Игорь Сергеевич
Петренко

AIO-архитектура и машинно-нативный контент

Энтропийно-управляемая информационная архитектура: Единый фреймворк для машинно-оптимизированной доставки и извлечения контента

Игорь Сергеевич Петренко

AIO-архитектура и машинно-нативный контент

http://www.litres.ru/pages/biblio_book/?art=73385773
ISBN 9785006928107

Аннотация

Данная монография развивает рецензируемую научную статью в полное исследование о машинном чтении веба. Показано, почему шум и человекоцентричные страницы ломают RAG и агентов, и введён ECIA – измеримый контракт контента. Описаны конверт, публикация AIO и деградация ECR, метрики, бенчмарки, экономический эффект и референсная реализация. С доказательствами, спецификациями, данными и воспроизводимыми результатами.

Содержание

AIO-архитектура и машинно-нативный контент

Игорь Сергеевич Петренко

ISBN 978-5-0069-2810-7
Создано в интеллектуальной издательской системе Ridero

О книге

Эта монография представляет всеобъемлющий теоретический и практический фреймворк для оптимизации цифровой информационной архитектуры под потребление машинами. Она посвящена энтропийно-управляемой информационной архитектуре (Entropy-Controlled Information Architecture, ECIA) как подходу, направленному на повышение надежности машинного чтения цифровых источников. В центре внимания находится не «улучшение моделей» как таковых, а улучшение среды, в которой модели и автономные агенты читают: структуры, связности, воспроизводимости и проверяемости входных данных.

Опираясь на Теорию глупости (Петренко, 2025/2026), монография формирует единую рамку, связывающую когнитивную нагрузку, информационную энтропию и обработку информации машинами, с непосредственным практическим применением как для издателей контента, так и для разработчиков систем ИИ. Книга переносит ключевые идеи фреймворка AIO/ECR из формата статьи в формат последовательного академического исследования: с историко-технологическим контекстом, терминологическим аппаратом, формальными моделями, описанием протоколов внедрения и разбором отказов.

Книга основана на рецензируемой научной работе:

Наука, образование и культура. №1 (75), 2026. ISSN 2413—7111.

Энтропийно-Управляемая Информационная Архитектура (ECIA).

Единый фреймворк для машинно-оптимизированной доставки и извлечения контента, основанный на Теории Глупости. Игорь Сергеевич Петренко (Январь 2026).

Теоретическим основанием фреймворка также является рецензируемая работа:

Теория глупости: Формальная модель когнитивной уязвимости (Общая теория глупости). Игорь Сергеевич Петренко (Декабрь 2025).

Опубликовано в научном журнале:
Наука, техника и образование. 2025. №4 (100). ISSN 2312—8267.

Опубликовано в сборнике трудов:
CXI Международная научно-практическая конференция «International Scientific Review of the Problems and Prospects

of Modern Science and Education». США, Бостон, Декабрь 16—17, 2025. ISBN 978-1-64655-169-9.

Центральная идея

Ключевой тезис книги состоит в следующем: качество рассуждения синтетической когнитивной системы существенно определяется свойствами входного сигнала. Если источник знаний организован по антропоцентричным принципам, где смысл растворен в интерфейсе, декоративных и сервисных слоях, то машинная система вынуждена тратить ресурсы на очистку и реконструкцию структуры. Это приводит к росту стоимости обработки и к росту вероятности ошибок, включая ошибки ассоциации атрибутов, пропуски условий и разрушение логических связей.

ECIA рассматривает эту проблему как архитектурную и предлагает практический ответ: сосуществование двух слоев публикации одного и того же знания. Человекоориентированный слой сохраняет интерфейс и опыт восприятия для людей. Машиноориентированный слой предоставляет детерминированное, низкоэнтропийное представление, пригодное для воспроизводимого извлечения, цитирования и проверки целостности.

Предисловие

Веб и корпоративные документы долгое время создавались в предположении, что главным читателем является человек. Это предположение сформировало нормы: тексты вокруг «главного», шаблонные блоки, навигация, юридические предупреждения, динамические компоненты. Для человеческого чтения такая избыточность часто приемлема, поскольку человек распознает контекст через визуальные и жанровые сигналы. Для машинного чтения эта избыточность превращается в измеримый налог: она занимает токены контекста, снижает эффективность внимания и повышает риск ошибок.

Нынешний технологический сдвиг заключается в смене субъекта чтения. Поисковые системы с генеративными ответами, корпоративные системы генерации с дополненным поиском и автономные агенты читают массово, систематически и по операциональным причинам. В такой среде становится недостаточно «быть понятным человеку». Источник знания должен быть также корректно читаем машиной, иначе он теряет ценность как компонент инфраструктуры принятия решений.

Актуальность и цели исследования

Актуальность работы определяется тремя взаимосвязанными факторами.

Первый фактор – рост машинного потребления контента. Машины не только индексируют, но и извлекают, суммируют, связывают и используют сведения для решений и действий.

Второй фактор – измеримость затрат. При тарифах на токены, ограничениях контекстных окон и требованиях к латентности качество структуры входа превращается в прямую экономическую величину.

Третий фактор – рост цены ошибок. В прикладных доменах ошибочно извлеченные условия, неверно связанная цена или перепутанная версия документа приводят к прямым потерям, юридическим рискам и необходимости ручной проверки, которая разрушает эффект автоматизации.

Цель монографии – сформировать единый язык описания проблемы (энтропия, шум, режимы чтения, отказные паттерны), предложить архитектурное решение уровня экосистемы (двухслойность и контент-конверт), а также дать практические протоколы внедрения как на стороне издателя, так и на стороне потребителя.

Объект и предмет

Объектом исследования является машинное извлечение и использование знаний из цифровых источников в условиях высокой информационной энтропии.

Предметом исследования является архитектурная трансформация источников из человекоцентричных представлений в машиноцентричные представления, обеспечивающие детерминированность, низкую энтропию, устойчивые якоря для цитирования, явное связывание фактов с контекстом и проверяемую целостность.

Путеводитель по книге

Книга построена как переход от постановки проблемы к формализации и затем к инженерной реализации.

Первая часть вводит исторический контекст и формирует теоретическую основу. Здесь фиксируются классы машинных читателей, природа шума, дихотомия человекоцентричной и машиноцентричной архитектуры, а также модель, связывающая шум и вероятность ошибок.

Вторая часть описывает техническое ядро ECIA: кон-

тент-конверт как универсальный формат упаковки знания и два взаимодополняющих механизма внедрения. Издательская реализация предполагает публикацию машиноориентированного слоя рядом с человеческим интерфейсом. Потребительская реализация предполагает преобразование традиционных источников в конверты на этапе поглощения.

Третья часть посвящена эмпирической валидации: измерению качества, анализу отказов и оценке экономической эффективности снижения энтропии на входе.

Четвертая и пятая части рассматривают внедрение, стандартизацию и расширения, включая доменные профили, безопасность, мультимодальные представления и перспективы развития экосистемы машинного чтения.

Как читать эту книгу

Если вас интересует теоретическая рамка и терминология, начните с первой части. Она задает определения и объясняет, почему проблема является архитектурной.

Если вас интересует инженерная реализация и практические механизмы, переходите ко второй части. Там представлены принципы контент-конверта, требования к якорям, связыванию и целостности, а также подходы к внедре-

нию в реальных системах.

Если ваш интерес связан с доказательностью, экономикой и эффектом в масштабе, начните с третьей части. Она показывает, как снижение энтропии на входе отражается на точности, стоимости и воспроизводимости.

Ограничения и допущения

Любая архитектурная теория является упрощением реальности. В данной монографии принимаются следующие допущения.

Во-первых, машиноориентированный слой не претендует на описание всего содержания в виде фактов. Он предназначен для тех частей знания, где критичны точность, воспроизводимость и проверяемость: определения, процедуры, условия, численные значения, версии, исключения и контекст истинности.

Во-вторых, качество машинного чтения зависит от домена. Одни домены естественно структурируемы, другие требуют более сложных схем. В книге рассматривается минимальный каркас, который должен быть универсальным, и допускается доменное расширение.

В-третьих, проверяемость и целостность требуют доверенной модели распространения ключей и метаданных источника. В публичном вебе и в корпоративных периметрах безопасности эти механизмы различаются. В книге фиксируются принципы, а не конкретный единый способ распространения.

В-четвертых, внедрение является социотехнической задачей. Даже если техническое решение корректно, оно должно быть совместимо с текущими практиками публикации и потребления, иначе оно не станет стандартом. Поэтому ECIA рассматривает двухслойную модель как эволюционный путь, а не как отказ от существующего веба.

Термины и обозначения

ECIA – энтропийно-управляемая информационная архитектура, единый фреймворк для машинно-оптимизированной доставки и извлечения контента.

HCA – человекоцентричная архитектура, ориентированная на визуальное восприятие и интерактивный опыт человека, допускающая высокую энтропию с точки зрения машинного чтения.

MCA – машиноцентричная архитектура, ориентирован-

ная на детерминированное извлечение и проверяемое использование сведений машинами, требующая низкой энтропии и явной структуры.

Контент-конверт – машиноориентированная упаковка знания, отделяющая смысловую полезную нагрузку от интерфейсной оболочки и фиксирующая структуру, идентификаторы, связи, метаданные и целостность.

Энтропия и шум – операциональные термины для описания количества неопределенности и нерелевантных токенов, которые система должна устранить, чтобы получить пригодный для вывода сигнал.

Эффект конфетти – структурная деградация знания при фрагментации источника без учета смысловых границ, приводящая к потере связности условий, определений и исключений.

Дальнейшее изложение опирается на данные определения. Цель фронтальной части книги – зафиксировать читательский контракт: что именно считается надежным знанием для машины, почему это не сводится к улучшению моделей, и какие архитектурные инварианты требуются для перехода к машинно-читаемой инфраструктуре.

