

IMPORT-02: точное восстановление данных и связка с KASVD R4M.1

Курпишев Иван Борисович

Проект «ЛОГИКА КУРПИШЕВА 2»

12 октября 2026 года · версия 0.1.0

Результат этапа: создан контроллер подготовки, подтверждения и проверки восстановления, связанный с типизированным ядром R4M.1. Семантический отчет выдается отдельно от исходных записей. Действующие APK/EXE, ключи активации и пользовательские базы на этом этапе не изменены.

Состояние: реализация проверена с модельным портом хранения; полный пользовательский маршрут с настоящей IndexedDB остаётся открытым. Ниже разделены доказанные свойства абстрактной операции, выполненные программные проверки и неисполненные платформенные испытания.

1. Источники и граница этапа

Нулевые точки: KASVD R4M.1 v0.1.1 (12.10.2026) и IMPORT-01 v0.1 (05.10.2026). Родительские архивы сохранены без изменений. Три исполняемых родительских модуля включены побайтно: backup-core.js, restore-core.js, typed-transfer-core.js. Контрольные суммы привязаны к конкретным архивам, а не только к названиям [S1, S2].

R4M.1 описывает конечную типизированную булеву модель, а не полную семантику естественного языка. Новый контроллер не распознаёт произвольный текст и не доказывает правильность разметки Канта. Он проверяет только явный контракт выбранной записи. Недоопределенные контексты не получают выдуманных значений.

Принятый в Томе V барьер сохранён: геометрический инвариант, достаточное основание и эмпирическое подтверждение различаются. Восстановление копии не превращается в новое независимое наблюдение [S3].

2. Контракт программного перехода

```
EMPTY -> PREPARED -> BACKUP_CONFIRMED
      -> COMMITTING -> COMMITTED_VERIFIED
                        | COMMITTED_UNVERIFIED
Other outcomes: COLLISION_BLOCKED, PREPARE_FAILED, WRITE_FAILED, CANCELLED.
```

Операция prepare снимает вход до первого асинхронного ожидания, проверяет схему и хеши, читает исходное состояние цели, строит план и предзаписную копию. На этом шаге нет записи в базу. Режим выбирается при создании сессии; его смена требует новой сессии.

merge-identical добавляет новые идентификаторы и сохраняет совпадающие записи. Разные значения при одном идентификаторе блокируют слияние. replace-all заменяет только объявленные хранилища и требует отдельного подтверждения удалений и замен. Схема базы не создаётся и не мигрирует контроллером.

Сохранение копии и доказательство чтения копии разделены. Кнопка скачивания не повышает состояние сессии. Пользователь должен снова выбрать сохраненный файл; контроллер сравнивает его

полный текст и SHA-256 с точными байтами предзаписной копии. Даже успешное чтение не является доказательством сохранности носителя при отключении питания [S4].

Перед записью требуются разрешение приложения-хозяина, прочитанная копия, явный код подтверждения текущего плана и, при замене, отдельная отметка согласия. Внутри родительской транзакции цель перечитывается; изменение с момента подготовки отменяет операцию. Источник основного транзакционного требования - спецификация IndexedDB [S5].

После завершения транзакции контроллер снова читает цель и сравнивает нормализованные по идентификаторам записи с ожидаемым состоянием. Ошибка этого чтения или расхождение даёт COMMITTED_UNVERIFIED. Это не выдаётся за откат: запись уже могла завершиться. Автоматическое восстановление поверх возможных позднейших правок не выполняется.

3. Математика точного восстановления

Рассматриваем базу как конечное частичное отображение из пары «хранилище, идентификатор» в полное JSON-значение записи. Обозначим ее D , а копию - B . Равенство сравнивает все поля и вложенные значения; порядок свойств объекта не существен. Порядок элементов внутри сохраненного массива существен.

$\text{Compat}(D,B) \iff \text{for all } k \text{ in } \text{dom}(D) \text{ intersect } \text{dom}(B): D(k) = B(k).$
 $M(D,B) = D \text{ union } B, \text{ if } \text{Compat}(D,B); \text{ otherwise } \text{BLOCKED}.$
 $R(D,B) = B, \text{ within the declared database/store scope}.$

Предложение 1 (сохранение и идемпотентность). Если $\text{Compat}(D,B)$, то $M(D,B)$ сохраняет все записи обоих аргументов, а повторное применение той же копии не меняет результат. Доказательство: значение каждого ключа берется из одного аргумента, а на пересечении значения равны. Повторное объединение с B не добавляет ни ключа, ни нового значения. Для замены $R(R(D,B),B)=B$. На совместимых аргументах слияние коммутативно. Обе скобки тройного слияния определены тогда и только тогда, когда все три аргумента попарно совместимы; тогда обе равны их объединению. Это следует из поключевого равенства на пересечениях.

Предложение 2 (инвариантность семантического расчёта). Пусть Q - фиксированный детерминистический анализатор полного контракта записи. Если восстановление сохранило запись x без преобразования, то $Q(x)$ до и после восстановления совпадает. Доказательство: аргумент функции и версия алгоритма не изменились. При этом не доказаны истинность исходных посылок и верность связи контракта с внешним источником.

$\text{ExactRestore}(x) = x \implies Q(\text{ExactRestore}(x)) = Q(x).$
Same declared premises $\Gamma \implies \text{same feasible set } V(\Gamma).$

Следствие: само по себе восстановление не сужает предсказательный коридор и не увеличивает число независимых оснований. Для сужения нужно новое обоснованное ограничение в той же сигнатуре, а не вторая копия того же данного. Это охраняет предсказательный метод от ложной уверенности при синхронизации архивов.

