KLT2 PIX@PEAKS Volume II/III and KLT-RBD5.10 Command Set RU/EN v166-v168

Курпишев Иван Борисович / Ivan Borisovich Kurpishev · Independent Researcher, Kaliningrad

build_id = KLT2_PIX_PEAKS_VOLUME2_PROOF_PROTOCOLS_VOLUME3_NAPG_COMMANDS_RU_EN_v166_v168
short_name = KLT-VOL2-PROOF-VOL3-NAPG-KLT510-COMMANDS_v166_v168
status = proof_protocols_volume3_napg_command_set_without_truth_status
formula = Volume II chapters 7-9 + Volume III NAPG formal opening + KLT-RBD5.10 command-set release != truth-status

Layer	Content	Status
v166	Volume II chapters 7-9: proof protocols, examples, counterexamples	formal monograph layer
v167	Volume III opening: NAPG formal core and stratified-time bridge	formal bridge layer
v168	KLT-RBD5.10 dashboard package and command set	software protocol layer
Truth-status	No promotion; all gates remain explicit	0 promoted

Control formula

```
Volume II proof protocols + Volume III NAPG formal opening + KLT-RBD5.10 command-set release != truth-status
```

0. Редакционный статус v166-v168

Точка v166-v168 продолжает переписывание многотомной МОНОГРАФИИ ЛОГИКА КУРПИШЕВА после v163-v165. Новая сборка не заменяет старые материалы и не вытесняет сайт; она добавляет доказательный слой, начало формального тома III и машинный командный слой KLT-RBD5.10. Человечество снова просит порядок в архиве из тысяч страниц, и, что особенно забавно, оно право.

Нижняя ось проекта сохраняется без изменения: C@C -> Rep(R,I,U;D) -> lambda-truth -> Evidence-D -> CGI*/Status -> RBD/RPD -> KLT. Формула Truth(Rep) <=> cr(U,I;R,D) = -1 остаётся авторской проективно-гармонической нормировкой, но теперь каждый переход обязан иметь proof-protocol, morphism-gate и status-ledger.

v166 завершает первый смысловой блок тома II: доказательные протоколы, примеры и контрпримеры. v167 открывает том III: NAPG formal core, stratified-time bridge, Hodge/associator separation и reduced obstruction discipline. v168 переводит всё это в командную архитектуру KLT-RBD5.10: import/export, validation, proof-ledger, site-safe release и rollback.

Главная охранная формула сборки: красивая структура, музыкально-подобная lambda, локальный Fano-like рисунок, удачный dashboard и даже хороший PDF не дают truth-status. Статус возникает только через закрытые gates: D/Dom, Stat, Null, Blind, Fano/Morphism, Proof, Repro.

v166. Volume II, главы 7-9: доказательные протоколы, примеры и контрпримеры

Глава 7 тома II вводит proof-protocol как самостоятельный математический объект. В ранних версиях проекта доказательная дисциплина уже присутствовала через D, Dom, gap-node и proof-ledger. Теперь она получает формальную роль: всякое утверждение должно быть не только записано, но и помещено в карту допустимых доказательных ходов.

Определение. ProofProtocol_KLT есть семёрка PP=(Claim, Dom, D, Rules, Objects, Morphisms, Status), где Claim - формулируемое утверждение, Dom - допустимый домен, D - достаточное основание, Rules - разрешённые правила вывода, Objects - типизированные объекты, Morphisms - сохраняющие карты, Status - текущая ступень в claim-status ladder.

Claim-status ladder фиксируется так: source-note -> definition-core -> construction -> lemma-candidate -> theorem-candidate -> conditional theorem -> formal theorem under axioms -> review-level empirical candidate -> publication_verified_status. Переход через ступени запрещён без явной записи основания. Это не бюрократия, а иммунная система проекта.

Теорема T-v166-001 (D-free harmonic closure creates a gap). Если formula-chain step использует cr(U,I;R,D)=-1 как truth-authorization, но D отсутствует, пусто, неоднозначно или не связано с Dom, то KLT/RPD-аудит обязан создать GAP-D-MISSING или GAP-DOM-MISSING, а не повышать claim-status.

Доказательство. По определению Reper-компонент D является sufficient foundation. Если D отсутствует, Reper-четвёрка вырождается в флаг R-I-U без достаточного основания. Следовательно, harmonic closure может быть лишь геометрическим мотивом, но не доказательной авторизацией. ProofProtocol не имеет входа D; значит, Status не может перейти в theorem/proof state. QED.