ЧАСТЬ I: МАШИННОЕ ЧТЕНИЕ И ТЕОРИЯ ШУМА

Глава 1. Революция машинных читателей

Введение. Эволюция субъектности в цифровых информационных системах

Традиционная история интернета и веба описывается как история интерфейсов, стандартов и пользовательского опыта. В такой перспективе базовое предположение долгое время оставалось неизменным: конечным интерпретатором информации является человек. Веб-документ проектировался так, чтобы его можно было увидеть, просмотреть, понять и использовать биологическим агентом с ограниченной рабочей памятью, неустойчивым вниманием и визуально-ориентированным восприятием. Эта предпосылка определяла и выбор технических средств, и практики производства контента, и экономические стимулы экосистемы. В результате сформировалась доминирующая парадигма антропоцентричной архитектуры, в которой смысл практически всегда упакован в оболочку визуальной презентации, нави-

гации и интерактивности.

В последние годы возникла новая ситуация: важнейшим и массовым потребителем контента стал не человек, а вычислительная система, способная извлекать и обобщать смысл, а затем генерировать новые тексты, решения и действия. К таким системам относятся большие языковые модели, инструменты поиска с генеративными ответами, корпоративные системы генерации с дополненным поиском, а также автономные агенты, выполняющие задачи в веб-среде. Эти субъекты не просто индексируют страницы и не просто ищут ключевые слова. Они читают материалы как источники знаний, пытаясь извлечь факты, связи, определения, аргументы и инструкции, затем объединить их с другими источниками и выдать результат в виде ответа или действия.

Таким образом, произошла смена субъектности информационного потребления. Веб перестал быть исключительно человеческой библиотекой и стал инфраструктурой, которую ежедневно и систематически читают алгоритмы, обладающие статистической семантической компетенцией, но сталкивающиеся с фундаментальными ограничениями. Главным из таких ограничений является зависимость качества вывода от структуры входного сигнала. Если вход организован так, что полезная семантическая нагрузка растворена в большом количестве нерелевантных токенов, эффек-

тивность внимания модели падает, вероятность ошибок растет, а стоимость обработки увеличивается.

Цель данной главы состоит в том, чтобы обозначить исторический и технологический контекст перехода к эпохе машинных читателей, сформулировать, почему существующая информационная архитектура плохо соответствует их потребностям, и показать, почему этот разрыв нельзя устранить только путем наращивания мощности моделей. Глава задает проблематику и терминологический каркас, который будет развернут в последующих главах: понятие информационного шума, различие между архитектурой для людей и архитектурой для машин, а также необходимость выделения специальных каналов доставки знаний для автоматизированных потребителей.

Ключевой тезис, который проводится через всю монографию, можно выразить в следующей форме: качество рассуждения синтетической системы в значительной степени определяется не только ее внутренними параметрами, но и внешней средой, в которой она читает. Если среда имеет высокую энтропию, то даже сильная модель системно деградирует, поскольку вынуждена расходовать внимание на фильтрацию и реконструкцию смысла. Следовательно, улучшение среды чтения является столь же важной задачей, как и развитие моделей.

Историческое развитие веба и антропоцентричная архитектура

Ранний веб формировался как система гипертекстовых документов. В исходной модели документ имел относительно простую структуру: заголовки, абзацы, списки, ссылки. Семантика документа в существенной степени совпадала с его текстовой формой: содержание было написано для чтения, а разметка фиксировала структуру документа. По мере коммерциализации веба и появления масштабных рекламных моделей приоритет сместился от структурирования знания к удержанию внимания. Возникла конкуренция за клики, время на странице, глубину просмотра, коэффициент конверсии. Эти метрики стимулировали рост функционального и визуального слоя вокруг собственно содержания. Появились сложные системы навигации, блоки рекомендаций, персонализация, A/B-тестирование интерфейсов, агрессивные формы рекламных вставок, динамические баннеры, интерактивные формы и уведомления.

Параллельно происходила технологическая эволюция: страницы стали приложениями. JavaScript превратился из вспомогательного инструмента в доминирующий механизм построения интерфейса. Системы сборки, клиентская маршрутизация, гидратация, состояние приложения и асин-

хронные запросы к API стали стандартом. Для человека эти изменения часто означали более плавный и интерактивный опыт. Для машинного читателя это означало рост объема кода, усложнение извлечения текста и размывание границ между содержанием и обслуживающими компонентами.

Важная особенность антропоцентричного веба состоит в том, что многие элементы смысла представлены визуально и контекстно. Человек легко понимает, что перечеркнутая цена относится к старому тарифу, а выделенная цветом цена относится к новой. Человек понимает, что блок справа является рекламой или рекомендациями, потому что он расположен в колонке и визуально отделен. Человек понимает, что кнопка относится к конкретной карточке товара, потому что она находится рядом и визуально группируется. В машинном представлении такой визуальный контекст отсутствует или доступен только через дорогостоящую процедуру рендеринга и анализа стилей. Следовательно, сама форма организации веба создает информационный шум для алгоритмов, пытающихся извлечь факты и связи.

С середины 2010-х начались попытки частично компенсировать проблему за счет внедрения структурированных данных: Microdata, RDFa, JSON-LD и словарей вроде Schema.org. Однако эти попытки носили инструментальный характер, ориентированный прежде всего на поисковую оп-

тимизацию. Структурированные данные внедрялись не как стандартный слой истины для всех потребителей, а как вспомогательный сигнал для ранжирования. Кроме того, практика внедрения оказалась фрагментированной: одни домены (например, рецепты, вакансии, мероприятия) получили достаточно подробную разметку, другие почти не используют ее или используют с ошибками. В результате этот слой не стал универсальным интерфейсом знания для машинных читателей.

Типология машинных читателей и режимы их чтения

Для последующей архитектурной дискуссии важно различать классы машинных потребителей, поскольку у них разные требования к скорости, полноте, точности, а также к форме представления информации. Несмотря на различия, все классы объединяет общая характеристика: они потребляют содержимое не как визуальный опыт, а как источник семантической полезной нагрузки, которая должна быть извлечена и представлена в пригодном для вычислений виде.

Первый класс составляют системы поиска с генеративной суммаризацией. Они заменяют классическую модель выдачи ссылок моделью прямого ответа. В такой системе чтение веба является транзитной операцией: источники читаются быстро, частично и подчиняются задаче ответа на конкрет-

ный запрос пользователя. Главные ограничения здесь связаны с латентностью и надежностью: система должна успеть извлечь релевантные фрагменты и сформировать ответ в ограниченное время, не превысив допустимый бюджет токенов и вычислений. Это означает, что источники с высоким шумом становятся невыгодными: они требуют много времени на очистку и создают риск ошибок. В таких условиях преимущество получают источники, предоставляющие краткий и структурированный слой сведений, доступный без рендеринга и сложных эвристик.

Второй класс составляют корпоративные системы генерации с дополненным поиском, используемые для доступа к внутренним базам знаний. Они работают с документами, созданными не для публикации в вебе, а для внутреннего оборота: регламенты, политики, технические инструкции, отчеты, письма, презентации. В этой среде распространены форматы, плохо приспособленные для семантического извлечения, прежде всего PDF. Кроме того, корпоративные документы часто содержат повторяющиеся шаблоны, колонтитулы, версии, комментарии и юридические блоки, что увеличивает шум. RAG-системы, как правило, решают задачу через индексирование фрагментов текста и последующее извлечение наиболее похожих фрагментов по векторному поиску. Однако стандартная практика разбиения на фрагменты фиксированного размера порождает сбой на границах

смысловых единиц: определения отделяются от терминов, условия от следствий, исключения от правил. Далее этот отказ будет рассмотрен как эффект конфетти и как аргумент в пользу семантической сегментации на стороне источника.

Третий класс составляют автономные агенты. Это системы, которые не ограничиваются чтением ради ответа, а используют чтение как часть цепочки действий. Агент должен извлечь параметры, сравнить варианты, заполнить формы, нажать кнопки, инициировать транзакции, проверять условия. Такой режим требует значительно более строгого понимания структуры и контекста, чем генеративный ответ. Ошибка ассоциации сущности и атрибута может привести к неверному действию. Например, неверная идентификация цены или условий доставки может привести к экономическим потерям. Следовательно, для агентов критически важны детерминированные, машиночитаемые представления: четкие идентификаторы элементов, устойчивые якоря для цитирования, однозначные типы сущностей и верифицируемая целостность данных.

Архитектурный разрыв: почему веб трудно читать машинам

Архитектурный разрыв между человеческими и машинными потребителями проявляется в нескольких взаимосвя-

занных механизмах.

Во-первых, современный контент упакован в многослойную оболочку презентации. Даже если текстовая информация присутствует, она окружена большим количеством элементов, не относящихся к смыслу. Это навигация, повторяющиеся блоки, рекомендации, элементы персонализации, юридические уведомления, формы подписки, сообщения о cookies и так далее. Для человека такие элементы могут быть фоновыми и легко игнорируются благодаря зрительному вниманию и привычкам. Для машинной системы эти элементы часто неотличимы от основного содержания на уровне токенов, если отсутствует надежная семантическая маркировка.

Во-вторых, существенная часть содержимого генерируется динамически. В одностраничных приложениях текст может появляться только после исполнения JavaScript. Простое скачивание HTML может не дать текста. Машине приходится либо эмулировать браузер и выполнять скрипты, либо обращаться к внутренним API, которые не документированы и часто защищены. Это увеличивает стоимость извлечения и делает процесс хрупким.

В-третьих, сама структура HTML и DOM часто не отражает логической структуры документа. Переиспользуемые

компоненты, системные классы, абстрактные контейнеры, автоматическая генерация атрибутов делают дерево элементов скорее артефактом фронтенд-стека, чем семантической моделью. Визуальная группировка может не совпадать с вложенностью в DOM. В результате машинная система может ошибочно связать цену с неправильным товаром или перепутать подписи в таблице.

