection for overfull or underfull boxes, and visual inspection of both generated PDFs (GD-D12). Technical specification prose, schemas, and build scripts are licensed under Apache-2.0 (LICENSE); own didactic illustrations are licensed under CC-BY-4.0 (LICENSES/CC-BY-4.0.txt, LICENSING.md). Build and formula checks complement document QA and replace neither runtime tests nor independent subject-matter and empirical validation.

**Sources.** source/README.md; spec-manifest.json; check-assessment-model.py; LICENSING.md.