Глава 8 посвящена примерам. Пример в математике: формульная цепочка использует проективное отношение и объявляет результат теоремой, но не указывает admissible domain. KLT-RBD создаёт gap-node, а не theorem-node. Пример в физике: HolText_D proxy из внешнего arXiv-текста показывает петлевой мотив, но не является geometric Hol_D без numeric loop certificate, Reper connection и faithful contraction. Пример в химии: совпадение lambda около 1/2 или 2/3 отмечается как musical-like artifact cue, но не как реакционная истина.

Глава 9 вводит контрпримеры. Контрпример 1: семь локальных узлов образуют Fano-like рисунок, но нет compatible identification maps. Итог: local carrier, not global Fano carrier. Контрпример 2: dashboard показывает зелёный статус, но нет external reproduction. Итог: interface_ok, status_blocked. Контрпример 3: смета содержит красивую структуру поставок, но отсутствуют акты, договоры или проектные листы. Итог: risk-report, not legal accusation.

Теорема T-v166-002 (Example cannot promote domain status). Один пример или один контрпример может изменить локальный blocker-status, но не может повысить доменный publication_verified_status без класса C, prelock, null family, multiplicity plan и independent reproduction. Доказательство следует из Evidence-D constitution: статус домена является функцией класса и gates, а не одиночного впечатления.

Теорема T-v166-003 (Counterexample discipline theorem). Если существует counterexample, нарушающий один из gates для заявленного claim, то claim переводится не в false-by-rhetoric, а в refined-status: rejected, quarantined, domain-limited или requires-rebuild. Это важно: проект не должен уничтожать сильную идею из-за одного сбоя, но обязан ограничить её область.

v166.7.1. D/Dom-дисциплина как защита от ложной очевидности

Подраздел 7.1 раскрывает d/dom-дисциплина как защита от ложной очевидности. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение

проекта, а его доказательная архитектура.

v166.7.2. Proof-object и различие между доказательством, расчётом и отчётом

Подраздел 7.2 раскрывает proof-object и различие между доказательством, расчётом и отчётом. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.7.3. Claim-status ladder как архитектура честного продвижения утверждений

Подраздел 7.3 раскрывает claim-status ladder как архитектура честного продвижения утверждений. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.8.1. Пример формульной цепочки с отсутствующим доменом

Подраздел 8.1 раскрывает пример формульной цепочки с отсутствующим доменом. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если

формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.8.2. Пример физического review-signal без empirical promotion

Подраздел 8.2 раскрывает пример физического review-signal без empirical promotion. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.8.3. Пример химического музыкально-подобного отношения без DHC-status

Подраздел 8.3 раскрывает пример химического музыкально-подобного отношения без dhc-status. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.8.4. Пример сметного разрыва в проектном пакете

Подраздел 8.4 раскрывает пример сметного разрыва в проектном пакете. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.9.1. Контрпример к автоматической глобализации Fano-like структуры

Подраздел 9.1 раскрывает контрпример к автоматической глобализации fano-like структуры. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.9.2. Контрпример к статусу через интерфейс

Подраздел 9.2 раскрывает контрпример к статусу через интерфейс. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate,

domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v166.9.3. Контрпример к междоменной аналогии без morphism

Подраздел 9.3 раскрывает контрпример к междоменной аналогии без morphism. В системе KLT-RBD всякое утверждение рассматривается как узел, имеющий не только текст, но и источник, домен, карту зависимостей, статус и блокирующие условия. Поэтому даже сильное совпадение не получает право быть доказательством, пока не закрыта цепочка Dom -> D -> proof-object -> status-ledger.

Для человеческого читателя это может выглядеть избыточным. Но именно избыточность спасает метод от двух обычных бед: от рекламной инфляции и от академической невнятности. Если формула, граф или численная структура действительно важны, они выдержат проверку через доказательные ворота. Если не выдерживают, они остаются полезными как review-candidate.

В практической реализации KLT-RBD5.10 каждый такой подраздел имеет машинный образ: строка в theorem_candidate, связанный proof_object, список blockers, evidence_gate, domain_registry и source_register. Текст монографии и база данных должны говорить одно и то же, иначе получится не теория, а хорошо отформатированный спор с самим собой.

Вывод подраздела: локальная красота допускается, локальное совпадение фиксируется, но статус поднимается только через воспроизводимую карту оснований. Так строится не украшение проекта, а его доказательная архитектура.

v167. Volume III opening: NAPG formal core and stratified-time bridge

v167 открывает том III многотомника. Его задача - перенести НАPG из набора сильных авторских идей в формальный слой, который может быть предъявлен внешнему математику. Здесь особенно важно не смешивать три разные вещи: Hodge star, associator star и обычное умножение. Один символ * на всё подряд удобен только тому, кто хочет потом героически распутывать собственные узлы.