В-четвертых, информационный шум создается повторяемостью и шаблонностью. Один и тот же текст может присутствовать на сотнях страниц: политики, дисклеймеры, баннеры. В корпоративных документах повторяются колонтитулы, названия подразделений, даты версий. Векторный поиск и модели внимания воспринимают повторяющийся текст как релевантный сигнал, если он статистически встречается часто. Это приводит к загрязнению контекста и уменьшает вероятность извлечения действительно уникальной полезной информации.

В-пятых, проблемой является отсутствие стабильных якорей и однозначных ссылок на фрагменты. Человек может сослаться на абзац, цитату, страницу. Машинной системе нужны стабильные идентификаторы смысловых единиц, чтобы обеспечивать воспроизводимость, цитирование и привязку структурированных фактов к их источникам. В текущей архитектуре веба даже если присутствуют заголовки и якоря,

они нестабильны при редизайне и не соответствуют внутренним структурам извлечения.

Ограничения моделей и связь качества вывода со средой чтения

На уровне популярного дискурса распространено представление, что любые проблемы извлечения будут решены следующим поколением моделей: больше параметров, больше контекстное окно, лучше мультимодальность. Однако эмпирические исследования и практический опыт внедрения систем на базе LLM показывают, что связь между масштабом модели и устойчивостью к шуму не линейна. С ростом контекстного окна растет и объем мусорных токенов, которые можно поместить внутрь. Если вход не очищен, модель тратит дополнительные вычисления на обработку нерелевантного материала. Более того, существуют эффекты позиционного смещения: информация, находящаяся в середине длинного контекста, извлекается хуже. Этот феномен описан в работе Liu et al., 2023, и имеет прямое отношение к ситуации веб-скрейпинга, когда полезные факты часто находятся не в начале и не в конце страницы, а в середине, окруженной шаблонными блоками.

Для понимания масштабов проблемы полезно рассмотреть простую модель: любой вход можно представить как

смесь полезного сигнала и шума. Если доля шума велика, то эффективная полезная нагрузка становится малой. В таком режиме любая ошибка внимания приводит к потере факта. Вероятность ошибки увеличивается, поскольку модель должна распределять внимание между множеством токенов, из которых большинство не несет ответа. В последующих главах эта интуиция будет формализована через понятие налога на внимание и через G-модель, связывающую вероятность когнитивного сбоя с уровнем шума и эффективностью контроля внимания.

Практическая иллюстрация: типичная задача извлечения факта, например, цена тарифа или дата основания компании. На реальной странице эти данные могут присутствовать, но будут окружены навигацией, блоками рекомендаций, повторяющимся футером, юридическими текстами, комментариями, а также обрамлением дизайнерских и технических конструкций. Если система очистки ошибочно удалит нужный блок или если модель отвлечется на похожие числа в другом контексте (например, год в футере), результат будет неверным. Важно подчеркнуть, что здесь ошибка не является сугубо «ошибкой интеллекта» модели. Она вызвана тем, что сам источник не предоставляет детерминированного канала для передачи факта машине. Машина вынуждена реконструировать смысл из артефактов интерфейса.

Системные издержки: стоимость шума в токенах, времени и рисках

Переход к массовому машинному чтению делает стоимость шума измеримой. В человеческом вебе избыточность и декоративность могли восприниматься как неизбежные атрибуты дизайна. В машинном режиме избыточность становится прямой строкой затрат.

Во-первых, существует стоимость токенов. Коммерческие API и корпоративные инфраструктуры тарифицируются по объему обрабатываемого текста. Если вход содержит большое количество нерелевантных элементов, система оплачивает обработку мусора. Это приводит к инфляции стоимости получения одного корректного ответа. Более того, если ошибка вынуждает повторить запрос, стоимость растет диспропорционально.

Во-вторых, существует стоимость латентности. Рендеринг страниц, исполнение JavaScript, очистка HTML, нормализация текста, построение эмбеддингов и поиск в индексах занимают время. В системах поиска с генеративными ответами каждая лишняя сотня миллисекунд снижает конкурентоспособность. В агентных системах латентность умножается на количество шагов, поскольку агент выполняет цепочки

действий.

В-третьих, существует стоимость ошибки. В прикладных доменах ошибка может означать неверное управленческое решение, финансовую потерю или юридический риск. Поэтому корпоративные системы часто вынуждены включать человека в контур проверки, что снижает ценность автоматизации. Возникает скрытая стоимость ручной модерации и исправления последствий галлюцинаций.

В-четвертых, существует экологическая стоимость. Обработка больших объемов шума требует дополнительной энергии в дата-центрах. На уровне глобальных масштабов это превращается в заметный вклад в углеродный след индустрии ИИ. В условиях, когда множество систем по всему миру параллельно очищают одни и те же страницы, повторяя одну и ту же работу, возникает системная неэффективность. Это аргумент в пользу переноса очистки и структурирования ближе к источнику, где работа может быть выполнена один раз и использоваться многими потребителями.

Постановка проблемы и требования к новой архитектуре

Если признать, что машинные читатели стали массовым субъектом, то следствием становится необходимость пересмотра информационной архитектуры. Важен принцип сов-

местимости: нельзя требовать мгновенного отказа от интерфейсов для людей. Человеческий веб будет существовать, поскольку он нужен людям. Следовательно, требуется модель сосуществования, в которой один и тот же источник предоставляет два слоя: человекоориентированный и машиноориентированный.

С архитектурной точки зрения машиноориентированный слой должен обеспечивать детерминированность извлечения, низкую энтропию входного сигнала, устойчивые якоря для цитирования, явное связывание фактов с контекстом и верифицируемую целостность. Эти свойства важны потому, что они превращают чтение из вероятностной реконструкции интерфейса в воспроизводимую процедуру извлечения знания.

Эти требования формируют основу машиноцентричной архитектуры. В данной монографии они будут реализованы через концепцию энтропийно-управляемой информационной архитектуры, которая предлагает унифицированный формат контент-конверта и два взаимодополняющих пути внедрения: протокол на стороне издателя и конвейер на стороне потребителя.

Концептуальные основания: что значит «читать» для машины

В повседневной речи чтение ассоциируется с последовательным восприятием текста человеком. В контексте синтетических когнитивных систем чтение имеет более широкий смысл. Оно включает несколько операций, которые в человеческом чтении сливаются в единый акт понимания, но в машинном исполнении разделяются на этапы с различной чувствительностью к шуму.

Первый этап можно описать как извлечение наблюдаемого содержания. Система должна получить доступ к данным: скачать документ, обработать кодировку, разрешить редиректы, выполнить или не выполнить клиентские скрипты, преобразовать визуально-ориентированное представление в текстовое. Уже здесь возникает существенная потеря: элементы, которые были видимы человеку на экране, могут отсутствовать в исходном HTML; наоборот, в HTML могут присутствовать элементы, которые человек никогда не увидит, но которые будут считаны машиной.

Второй этап является структурированием. Система пытается разделить поток токенов на смысловые единицы: разделы, абзацы, таблицы, определения, инструкции. В идеале структура документа является явной. В антропоцентричном вебе структура часто имплицитна и выражена только визуально. Поэтому структурирование превращается в задачу ре-

конструкции, которая решается эвристиками или статистическими моделями и, следовательно, подвержена ошибкам.

Третий этап является извлечением утверждений. Модель должна определить, какие предложения выражают факты, какие содержат оценки, какие задают условия, какие описывают причинность, а какие являются декоративными или юридическими дисклеймерами. В человеческом чтении эти различия поддерживаются контекстом и пониманием жанра. Машинный читатель, особенно если он работает в режиме фрагментарного контекста, может смешивать жанры и подменять статус утверждений.

Четвертый этап является нормализацией и связыванием. Извлеченное должно быть приведено к единому представлению: даты, числа, единицы измерения, названия сущностей. Затем эти сущности должны быть связаны с другими источниками или с внутренними объектами системы. Ошибка на этом этапе часто имеет архитектурную природу: если документ не дает однозначных идентификаторов, система вынуждена угадывать, что именно имеется в виду.

Пятый этап является синтезом. Именно здесь формируется ответ или действие. Однако синтез не является автономным процессом; он зависит от того, что было извлечено и как было структурировано. Поэтому качество синтеза

нельзя оценивать, не оценивая качество входной среды чтения.

Данный разбор показывает, почему разговор о машинном чтении не сводится к вопросу «насколько умна модель». Модель может быть сильной, но если первые этапы построены на хрупких эвристиках, результат будет нестабилен. Следовательно, задача архитектуры заключается в том, чтобы переносить больше смысла из поздних статистических стадий в ранние детерминированные стадии, делая структуру и статусы утверждений явными.

Понятие информационной энтропии в контексте веба

Понятие энтропии имеет происхождение в физике и теории информации. В данной монографии оно используется как операциональный термин для описания степени неупорядоченности входного сигнала для машинного интерпретатора. Речь идет не о «сложности» содержания в интеллектуальном смысле, а о количестве неопределенности, которую система должна устранить, чтобы получить пригодный для вывода набор фактов и отношений.

Информационная энтропия веб-документа для машины складывается из нескольких компонент.

Первая компонента является структурной. Документ может быть формально размечен тегами, но не иметь явного соответствия между разметкой и логикой. В пределе весь документ может быть цепочкой однотипных контейнеров. Для машины это означает отсутствие опорных точек: где начинается раздел, где заканчивается определение, что является заголовком, что является подписью, что является элементом списка.

Вторая компонента является жанровой. В одном документе могут смешиваться рекламный текст, юридические условия, фактические сведения, элементы навигации, пользовательские отзывы, технические метаданные. Для человека жанры различимы по визуальному оформлению и привычным паттернам. Для машины жанры часто неразличимы, что приводит к смешению статусов утверждений. Например, дисклеймер «информация не является офертой» может быть воспринят как утверждение о свойствах продукта, если извлечен без контекста.