Предложение 3 (два типа отказа). При атомарном порте хранения отказ с отменой транзакции до ее завершения не меняет базу; отказ проверки после завершения не позволяет сделать тот же вывод. Доказательство: в первом случае используется условие атомарности и подтвержденная отмена транзакции; во втором этого условия уже недостаточно, поскольку запись могла пройти. Поэтому второму случаю соответствует отдельный исход COMMITTED_UNVERIFIED, а не обещание автоматического отката.

Эти предложения применяют обычные свойства частичных функций и объединения совместимых данных. Библиографический приоритет для них не заявляется. Вклад этого этапа в проект Курпишева - исполнимая связка точного восстановления, типизированного проверяемого контракта и отдельного доказательного отчета.

4. Подключение R4M.1 и пользовательский интерфейс

Анализ требует явной привязки {store, key, field}. В текущей версии разрешено поле kasvd_contract и до 64 привязок. Каждый отчет содержит хеш конкретной записи и результат R4M.1. Он не вставляется в repers/runs/snapshots и не переписывает исходный статус. Обнаруженная несовместимость посылок не превращает архивное восстановление в редактирование источника.

Локальный STAND.html предназначен для явных синтетических примеров. START_LOCAL.py запускает только статический сервер на 127.0.0.1. Инициализация теста допустима только если соответствующей базы еще нет; автоматического удаления существующих данных нет. Стенд не следует встраивать в действующий анализатор до его собственной приемки.

Хранилища родительского контракта: KLT_RBD_ARBD_M2/snapshots для веб-профиля; kltbavt/repers,runs для автономного. Встроенные repers.jsonl и edges.jsonl проверяются по родительским хешам, но не переписываются. Новая оболочка ограничена 8 MiB входной копии и 10 000 записей на хранилище. Это программный лимит, не аттестация предельных объемов на телефоне.

5. Проверки и невыполненные испытания

Новые 38 проверок исполняют реальный код контроллера, родительский валидатор копий и реальный модуль R4M.1, но используют явный модельный порт хранения. Проверены блокировка без разрешения, подмена копии, конфликты, несвежая цель, повторное использование плана, отмена, отзыв разрешения, сохранение неизвестных полей и отдельный исход ошибки после записи. Они не проверяют имплементацию IndexedDB.

Отдельная Python-проверка перебирает девять баз, 81 пару и 729 троек. Проверены законы сохранения, идемпотентности и частичной ассоциативности. Это конечная проверка абстрактной модели; общие доказательства приведены в разделе 3. Родительские 54 теста R4M.1 запущены повторно, как и восемь булевых примеров в отдельной реализации.

В документе Chromium в памяти прошли восемь проверок интерфейса: запрет записи до подтверждения, чтение JSON, сброс старого выбора при ошибке нового файла и отсутствие горизонтального переполнения при ширине 320, 390, 768 и 1365 пикселей. Это не полный маршрут с базой.

Для 39 сценариев настоящей IndexedDB подготовлен запускаемый набор. В текущей среде они не выполнены: переход на локальный адрес возвращает ERR_BLOCKED_BY_ADMINISTRATOR. Ограничение не отключалось. Родительские 26 тестов IMPORT-01 в Node/fake-indexeddb также не повторялись: загрузка закрепленной зависимости недоступна. Они не включены в число текущих успехов.

Не выполнены приемка Android WebView и Windows Electron, проверка квоты носителя, отключения питания, долговременного хранения и настоящего межоконного конфликта. Важные следующие сценарии: отказ второго хранилища с откатом первого, посторонняя запись между подготовкой и применением, закрытие процесса, переоткрытие базы и восстановление предзаписной копии.

6. Порядок интеграции и сохранения

Следующий допуск IMPORT-02A требует реального прохода 39 транзакционных сценариев и работы полного интерфейса на отдельном тестовом профиле. Затем надо снова открыть актуальные исходники анализатора, привязать authorize к его существующему входу и подписанной активации, проверить одинаковый результат сетевого и автономного адаптеров. Заглушка authorize: () => true допустима только в изолированных тестах; она не является механизмом лицензирования.

Только после этих проверок формируется новая подписанная сборка APK и согласованные сборки Windows. Выпуск 1.05.5/1.05.05, кабинет владельца, его ключи и исторические тома на этом шаге не меняются. Пакет является добавочным; возврат к родителю означает исключение новой оболочки, а не замену всех проектных источников.

```
SCIENCE=ABSTAIN; EXT_RUN=NOT_RUN; truth_layer_promotion=0  
production_integration=NOT_RUN; new_apk_exe=NOT_BUILT  
semantic_source_review=NOT_RUN; r4k_semantic_run=NOT_RUN
```

Источники

[S1] KASVD_R4M1_20261012_v0_1_1_FULL.zip; CHECKPOINT.json; typed-transfer-core.js. SHA-256: dc212e0ae56e561cd1d2c4e7dc058a18803003bd370908aa133a238ba1a68eae.

[S2] KLT_M4_IMPORT01_CORE_20261005_v0_1.zip; restore-core.js; backup-core.js. SHA-256: 53ecd45c04206dbdfa5ab9dcce68742a60646be7304e14694edad69a0240abe9.

[S3] Курпишев И. Б. Базовый проект KLT-RBD. Том V, T5-AUDIT-MATH-v0.4, паспорт выпуска, стр. 2.

[S4] Chrome for Developers. The File System Access API: simplifying access to local files.
<https://developer.chrome.com/docs/capabilities/web-apis/file-system-access> (accessed 2026-10-12).

[S5] W3C. Indexed Database API 3.0, sections 2.7.1, 5.4, 5.5. <https://www.w3.org/TR/IndexedDB/> (accessed 2026-10-12).