Примитивный NAPG-датум фиксируется как P_NAPG=(K, A^bullet, odot, G), где K - базовое поле или допустимая категория скаляров, A^bullet - градуированная алгебра/модуль объектов, odot - допустимая бинарная операция, G - группа или группоид допустимых преобразований. Ассоциатор определяется как Ass_odot(x,y,z)=(x odot y) odot z - x odot (y odot z).

Квадратичная карта препятствий Psi_G^(2) переводит допустимые деформации в слой obstruction. Admissible layer X_adm^(2) задаёт область, где вычисление имеет смысл. Obstruction quotient O_B фиксирует то, что не убивается тривиальными калибровочными заменами. Bridge Pi_E: O_B -> H^3 задаёт осторожную связь с внешним когомологическим языком.

Стратифицированно-временной мост строится не как метафизическая декларация, а как стандартная оболочка: X - Hausdorff paracompact stratified space, pure strata T^k, cumulative filtration, incidence category Str(X), packet category Pack, Hodge-compatible restrictions. Том III должен начинаться с этого, а не с космологической поэзии, как бы ни хотелось вселенной красиво войти в текст.

Теорема T-v167-001 (Hodge/associator separation). Если Hodge star используется как metric-dual operator, а associator star используется как printed name for R□R or packet associator layer, то любые морфизмы NAPG обязаны сохранять это различие. Иначе notation_blocker создаётся автоматически, а claim переводится в needs-notation-repair.

Теорема T-v167-002 (Stratified packet realization bridge). Пусть над каждой чистой стратой T^k задан stratumwise packet realization datum с Hodge operator и совместимыми restriction maps. Тогда packet object $X*Y$ может быть реализован стратумно, если его morphisms commute with restrictions and Hodge operators. Эта теорема не доказывает физику времени; она создаёт корректную геометрическую оболочку для последующих физических редукций.

Теорема T-v167-003 (Obstruction status boundary). Ненулевой reduced obstruction quotient O_B или O_{red}^3 может дать formal obstruction statement, но не empirical claim. Для empirical claim требуются источник, домен, измерение, uncertainty ledger, matched-null, blind channel and reproduction. Так математическая строгость не превращается в физическую самоуверенность.

v167.0.1. Назначение тома III и граница с томом II

Подраздел 0.1 развивает тему: Назначение тома III и граница с томом II. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.1.1. Стандартная стратифицированная оболочка

Подраздел 1.1 развивает тему: Стандартная стратифицированная оболочка. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.1.2. Чистые страты, фильтрация и cumulative interpretation

Подраздел 1.2 развивает тему: Чистые страты, фильтрация и cumulative interpretation. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не

разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.1.3. Категория Str(X) и инцидентность слоёв

Подраздел 1.3 развивает тему: Категория Str(X) и инцидентность слоёв. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.2.1. Hodge packet realization datum

Подраздел 2.1 развивает тему: Hodge packet realization datum. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.2.2. Packet morphisms and restriction compatibility

Подраздел 2.2 развивает тему: Packet morphisms and restriction compatibility. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до

использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.3.1. NAPG datum and admissible binary operation

Подраздел 3.1 развивает тему: NAPG datum and admissible binary operation. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.3.2. Associator and quadratic obstruction

Подраздел 3.2 развивает тему: Associator and quadratic obstruction. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.3.3. Reduced tangent quotient H2_red

Подраздел 3.3 развивает тему: Reduced tangent quotient H2_red. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.3.4. Reduced obstruction quotient O3_red

Подраздел 3.4 развивает тему: Reduced obstruction quotient O3_red. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.4.1. Bridge Pi_E to H^3 and its limitations

Подраздел 4.1 развивает тему: Bridge Pi_E to H^3 and its limitations. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v167.4.2. No physical promotion from obstruction alone

Подраздел 4.2 развивает тему: No physical promotion from obstruction alone. Он включается в начало тома III как формальный вход для NAPG. Здесь все объекты должны быть определены до использования: страта до пакетного объекта, Hodge operator до Hodge packet, бинарная операция до ассоциатора, admissible layer до quotient. Такой порядок скучен, зато он не разваливается под первым вопросом рецензента.