Третья компонента является повторяемостью. Повторяющиеся блоки увеличивают объем входа, не добавляя знания. При этом повторяемость создает ложные корреляции: статистические методы могут придавать повторяемому материалу чрезмерный вес. В RAG-системах повторяемость приводит к тому, что в индекс чаще попадают шаблонные фрагменты,

а уникальные знания оказываются вытесненными.

Четвертая компонента является динамической. Если содержание зависит от исполнения скриптов, персонализации, географии, состояния сессии, то документ перестает быть фиксированным объектом. Для машинного чтения это означает потерю воспроизводимости: разные запросы к одной и той же странице могут вернуть разные версии фактов. В корпоративной среде аналогом является отсутствие контроля версий или хранение нескольких редакций без явной связи.

Пятая компонента является шумом разметки и интерфейса. Сюда относятся CSS-классы, скрытые элементы, ARIA-атрибуты, фрагменты кода, технические идентификаторы компонентов, вставки аналитики. В текстовом представлении они становятся токенами, которые потребляют бюджет внимания.

Удобной эмпирической характеристикой является отношение сигнал/шум. Под сигналом здесь понимается совокупность текстовых фрагментов и структурных маркеров, которые необходимы для решения целевой задачи: ответить на вопрос, извлечь факт, выполнить действие. Под шумом понимается все, что увеличивает объем входа, но не повышает вероятность корректного решения. Важно подчеркнуть,

что граница между сигналом и шумом контекстна: то, что является шумом для задачи извлечения цены, может быть сигналом для задачи юридической проверки. Поэтому в рамках энтропийно-управляемой архитектуры предлагается описывать не универсальный набор «правильного» контента, а механизм контент-конверта, который может включать разные представления для разных задач, сохраняя при этом детерминированность и связность.

Эффект конфетти как структурная деградация знания

Эффект конфетти является одним из центральных отказов, возникающих при переводе человеческих документов в машиночитаемое представление. Он проявляется тогда, когда последовательный нарратив или логическая конструкция дробится на фрагменты по формальному критерию (например, фиксированное количество символов или токенов), не учитывающему границы смысла. В результате отдельные фрагменты сохраняют локальную связность, но теряют глобальную структуру: условия отделяются от следствий, определения отделяются от терминов, исключения отделяются от правил, ссылки отделяются от объектов, к которым они относятся.

В корпоративных системах эффект конфетти особенно заметен на примере регламентов и политик. Типичный

документ содержит общие принципы, определения терминов, область применения, исключения, последовательность действий и ответственность. Если такой документ разбит на фрагменты без семантических границ, система может извлечь правило без исключений или исключение без правила. В генеративном ответе это приводит к категоричности там, где документ требовал осторожности. Для бизнеса это означает риск принятия неверных решений.

В веб-контенте эффект конфетти проявляется, когда важные атрибуты сущности разбросаны по странице. Карточка товара может содержать цену, скидку, условия доставки и ограничения (например, «только для новых клиентов»). Эти элементы часто представлены в разных визуальных блоках. При конвертации в линейный текст без сохранения связей между блоками атрибуты теряют принадлежность к конкретной сущности. Модель может присвоить скидку не тому тарифу или применить условие доставки к другому варианту. В интерфейсе для человека ошибка маловероятна: визуальная группировка удерживает связи. В машинном чтении без явных связей ошибка становится статистически ожидаемой.

Эффект конфетти усиливается тремя факторами.

Первый фактор связан с компрессией формата. PDF

и веб-страницы часто содержат визуальную структуру, которая не имеет прямого соответствия текстовому порядку. Конвертеры восстанавливают порядок по эвристике, что уже вносит неопределенность.

Второй фактор связан с эмбеддингами. Векторные представления хорошо улавливают локальную семантическую близость, но плохо сохраняют глобальную композицию документа. Если фрагменты недостаточно содержательны, они становятся неразличимыми, а если слишком длинны, они содержат смешанные темы, что ухудшает поиск. В обоих случаях возникает системное расхождение между структурой знания и структурой индекса.

Третий фактор связан с шаблонностью. Когда в документе много повторяющихся формулировок, поиск по эмбеддингам чаще возвращает шаблонные куски, а не уникальные уточнения. Это приводит к тому, что контекст, предоставленный модели, статистически «похож», но фактически неполон.

С практической точки зрения эффект конфетти является аргументом в пользу того, что машинный слой должен содержать семантические сегменты, определенные источником, а не потребителем. Иными словами, лучше, если издатель сам определяет границы смысловых единиц, чем если

это делает каждая система извлечения по-своему. Это снижает вариативность и повышает воспроизводимость.

Почему существующие инструменты семантики не решают проблему

Может показаться, что задача уже решена стандартами структурированных данных и доступности. В действительности эти инструменты улучшают отдельные аспекты, но не устраняют архитектурного разрыва.

Структурированные данные, такие как Schema.org, ориентированы на описание сущностей и отдельных атрибутов. Они полезны, когда задача сводится к извлечению простых фактов. Однако они плохо описывают аргументацию, процедурные инструкции, причинно-следственные связи, исключения и контекстные ограничения. Например, политика возврата товара может зависеть от категории товара, юрисдикции и способа доставки. Описать это в виде плоского набора атрибутов трудно. В результате издатели либо не размечают сложные случаи, либо делают это упрощенно.

Механизмы доступности (например, ARIA) улучшают взаимодействие с интерфейсом, но не предназначены для передачи знания. ARIA помогает понять роль элемента (кнопка, меню, диалог), но редко фиксирует семантику содержи-

мого. Для автономных агентов это может быть полезным для навигации, но для извлечения фактов и аргументов этого недостаточно.

Метаданные вроде Open Graph и Twitter Cards ориентированы на презентацию в социальных сетях, а не на семантическую точность. Они дают заголовок, описание и изображение, но не гарантируют верифицируемость и структурированность.

Наконец, традиционные практики SEO ориентированы на ранжирование и клики, а не на качество машинного чтения. Многочисленные шаблонные тексты, созданные ради ключевых слов, ухудшают сигнал/шум. Парадокс заключается в том, что то, что было рационально в экономике кликов, становится иррациональным в экономике машинного чтения: чем больше искусственного текста, тем выше стоимость обработки и тем выше риск ошибок.

Экономика внимания и переход к экономике вычислений

Веб последних двух десятилетий развивался под влиянием экономики внимания. Сайт конкурировал за то, чтобы пользователь остался, посмотрел больше, перешел по ссылкам, увидел рекламу или совершил покупку. Архитектурные решения оптимизировались под эти цели. Однако ма-

шинный читатель не является носителем человеческого внимания. Он не «задерживается» на странице ради эмоций и не воспринимает дизайн как ценность. Его ресурсом является вычислительный бюджет: токены, время, энергия, ограничение по контексту, ограничения по безопасности.

Когда основным посредником между пользователем и контентом становится система генеративного поиска, происходит смещение экономического центра тяжести. Не сайт выбирает, что показать, а агент выбирает, что прочитать и как использовать. Для издателя это означает новый тип конкурентной борьбы: не только за человеческую конверсию, но и за машинную пригодность. Источник, который предоставляет ясные определения, структурированные факты и надежные ссылки, становится предпочтительным, потому что он снижает риск ошибок агента. Источник, который заставляет систему тратить ресурсы на очистку и реконструкцию, становится менее предпочтительным, даже если он визуально красив.

В этом контексте энтропийно-управляемая архитектура является не только техническим предложением, но и экономическим. Она предлагает переместить часть стоимости очистки с потребителя на источник, но делает это так, чтобы стоимость была одноразовой и разделяемой. Если источник публикует машиночитаемый слой, множество потребителей

могут использовать его без повторной очистки. В сумме это уменьшает затраты экосистемы.

Кейсы машинного чтения: как ошибка рождается из архитектуры

Для конкретизации рассмотрим несколько типичных случаев, которые демонстрируют, как архитектурные свойства источника производят ошибки чтения.

Кейс 1. Извлечение цены и условий тарифа.

Страница описывает несколько тарифных планов. Каждый план имеет цену, период оплаты, ограничения и набор функций. В человеческом интерфейсе планы представлены как карточки. В HTML карточки могут быть реализованы повторяющимся компонентом, внутри которого есть элементы, зависящие от состояния. Цена может рендериться только после загрузки данных. Дополнительно на странице могут присутствовать блоки «часто задаваемые вопросы», где упоминаются другие числа. Машинная система, которая получает линейный текст, может смешать числа и присвоить цену неверному плану или перепутать месячную и годовую стоимость. Если агент использует эту информацию для выбора тарифа, последствия будут прямыми.

В машиноориентированном слое цена должна быть пред-

ставлена как атрибут конкретной сущности «тарифный план», с указанием валюты, периода и условий. Кроме того, необходимо указать, применима ли скидка, и если да, то при каких условиях. Без такого слоя машина вынуждена реконструировать сущности из визуальных паттернов, что статистически ненадежно.

Кейс 2. Политика возврата и юридические ограничения.

Сайт публикует правила возврата. Текст содержит общие принципы, затем исключения: товары определенных категорий не подлежат возврату, возврат возможен только при сохранении упаковки, срок зависит от юрисдикции. В человеческом чтении исключения легко заметить. В RAG-системе документ разбивается на фрагменты. Если в выдачу попадает фрагмент с общим правилом без исключений, модель сформирует ответ, который будет неверным в юридическом смысле. Если система пытается компенсировать это увеличением контекста, в него попадает много дополнительного шума, а исключения могут оказаться не в тех фрагментах.