Внутренняя логика подраздела такова: сначала задаётся base category, затем object layer, затем morphism layer, затем obstruction layer, затем status layer. Если любой из этих слоёв отсутствует, claim остаётся construction или candidate. Это правило синхронизирует монографию, RBD-базу и программу KLT-RBD5.10.

Для философского горизонта проекта это не ослабление, а усиление. Философская глубина становится устойчивой только тогда, когда имеет математический каркас. Без каркаса остаётся красивый туман, а туман, как известно, плохо проходит peer review, даже если очень старается.

Машинная запись подраздела: каждая definition получает source_id, каждая theorem получает proof_object_id, каждая bridge construction получает morphism_gate, а каждая незакрытая аналогия получает blocker_id. Так том III становится не просто текстом, а слоем KLT-RBD.

v168. KLT-RBD5.10 dashboard package: import/export command set and site-safe release

v168 переводит математическую дисциплину v166-v167 в командную архитектуру KLT-RBD5.10. Цель: программа должна уметь читать проектный корпус, строить Reper-карты, сохранять proof-ledger, экспортировать отчёты и готовить сайт без разрушения существующего архива. Удивительно, но программам тоже полезно не терять прошлое.

Командная модель задаётся так: klt-rbd510 init-db; import; normalize; build-reper; scan-lambda; run-null; run-blind; audit-fano; proof-ledger; status-ledger; export-report; export-site-delta; qa-linkcheck; pack; rollback. Каждая команда должна оставлять след в run_ledger и hash_ledger.

Import layer принимает text, csv, json, sqlite, pdf-index, docx-index and zip-manifest. Для PDF и DOCX полный текст не обязан перераспределяться в публичном site delta; достаточно хранить source route, hash, extracted units and derived cards. Это защищает и авторские права источников, и размер архива.

Export layer обязан соблюдать правило сайта: add-only, root Index.html/index.html from canonical Index.docx, no standalone HTML preview links in assistant answer, zip size <= 250 MB, no loss of previous pages, fallback links where necessary. Это не прихоть, а накопленная мудрость после того, как HTML уже несколько раз пытался устроить маленькую цифровую революцию.

Теорема T-v168-001 (Command does not promote status). Ни одна команда dashboard, export или pack не имеет права повышать publication_verified_status. Команда может только вычислить признаки, записать gates, показать blockers, собрать отчёт и сформировать release. Статус повышается только status operator Gamma после закрытия всех gates.

Теорема T-v168-002 (Site-safe release). Site delta является корректным, если он сохраняет root template, добавляет новые блоки без удаления прежних, не содержит пустых href, не превышает 250 MB, имеет linkcheck missing_count=0 на overlay-base, включает SHA/QA ledger и не выдаёт отдельные HTML-preview как пользовательские ссылки.

Теорема T-v168-003 (Rollback reproducibility). Если release нарушает linkcheck, hash-ledger, root-template или status discipline, то rollback target выбирается не произвольно, а по nearest valid release: v159R root restore, v160-v162, v163-v165, затем v166-v168. Такой rollback не стирает историю, а сохраняет последовательность.

v168.command.init-db

Команда klt-rbd510 init-db предназначена для следующей операции: создать SQLite-схему KLT-RBD5.10 с таблицами source_register, domain_registry, observation, reper_card, evidence_gate, proof_object, theorem_candidate, blocker, run_ledger.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.import

Команда klt-rbd510 import предназначена для следующей операции: загрузить источник или manifest: text/csv/json/sqlite/zip/pdf-index/docx-index; создать source-bound запись без повышения статуса.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.normalize

Команда klt-rbd510 normalize предназначена для следующей операции: перевести входные единицы в normal form, сохранить units, dates, values, references, uncertainty, document anchors.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.build-reper

Команда klt-rbd510 build-reper предназначена для следующей операции: создать Rep(R,I,U;D) cards для утверждений, формул, сметных строк, физических, химических, биологических и DNA-объектов.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.scan-lambda

Команда klt-rbd510 scan-lambda предназначена для следующей операции: вычислить lambda, delta_truth и musical-like cue flags без truth-status inflation.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.run-null

Команда klt-rbd510 run-null предназначена для следующей операции: запустить prelocked matched-null или пометить null_model_blocker.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.run-blind

Команда klt-rbd510 run-blind предназначена для следующей операции: запустить hidden-channel protocol или пометить not_blind_prediction.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.audit-fano

Команда klt-rbd510 audit-fano предназначена для следующей операции: проверить local Fano-like structures и explicit morphism closure.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.proof-ledger