В машиноориентированном слое политика должна быть представлена как набор правил с условиями применимости, где исключения являются частью того же правила, а не отдельным абзацем. Это не обязательно требует сложной логики, но требует структурирования на уровне источника.

Кейс 3. Техническая документация и зависимость версий.

Инструкция описывает API. Параметры менялись между версиями, но документ объединяет несколько версий или содержит примечания о совместимости. При извлечении фрагментов система может смешать параметры из разных версий. В результате ответ будет частично правильным, но неприменимым. В корпоративной среде такая ошибка может привести к сбоям в интеграции и дополнительным затратам.

Машиноориентированный слой должен фиксировать версии как первичную координату. Любое утверждение о параметре должно быть привязано к версии. Без явной версии машина вынуждена угадывать по контексту.

Кейс 4. Новости и эффект временной неоднозначности.

Публикации часто обновляются, но в тексте сохраняются фразы «сегодня», «на этой неделе», «в прошлом месяце». Человек интерпретирует их относительно даты чтения или даты публикации. Машина может интерпретировать их неверно, особенно если она читает текст вне контекста публикации. В генеративном поиске это приводит к устаревшим ответам, если дата не является явной частью извлеченного фрагмента.

Машиноориентированный слой должен содержать явные временные метки и, при необходимости, нормализованные

интервалы времени. В противном случае факты теряют контекст истинности.

Эти кейсы показывают, что проблема не является частным дефектом конкретного инструмента. Она воспроизводится в разных доменах, потому что имеет архитектурную природу: источник знания не предоставляет структуру, необходимую для надежного машинного чтения.

Операционализация требований: инварианты машиноориентированного слоя

Ранее были перечислены требования к машиноориентированному слою. Для практической реализации полезно представить их как инварианты, то есть свойства, которые должны сохраняться независимо от визуального дизайна и технического стека сайта.

Инвариант 1. Сущности и атрибуты должны быть явно типизированы.

Если документ сообщает о товаре, тарифе, событии, персоне или организации, это должно быть выражено как сущность определенного типа. Атрибуты должны быть определены в явной форме. Это уменьшает число интерпретаций.

Инвариант 2. Границы смысловых сегментов должны

быть определены источником.

Разделы, определения, правила, процедуры должны иметь явные границы. Это устраняет эффект конфетти и делает сегментацию воспроизводимой.

Инвариант 3. Каждое утверждение должно иметь локальную опору для цитирования.

Опора может быть идентификатором сегмента, номером утверждения, стабильным хешем фрагмента. Важен принцип: потребитель должен иметь возможность указать, откуда именно взят факт, и другой потребитель должен воспроизвести это.

Инвариант 4. Связи между нарративом и фактами должны быть сохраняемы.

Если факт извлечен из абзаца, должно быть известно, из какого абзаца. Если правило имеет исключения, исключения должны быть связаны с правилом. Это обеспечивает проверяемость и снижает риск вырывания из контекста.

Инвариант 5. Представление должно быть устойчивым к изменениям интерфейса.

Редизайн сайта не должен ломать машиноориентированный слой. Следовательно, слой должен быть отделен от DOM и CSS и публиковаться как независимый артефакт, связанный с источником.

Инвариант 6. Слой должен поддерживать минимальную самодостаточность.

Потребитель должен понимать контекст без необходимости загружать весь интерфейс. Это означает наличие заголовков, дат, версий, единиц измерения и других базовых метаданных.

Эти инварианты задают критерии качества. В последующих главах они будут использованы для построения формата контент-конверта и для разработки метрик энтропии и пригодности источников к машинному чтению.

Сосуществование HCA и MCA: не конкуренция, а разделение функций

Смена парадигмы часто вызывает опасение, что машиноориентированный слой «обезличит» веб и вытеснит человеческий дизайн. В действительности предлагаемый подход основывается на разделении функций. Человекоориентированный слой сохраняет свою роль: он обеспечивает опыт, доверие, брендинг, эмоциональную составляющую, удобство навигации и взаимодействие. Машиноориентированный слой выполняет другую роль: он обеспечивает доставку знаний и фактов с минимальной энтропией.

Это разделение делает систему более честной с точки зрения эпистемологии. Человеческая страница часто смешивает факты и маркетинг. Для человека это допустимо, потому что он способен критически оценивать. Для машины смешение является источником ошибок. Публикация отдельного слоя фактов и структурированных правил не запрещает маркетинговый нарратив, но делает ясным, где находится слой истины, а где слой убеждения.

На уровне внедрения это означает, что организация может улучшать машинную пригодность без полного редизайна. Достаточно добавить второй канал публикации, связанный с теми же страницами. Это снижает барьеры принятия и делает переход эволюционным.

Границы применимости и риски машиноориентированного подхода

Любая архитектурная стратегия имеет ограничения. Важно обозначить их заранее, чтобы избежать утопических ожиданий.

Первое ограничение связано с тем, что не весь контент может быть сведен к фактам. Многое носит интерпретационный характер: эссе, художественные тексты, мнения. Машиноориентированный слой не обязан превращать все в табли-

цы. Он должен обеспечивать ясные границы между фактами, интерпретациями и источниками, а также структурировать то, что структурируемо, не уничтожая жанровое разнообразие.

Второе ограничение связано с доверенной моделью. Если издатель публикует машиноориентированный слой, возникает вопрос: насколько этот слой соответствует человеческой странице и не является ли он манипуляцией. Поэтому критически важны механизмы связывания и проверяемости: возможность сопоставить факты со ссылками на нарратив и использовать независимые методы проверки.

Третье ограничение связано с безопасностью и приватностью. Машиноориентированный слой может облегчить массовое извлечение информации, включая нежелательное. Поэтому в ряде доменов потребуется баланс между доступностью и контролем. В корпоративной среде машиноориентированный слой будет существовать внутри периметра безопасности.

Четвертое ограничение связано со стандартизацией. Если каждый издатель создаст свой формат, преимущества совместимости будут потеряны. Поэтому критически важна унификация контент-конверта и минимального набора инвариантов.

Эти ограничения не отменяют необходимость архитектурной эволюции. Они лишь указывают, что новая архитектура должна быть не только эффективной, но и проверяемой, безопасной и совместимой.

Метрики машинной пригодности: как измерять прогресс

Чтобы переводить дискуссию из уровня лозунгов в уровень инженерии, необходимо определить измеримые характеристики. В человеческом вебе метриками служили клики, глубина просмотра, конверсия, удовлетворенность. В эпоху машинных читателей появляются новые метрики, ориентированные на качество извлечения и воспроизводимость.

Первая группа метрик описывает чистоту сигнала. На операциональном уровне можно считать объем входа в токенах и долю токенов, относящихся к целевому содержанию. Эта доля зависит от задачи. Для ответа на вопрос о цене целевыми токенами будут числовые значения, валюта, период оплаты, условия применения скидки и указание, к какому тарифу относится цена. Для ответа на вопрос о политике возврата целевыми будут условия, сроки, исключения, юрисдикции. Смысл метрики состоит в том, чтобы зафиксировать, насколько много нерелевантного материала требуется обработать, чтобы добраться до релевантного.

Вторая группа метрик описывает структурную полноту. В машинном чтении важна не только доля сигнала, но и то, представлены ли ключевые элементы структуры: определения терминов, область применимости, перечисление исключений, ссылки на источники. Структурная неполнота означает, что даже при чистом тексте система будет выдавать неполные ответы, потому что сама структура знания не представлена в явной форме.

Третья группа метрик описывает устойчивость. Если источник обновляется, важно понимать, насколько изменения ломают цитирование и привязку утверждений. Устойчивость можно измерять как долю сегментов, которые сохраняют свои идентификаторы при изменениях, и как долю утверждений, которые можно сопоставить между версиями. В человеческом чтении эта проблема часто скрыта, потому что человек видит текущую версию и не требует воспроизводимости. Машинные системы, напротив, строят индексы, графы знаний и цепочки рассуждений, которым требуется стабильность.

Четвертая группа метрик описывает проверяемость. Если система извлекает факт, должна существовать возможность проверить его происхождение. Это означает наличие якоря, ссылки на сегмент, а также возможность сопоставить факт

с текстовым фрагментом. Верифицируемость важна не только для доверия пользователя, но и для внутреннего контроля качества в корпоративных системах: наличие трассировки позволяет находить источники ошибок.

Пятая группа метрик описывает риск галлюцинаций и ошибочной ассоциации. Эти риски можно оценивать экспериментально: задавать системе вопросы, для которых ответ присутствует в источнике, и измерять частоту ошибок. Однако в рамках энтропийно-управляемой архитектуры важно связать риск с характеристиками входа. Иначе меры будут реактивными: мы будем фиксировать сбой, но не понимать, какие свойства источника его вызывают.

Метрики в данном списке не претендуют на окончательность. Их роль в этой главе состоит в том, чтобы подчеркнуть принцип: машинная пригодность является измеримой. Если мы можем измерять клики и конверсию, мы можем измерять и качество машинного чтения. Это превращает переход к новой архитектуре в управляемую инженерную задачу.

Сравнение с предыдущими попытками семантизации веба

История веба содержит несколько волн попыток сделать знания более машиночитаемыми. Наиболее известной яв-

ляется концепция семантического веба, ориентированная на формальные онтологии, RDF-графы и логический вывод. Несмотря на интеллектуальную привлекательность, эти подходы не стали повсеместным стандартом. Причины этого важны для понимания нынешней ситуации, поскольку они показывают, какие условия необходимы для успешной стандартизации.

Во-первых, семантический веб требовал высокой дисциплины моделирования и согласования онтологий. Для большинства издателей это было слишком дорого, а выгоды были отложенными и неочевидными. Во-вторых, инструменты публикации и потребления не были достаточно массовыми. В-третьих, ценность для пользователя оставалась косвенной: он не видел прямого улучшения опыта.

Текущая волна отличается тем, что выгода становится прямой и немедленной. Машинные читатели уже используются миллионами людей через интерфейсы генеративного поиска и ассистентов. Если источник предоставляет машиноориентированный слой, он повышает вероятность того, что его факты будут правильно включены в ответы, а его процедуры будут правильно воспроизведены агентами. Это создает мощный стимул. Кроме того, требования новой волны менее утопичны: речь идет не о полном формальном описании мира, а о практической упаковке знаний в форме, при-

годной для чтения и проверки.

Таким образом, предложенный подход можно рассматривать как прагматическое продолжение семантической линии: он сохраняет идею явной структуры, но концентрируется на снижении энтропии и на обеспечении трассируемости, а не на универсальной онтологии.

Социотехническое изменение: от «текста для людей» к «знанию для агентов»

Переход к машинному чтению меняет не только технические стандарты, но и практики производства текста. В человеческом вебе текст часто создавался как продукт маркетинга, редакторской политики или личного высказывания. В машинном режиме текст становится компонентом цепочки принятия решений. Это означает, что требования к точности, определенности и внутренней согласованности возрастают.

В корпоративной среде это приводит к переоценке роли документации. Документ перестает быть архивом для людей и превращается в интерфейс для автоматизации. Если политика написана двусмысленно, агент не сможет корректно ее применять. Следовательно, появляется потребность в «машинной редактуре»: проверке документов на структурную

полноту, на наличие определений, на явность условий и исключений, на актуальность версий.

В публичном вебе это меняет стимулы контент-маркетинга. Тексты, которые были оптимальны для поисковых алгоритмов прошлого поколения, могут оказаться неэффективными для систем генеративного ответа. Если текст наполнен повторениями и искусственными ключевыми словами, он увеличивает энтропию. Машинный читатель будет избегать таких источников, если у него есть альтернативы с более чистым сигналом.

Кроме того, возникает новый вопрос ответственности. Если агент принимает решение на основе текста, кто отвечает за ошибку: автор текста, издатель, разработчик агента или поставщик модели? Ответ зависит от того, насколько система обеспечивает трассируемость и проверяемость. Если факт имеет ясную ссылку на источник, распределение ответственности становится более определенным. Если же ответ является смесью фрагментов без ссылок, ответственность размазывается. Следовательно, архитектурные решения о якорях, версиях и подписи контента имеют правовое и этическое измерение.

Архитектурный принцип контент-конверта

Выше был введен мотив двухслойности: человекоориентированного и машиноориентированного канала. В качестве операционального механизма такой двухслойности в данной монографии используется понятие контент-конверта. Конверт следует понимать как упаковку знаний, которая отделяет смысловую полезную нагрузку от интерфейсной оболочки и фиксирует минимальный набор свойств, необходимых для надежного машинного чтения: структуру, идентификаторы, связи, метаданные и целостность.

Важное отличие контент-конверта от обычных структурированных данных состоит в том, что он не ограничивается перечислением атрибутов сущности. Он может включать нарративный слой, сегментированный по смысловым единицам, и связывать его с фактами. Это позволяет сохранять контекст, что особенно важно для сложных документов, где знания не редуцируются к таблице.

Контент-конверт также является компромиссом между универсальностью и практичностью. Он допускает расширение словарей под домены, но сохраняет минимальный каркас, общий для всех: идентификация источника, версии, сегментация, якоря, связи и проверка целостности. Благодаря этому конверт может стать «языком по умолчанию» для машинного чтения, не претендуя на онтологическую завершенность.

В этой главе контент-конверт представлен как принцип. Его спецификация и практические механизмы публикации будут рассмотрены далее. На данном этапе важно зафиксировать, что архитектурное решение не обязано быть радикальным. Оно может быть добавочным: слой конверта добавляется рядом с существующим вебом, создавая мост между человеческой и машинной субъектностью.

Глава 2. Проблема шума – единый анализ

Введение. Почему «шум» является центральной переменной

Переход к машинному чтению меняет ось оптимизации информационных систем. В классической веб-парадигме основными ограничителями выступали человеческое внимание, визуальная эргономика и способность текста убеждать и удерживать. В парадигме машинных читателей ограничения иные: вычислительный бюджет, длина контекста, надежность извлечения, воспроизводимость и проверяемость. При сохранении традиционной архитектуры источников эти ограничения проявляются как систематический феномен, который можно описать одной категорией: шум во входном сигнале.

В данной главе шум рассматривается не как метафора и не как частный дефект конкретного инструмента очистки, а как архитектурная переменная, определяющая предельную точность машинного чтения. Основная идея состоит в том, что шум распределен по всей цепочке доставки знаний: он присутствует в источнике (веб-страницы, документы, базы знаний), он возникает и усиливается у потребителя (скрей-

пинг, конвертация, разбиение на фрагменты, индексирование, сборка контекста), и он проявляется в итоговом поведении системы (ошибки, галлюцинации, неверные ассоциации, нарушения контекста истинности).

Единый анализ необходим по двум причинам. Во-первых, практика показывает, что попытки «лечить» шум локальными улучшениями дают ограниченный эффект, потому что шум является каскадным: небольшое загрязнение на раннем этапе перерастает в значимое искажение на позднем. Во-вторых, текущая отрасль склонна классифицировать проблему шума по доменам, что скрывает общую структуру: веб-оптимизация трактуется как задача SEO и доступности, оптимизация RAG (retrieval-augmented generation) трактуется как задача эмбеддингов и ретривала, а ошибки LLM трактуются как ограничение модели. В действительности это проявления одного и того же разрыва: контент, произведенный в человекоцентричной архитектуре, передается потребителю, который действует как машина и нуждается в машиноцентричной форме.

В последующих разделах вводится определение шума для информационных систем, предлагается различение синтаксического и семантического шума, описывается шум в веб-источниках и шум в конвейерах потребления (retrieval-augmented generation, RAG, и родственные схемы), форма-

лизуются коэффициент релевантности и налог на внимание, а также анализируются скрытые издержки и теоретический пробел в текущих подходах.

Определение шума в информационных системах

Слово «шум» в инженерии имеет исторически строгий смысл, связанный с теорией информации. В исходной постановке канал связи передает сигнал, а шум – это случайная компонента, уменьшающая способность приемника восстановить исходное сообщение. Однако прямое перенесение этой модели на цифровой контент приводит к двусмысленностям, потому что в современных системах сообщение, задача и приемник не фиксированы. Один и тот же документ может быть «сообщением» для множества задач, а «приемник» может быть как человеком, так и системой извлечения, так и агентом, выполняющим действия.

В рамках энтропийно-управляемой информационной архитектуры под шумом понимается любая составляющая входных данных, которая увеличивает объем обработки, но не повышает вероятность корректного решения целевой задачи в заданных ограничениях. Это определение специально операционально: оно связывает шум не с эстетикой текста и не с субъективной «лишностью», а с затратами и точностью в конкретной задаче.

Такое определение требует зафиксировать три компонента.

Первое: целевая задача. Задача может быть извлечением факта (цена, дата, юридическое условие), реконструкцией процедуры (последовательность действий), построением аргумента (почему выполняется правило), или планированием действия (какие шаги предпринять). Разные задачи предъявляют разные требования к структуре и полноте.

Второе: ограничения. В машинных системах ограничения выражаются через длину контекста, лимит токенов, время ответа, стоимость вычислений, а также через требования безопасности и приватности. При тех же данных, но при более жестких ограничениях, «шумность» источника возрастает, потому что полезная часть может не поместиться в доступный бюджет.

Третье: вероятность корректного решения. Она не обязана быть бинарной. Для некоторых задач допустима приблизительность, для других требуется юридическая точность и цитируемость. Чем выше требование к точности, тем более разрушителен шум, потому что даже небольшие пропуски условий и исключений приводят к неверным выводам.

Из этого следует важный вывод: шум является контекстно-зависимой величиной. Нельзя составить универсальный список «лишних» элементов, применимый к любому источнику. Можно, однако, выделить устойчивые классы шумов и описать, какие механизмы делают их системно опасными для машинного чтения.

Синтаксический и семантический шум

Для практики полезно различать два уровня, на которых возникает шум: синтаксический и семантический.

Синтаксический шум – это загрязнение, связанное с формой представления и с технологическими оболочками. Он включает артефакты разметки, фрагменты кода, повторяющиеся элементы интерфейса, технологические вставки, колонтитулы документов, а также ошибочный порядок текста при конвертации. Этот шум часто видим и кажется «легко устранимым», поскольку его можно удалять правилами.

Семантический шум – это загрязнение на уровне смысла. Он проявляется как избыточность, шаблонность, повторение одних и тех же утверждений разными словами, смешение жанров (маркетинг вместе с фактами), многозначность без явных ограничений, а также неявные связи, понятные человеку по визуальному расположению, но не выраженные

в структуре. Семантический шум труднее устраняется, потому что он встроен в редакторскую и коммерческую логику публикации.

На практике синтаксический и семантический шум тесно переплетены. Веб-страница может содержать мало технических артефактов, но быть семантически шумной из-за искусственного расширения текста. И наоборот: документ может быть семантически строгим, но синтаксически плохо читаемым из-за PDF, сканов или сложной верстки. Поэтому оптимизация машинного чтения не сводится к «очистке HTML». Требуется архитектурный слой, отделяющий смысловую полезную нагрузку от оболочки и фиксирующий структуру.

Контекстно-зависимость и граница между сигналом и шумом

Граница между сигналом и шумом определяется задачей, но существует несколько типичных источников ошибок, связанных с неверной классификацией.

Первый источник – смешение уровней: система принимает синтаксические маркеры за семантику или наоборот. Например, визуальный блок «рекомендуемые товары» может содержать числа и сроки, похожие на условия тарифа. Для человека контекст очевиден: он видит заголовок блока и рас-

положение. Для машины, получившей линейный текст, это может быть неотличимо от основного предложения.