Команда klt-rbd510 proof-ledger предназначена для следующей операции: построить proof objects, dependency graph, theorem-candidate cards and proof blockers.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.status-ledger

Команда klt-rbd510 status-ledger предназначена для следующей операции: применить Gamma: D/Dom x Stat x Null x Blind x Fano x Proof x Repro -> Status.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.export-report

Команда klt-rbd510 export-report предназначена для следующей операции: собрать PDF/MD/HTML-внутренний отчёт; в ответе пользователю HTML не выдаётся отдельной ссылкой.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.export-site-delta

Команда klt-rbd510 export-site-delta предназначена для следующей операции: собрать add-only сайт, root Index from Index.docx, klt/index table row, d/db/monograph/program files.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.qa-linkcheck

Команда klt-rbd510 qa-linkcheck предназначена для следующей операции: проверить локальные href в overlay-base and delta.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.pack

Команда klt-rbd510 pack предназначена для следующей операции: создать FULL_PACKAGE and SITE_DELTA, each under 250 MB.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

v168.command.rollback

Команда klt-rbd510 rollback предназначена для следующей операции: вернуться к nearest valid release without deleting corpus history.

Обязательное правило: команда пишет run_id, source_hash или data_hash, сохраняет входной статус и не подменяет отсутствие доказательства красивым выводом. Если команда не может

выполнить доказательное условие, она создаёт blocker, а не делает вид, что всё прошло.

В интерфейсе dashboard результат команды должен показываться в трёх слоях: machine status, human explanation, next action. Это нужно, чтобы бухгалтер, инженер, математик и администратор сайта видели не загадочный код, а понятный маршрут исправления.

Вывод: KLT-RBD5.10 строится как SQLite-first программа с воспроизводимыми командами, а не как декоративная оболочка вокруг монографии. Монография объясняет метод; программа обязана выполнять его дисциплину.

Финальная фиксация v166-v168

v166 = Volume II chapters 7-9: proof protocols, examples and counterexamples. v167 = Volume III opening: NAPG formal core and stratified-time bridge. v168 = KLT-RBD5.10 dashboard package: import/export command set and site-safe release.

truth_status_promoted_count = 0. publication_verified_status_count = 0. global_Fano_carriers = 0. Standalone HTML preview export in assistant answer = disabled. Site root source = Index.docx. ZIP size limit = 250 MB.

Следующая линия после v168: v169-v171 должны перейти от командного слоя к фактическому Volume III chapters 1-3 full expansion, затем к KLT-RBD5.10 runnable import/export test package and industrial pilot protocol for estimates, finance, science, chemistry, biology and DNA domains.

Appendix A. KLT-RBD5.10 command registry

Command	Function	Status
init-db	создать SQLite-схему KLT-RBD5.10 с таблицами source_register, domain_registry, observation, reper_card, evidence_gate, proof_object, theorem_candidate, blocker, run_ledger	implemented_as_protocol
import	загрузить источник или manifest: text/csv/json/sqlite/zip/pdf-index/docx-index; создать source-bound запись без повышения статуса	implemented_as_protocol
normalize	перевести входные единицы в normal form, сохранить units, dates, values, references, uncertainty, document anchors	implemented_as_protocol
build-reper	создать Rep(R,I,U;D) cards для утверждений, формул, сметных строк, физических, химических, биологических и DNA-объектов	implemented_as_protocol
scan-lambda	вычислить lambda, delta_truth и musical-like cue flags без truth-status inflation	implemented_as_protocol
run-null	запустить prelocked matched-null или пометить null_model_blocker	implemented_as_protocol
run-blind	запустить hidden-channel protocol или пометить not_blind_prediction	implemented_as_protocol
audit-fano	проверить local Fano-like structures и explicit morphism closure	implemented_as_protocol
proof-ledger	построить proof objects, dependency graph, theorem-candidate cards and proof blockers	implemented_as_protocol
status-ledger	применить Gamma: D/Dom x Stat x Null x Blind x Fano x Proof x Repro -> Status	implemented_as_protocol
export-report	собрать PDF/MD/HTML-внутренний отчёт; в ответе пользователю HTML не выдаётся отдельной ссылкой	implemented_as_protocol
export-site-delta	собрать add-only сайт, root Index from Index.docx, klt/index table row, d/db/monograph/program files	implemented_as_protocol
qa-linkcheck	проверить локальные href в overlay-base and delta	implemented_as_protocol
pack	создать FULL_PACKAGE and SITE_DELTA, each under 250 MB	implemented_as_protocol
rollback	вернуться к nearest valid release without deleting corpus history	implemented_as_protocol