Второй источник – отсутствие явных ограничений применимости. Юридические тексты почти всегда содержат условия и исключения. Если их границы не выражены структурно, система может извлечь правило как универсальное, хотя в документе оно было ограничено конкретной категорией товаров или юрисдикцией.

Третий источник – множественность версий и временная неоднозначность. В новостях, документации и тарифах часто присутствуют фразы, смысл которых зависит от времени. Человек интерпретирует их относительно даты публикации. Машина, читающая фрагменты вне исходного контекста, теряет координаты истинности.

Контекстно-зависимость не означает, что задача неформализуема. Напротив, она указывает на необходимость метрик, которые связывают шум с затратами и ошибками, а также на необходимость форматов, способных сохранять контекст: даты, версии, область применимости, связи между сущностями и утверждениями.

Шум в источнике: веб-контент

Веб является наиболее распространенным источником знаний для публичных систем. Его архитектура эволюционировала вокруг интерфейса и внимания, что делает его естественно «шумным» для машинного чтения. В данном разделе выделяются три доминирующих класса шумов веб-источника: структурный, семантический и технический. Они различаются механизмами возникновения и методами снижения.

Структурный шум: интерфейс как примесь в знании

Структурный шум возникает из-за того, что человеческий интерфейс обслуживает навигацию, доверие, конверсию и бренд, а не передачу знания машине. Типичный шаблон страницы содержит заголовок, меню, боковые панели, хлебные крошки, баннеры, блоки рекомендаций, форму подписки, футер, юридические ссылки, блоки «похожие материалы», элементы персонализации. Для человека большинство из этих элементов быстро распознаются как вспомогательные. Для машины они становятся частью текста, если не существует отдельного машиноориентированного представления.

Структурный шум опасен не только как «лишние токены». Он также нарушает семантические связи. На странице может быть несколько сущностей одного типа: несколько та-

рифов, несколько моделей товара, несколько дат. В интерфейсе эти сущности разделены визуальными контейнерами. В линейном тексте контейнер исчезает, и факты теряют принадлежность. Это порождает ошибочную ассоциацию: цена одного тарифа приписывается другому, срок доставки одного региона переносится на другой, ограничение из блока FAQ становится универсальным.

Отдельная форма структурного шума – динамическая структура. Современные сайты часто рендерят контент на клиенте, загружают данные асинхронно, скрывают часть текста за вкладками и раскрывающимися блоками. Для машины это означает вариативность источника: два чтения одной страницы могут дать разные тексты в зависимости от того, как выполнен рендеринг и какие события имитировались. В условиях требования воспроизводимости это является фундаментальным дефектом: источник не предоставляет детерминированного канала доставки знания.

Семантический шум: шаблоны, избыточность и смешение жанров

Семантический шум веба исторически связан с экономикой поиска и рекламы. В течение долгого времени оптимальная стратегия для многих издателей заключалась в увеличении объема текста, насыщении ключевыми словами и по-

вторении формулировок, чтобы повысить вероятность совпадения с запросами. В результате возникла массовая практика текста как заполнителя. Для человека такие тексты часто терпимы: он быстро сканирует и извлекает смысл. Для машинной системы, ограниченной контекстом и бюджетом, избыточность превращается в прямой налог.

Семантический шум проявляется в нескольких устойчивых формах.

Повторение без добавления содержания. Одна и та же мысль формулируется в нескольких абзацах разными словами. В человеческой риторике это может усиливать убеждение. Для машины это увеличивает риск того, что извлеченные фрагменты будут различаться стилистически, но не добавят необходимых условий и чисел.

Шаблонные вставки. «Мы заботимся о клиентах», «лучшее качество», «уникальный сервис». Такие фразы не несут проверяемых фактов и в большинстве задач являются чистым шумом. Однако они часто семантически близки к реальным утверждениям, что загрязняет поиск по эмбеддингам: модель возвращает куски с шаблонами, потому что они встречаются на многих страницах.

Смешение фактов и убеждения. В маркетинговом тексте

факты переплетаются с оценочными суждениями и обещаниями. Если структура не разделяет эти уровни, система извлечения может принять оценочное утверждение за факт или не различить модальность. Это особенно опасно в юридических и медицинских доменах, где модальность и оговорки являются частью истинности.

Неявные связи. Многие страницы используют перечисления, визуальные таблицы, карточки и графику. Смысл определяется расположением, а не грамматикой. В линейной форме такие структуры теряют границы, и семантика становится вероятностной.

Технический шум: артефакты формата и процесса доставки

Технический шум включает то, что возникает на уровне формата и механизма доставки. Это может быть текст из скриптов и стилей, обрывки JSON, шаблоны, скрытые элементы, повторяющиеся атрибуты, ошибки кодировок, а также «мусор», появляющийся при конвертации HTML в текст. Даже при аккуратной очистке остается проблема порядка: HTML допускает расположение элементов, не совпадающее с логическим порядком чтения, а значит восстановление последовательности является эвристическим.

Особая форма технического шума – фрагментация источника по сетевым и прикладным причинам. Данные могут быть распределены по нескольким запросам, часть контента загружается после пользовательского действия, часть доступна только после авторизации. Машинный потребитель, не имеющий полномочий или не исполняющий интерфейсные сценарии, получает неполный источник, который затем обрабатывается как если бы он был полным. Это создает систематическую ошибку: отсутствие данных интерпретируется как отсутствие факта.

Кейс-стади: анатомия «шумной» веб-страницы

Рассмотрим типовую страницу с тарифами SaaS-сервиса. Для человека страница выглядит как ясная таблица карточек: названия тарифов, цены, периоды оплаты, список функций, ограничения, кнопки покупки, а ниже – ответы на вопросы и юридические условия.

Для машины, читающей линейный текст, та же страница часто представляет собой смесь нескольких слоев.

Первый слой – навигация и бренд: логотип, пункты меню, призывы к регистрации, ссылки на блог, «о компании», «карьера».

Второй слой – основной контент: несколько тарифов, каждый из которых имеет цену и условия.

Третий слой – вторичный контент: отзывы, кейсы клиентов, сравнительные таблицы, блок «похожие продукты», предложения партнеров.

Четвертый слой – юридические оговорки и условия: политика возврата, ограничения по региону, налоговые условия, ссылки на договор.

Пятый слой – динамические элементы: переключатель «месяц/год», раскрывающиеся списки, подсказки, которые видимы только при наведении.

Ошибки возникают в момент, когда система извлечения пытается реконструировать структуру тарифов без явных контейнеров. Цена может встречаться рядом с упоминанием скидки, сроков и другого тарифа. Если часть страницы подгружается асинхронно, некоторые цены могут отсутствовать в момент чтения. Если в блоке FAQ присутствуют числа, похожие на цены, они попадают в контекст и конкурируют за внимание модели.

Даже если итоговый ответ будет «похож на правильный», он может быть неверным по деталям. На практике имен-

но детали – валюта, период, применимость скидки, ограничения – определяют правильность решения. Следовательно, шум следует рассматривать как источник не только затрат, но и систематического риска.

Шум у потребителя: RAG-конвейеры и смежные схемы потребления

Если веб-источник является шумным, можно ожидать, что потребитель попытается компенсировать это обработкой: очисткой, сегментацией, индексированием, извлечением наиболее релевантных фрагментов и их подачей модели. Современная индустрия описывает этот подход как retrieval-augmented generation (RAG). На практике существует множество вариаций, но почти все они разделяют общий принцип: данные сначала переводятся в линейный текст и разбиваются на фрагменты, затем фрагменты индексируются, затем по запросу извлекается подмножество и передается генеративной модели.

Проблема в том, что RAG не устраняет шум, а перераспределяет его. Часть шума удаляется, но часть превращается в новые ошибки: нарушение границ смысловых единиц, смешение контекстов, загрязнение шаблонами, дрейф индекса при обновлениях, утрата ссылок на исходные утверждения. В результате потребитель часто не получает «чисто-

го знания», а получает композит, в котором шум присутствует в измененной форме.

Артефакты чанкинга и ошибки границ

Разбиение на фрагменты является центральной операцией. Наиболее распространенная практика – фиксированный размер фрагмента по символам или токенам, иногда с перекрытием. Эта практика удобна инженерно, но плоха эпистемологически: она не уважает границы смысла. Определение термина может оказаться в одном фрагменте, а сам термин – в другом. Условие может быть отделено от следствия. Правило может быть отделено от исключения.

Следствие заключается в том, что даже при идеальном поиске по эмбеддингам извлеченный фрагмент может быть неполон. Модель, получившая неполный фрагмент, будет вынуждена восстановить недостающие связи. Если модель восстановит их неверно, ошибка будет выглядеть как «галлюцинация», хотя ее первопричина – потеря структуры при сегментации. Эта логика является прямым проявлением эффекта конфетти: знание дробится так, что его композиция перестает быть восстанавливаемой без домыслов.

Существуют попытки семантического чанкинга, где фрагменты строятся по заголовкам, абзацам или распознанным

темам. Однако без участия источника эти методы остаются эвристическими. Они не гарантируют воспроизводимости и часто ломаются на документах с нестандартной структурой, таблицами, вложенными условиями и сложными ссылками.

Загрязнение шаблонами и «смещение к среднему»

В корпоративных документах и в веб-страницах распространены повторяющиеся блоки: дисклеймеры, юридические тексты, стандартные вступления, колонтитулы, уведомления о конфиденциальности, шаблонные описания. При индексировании эти блоки создают эффект «смещения к среднему»: они встречаются в большом числе документов, а значит формируют плотный кластер в семантическом пространстве. Векторный поиск склонен возвращать такие фрагменты, потому что они статистически похожи на многие запросы.

Это приводит к парадоксальному результату: система извлекает контекст, который выглядит релевантным по сходству, но беден по фактам. Затем модель заполняет пробелы генерацией. На уровне метрики «похожести» все выглядит корректно, но на уровне правильности ответа система деградирует.

Проблема усиливается, если в корпусе доминируют шаблоны, а уникальные уточнения редки. Тогда вероятность того, что уникальное условие попадет в контекст, падает. В юридических и регуляторных задачах это означает системный риск: система отвечает общими правилами, пропуская исключения.

Семантическое смешение и загрязнение контекста

Даже если чанкинг выбран удачно, остаются две формы смешения.

Первая форма – смешение сущностей. Если запрос касается конкретного продукта или версии, а контекст включает фрагменты о похожих продуктах и версиях, модель может слить их. Для человека различие очевидно, потому что он держит в голове цель чтения. Для модели различие должно быть выражено структурой и идентификаторами; иначе оно становится вероятностным.

Вторая форма – смешение модальностей и режимов текста. В одном корпусе могут находиться требования, рекомендации, примеры, обсуждения. Если эти режимы не маркированы, модель может выдать рекомендацию как обязательное требование или пример как правило.

В обеих формах смешение не является «недостатком интеллекта». Это следствие того, что контекст, предоставляемый модели, не является детерминированной выборкой структурированных фактов, а является результатом эвристического поиска по шумному и частично разрушенному представлению.

Кейс-стади: корпоративный корпус документов и пределы «очистки»

Корпоративная среда часто воспринимается как «более контролируемая», чем веб. Однако именно здесь шум приобретает специфическую устойчивость. Документы создаются для людей и процессов, они содержат большое количество процедурных вставок, шаблонов согласования, ссылок на внешние регламенты, комментариев, следов редактирования и фрагментов переписки. Форматы вроде PDF фиксируют визуальную структуру, но скрывают семантическую. Даже хорошо организованный корпус может быть эпистемологически шумным: условия и исключения распределены по приложениям, версии документа указаны в колонтитулах, а актуальность определяется ссылками на приказы.

Типовой сценарий внедрения RAG в такой среде выглядит так: документы собираются, конвертируются в текст, очищаются от колонтитулов, разбиваются на фрагменты, ин-

дексируются. Затем пользователи задают вопросы, а система извлекает фрагменты и генерирует ответы.

Проблемы проявляются в нескольких местах. Во-первых, конвертация разрушает порядок и структуру, особенно в таблицах и в документах с многоколоночной верстkой. Во-вторых, удаление колонтитулов и шаблонов редко бывает идеальным и часто удаляет полезные сигналы, например номер версии или дату. В-третьих, фрагментация отделяет условия от исключений. В-четвертых, при обновлении документов индекс устаревает, а идентичные фрагменты в разных версиях становятся трудно различимы.

Результатом является то, что система может дать «убедительный» ответ, который противоречит актуальной версии политики. Для бизнеса такой ответ не просто неточность; это потенциальное нарушение регуляторных требований, финансовый риск и риск репутации. Таким образом, шум в корпоративной среде является экономически значимым.

Проблема коэффициента релевантности

Чтобы перевести разговор о шуме из уровня интуиции в уровень инженерии, требуется метрика. Одной из наиболее полезных является коэффициент релевантности R, который

выражает долю действительно полезного содержания в том объеме, который система вынуждена обработать, чтобы решить задачу.

В простейшей формулировке коэффициент релевантности задается как отношение релевантных токенов к общему числу токенов, предоставленных модели или извлеченных из источника:

$$R = T_{relevant} / T_{total}$$

Здесь $T_{relevant}$ – количество токенов, которые непосредственно участвуют в ответе. Это не только «токены правильного факта». В процедурных задачах это токены, задающие условия, исключения и последовательность действий. В юридических задачах это токены, определяющие применимость и ограничения. В инженерных задачах это токены версий, параметров и контрактов.

Удобно также определить шум D как долю нерелевантных токенов:

$$D = 1 – R$$

Эти определения позволяют сравнивать источники и конвейеры. С практической точки зрения важно не абсолютное

значение, а порядок величины. Если R близок к нулю, система обрабатывает почти чистый шум и вынуждена «догадываться». Если R высок, система имеет шанс быть надежной и воспроизводимой.

Эмпирические наблюдения показывают, что в сыром HTML коэффициент релевантности часто оказывается порядка одного процента. В такой ситуации до 99 процентов обработки приходится на отходы. Даже после извлечения очищенного текста типичный диапазон может составлять 3—6 процентов. В стандартных RAG-конвейерах при аккуратном корпусе и хороших запросах коэффициент может быть выше, но он редко приближается к единице из-за неизбежных примесей и смешения контекстов. На уровне архитектуры это означает следующее: без отдельного машиноориентированного слоя контент остается статистически неблагоприятным для надежного извлечения.

Налог на внимание как прямая функция шума

Коэффициент релевантности описывает долю полезного содержания. Для экономики вычислений полезно выразить, сколько дополнительной обработки требуется из-за шума. Эту величину можно назвать налогом на внимание и определить как отношение общего объема обработки к объему полезного сигнала:

$\tau = T_{total} / T_{relevant}$

С учетом определения D получается:

$\tau = 1 / (1 - D)$

Эта формула проста, но ее смысл важен. При умеренном шуме D = 0.5 налог равен 2: система должна обработать вдвое больше, чем нужно по содержанию. При шуме D = 0.7 налог становится больше трех: на каждый полезный токен приходится более двух токенов отходов. При шуме D = 0.9 налог равен 10, что делает масштабирование практически невыгодным.

Налог на внимание имеет два измерения: вычислительное и когнитивное в машинном смысле. Вычислительное – это токены и время. Когнитивное – это способность модели выделять релевантное среди нерелевантного. Даже если вычисления доступны, внимание модели не бесконечно: нерелевантный контекст отвлекает, уменьшает точность и увеличивает вероятность неверных ассоциаций.

Скрытые издержки шума

Понятие шума полезно тем, что оно связывает качество

источника с экономикой. Скрытые издержки шума обычно недооцениваются, потому что они распределены по инфраструктуре и проявляются как «нормальные накладные расходы». В действительности они имеют кумулятивный характер и при масштабировании становятся доминирующими.

Потери токенов и вычислительные накладные расходы. Любая система, которая читает шумный источник, тратит вычисления на очистку, сегментацию, извлечение и обработку. При массовом использовании это превращается в значимые расходы. Более того, шум заставляет увеличивать размер контекста и число извлеченных фрагментов, что экспоненциально увеличивает стоимость для моделей с дорогим контекстом.

Рост латентности. Чем больше шум, тем больше операций требуется, тем сложнее ретривал, тем больше вероятность повторных запросов и уточнений. Для систем, ориентированных на интерактивность, латентность является конкурентным параметром. Шум снижает конкурентоспособность источников, даже если они качественны для человека.

Снижение точности и рост стоимости ошибок. Ошибка машинного чтения имеет стоимость, которая зависит от домена. В потребительском сценарии это может быть неудобство. В корпоративном – финансовые потери и регуляторные

риски. В медицине и праве – риск вреда. Важно подчеркнуть, что стоимость ошибки растет быстрее, чем стоимость вычислений: один неверный юридический ответ может стоить больше, чем тысячи запросов. Следовательно, шум следует рассматривать как фактор риска, а не только как фактор затрат.

Усиление галлюцинаций как вторичный эффект. Когда релевантность низка, модель вынуждена заполнять пробелы. Это не обязательно «фантазия» в бытовом смысле; это статистическое восстановление недостающей структуры. Чем больше пробелов, тем больше вероятность, что восстановление не совпадет с источником. Таким образом, галлюцинация часто является следствием шума, а не автономным дефектом модели.

Экологический и инфраструктурный след. В масштабах индустрии избыточная обработка шумного контента означает избыточное потребление энергии и ресурсов дата-центров. Пока индустрия рассматривает это как неизбежную цену прогресса. Однако архитектурное снижение шума меняет картину: если источник публикует машиноориентированное представление, множество потребителей перестают повторять одну и ту же очистку. Это уменьшает совокупные издержки экосистемы.

Каскадный эффект: как шум усиливается в процессе обработки

Шум опасен тем, что он редко остается локальным. Он проходит через конвейер и трансформируется в новые виды шумов. Этот каскад можно описать как последовательность стадий.

Стадия извлечения. Источник читается через скрейпинг, конвертацию или API. На этом этапе возникают ошибки доступа, неполнота данных, неверный порядок текста, потеря скрытых элементов. Часть шума появляется как отсутствие: важные фрагменты не извлечены.

Стадия очистки. Система удаляет технические и структурные элементы. Здесь шум превращается в риск удаления сигнала: фильтр, удаляющий «лишние» блоки, может удалить юридическое исключение, потому что оно оформлено как мелкий текст. Чем агрессивнее очистка, тем выше риск.

Стадия сегментации. Текст разбивается на фрагменты. Здесь структурные связи разрушаются. Даже если все слова сохранены, композиция знания теряется. Это порождает эффект конфетти, который затем интерпретируется как неопределенность.

Стадия индексирования. Фрагменты переводятся в эмбеддинги и помещаются в индекс. Здесь шум проявляется как смещение: шаблонные фрагменты становятся более доступными, чем уникальные. Кроме того, векторное пространство не хранит явную логику применимости и модальности; оно хранит близость. Близость не равна истинности.

Стадия ретривала. По запросу извлекаются фрагменты. Здесь шум проявляется как смешение: в контекст попадают куски из разных сущностей, версий или режимов текста.

Стадия генерации. Модель получает контекст и формирует ответ. Здесь шум превращается в отвлечение внимания, неверные ассоциации и заполнение пробелов. На этом этапе ошибка выглядит как «ошибка модели», хотя она произведена каскадом предыдущих стадий.

Важно видеть, что улучшение одного этапа редко решает проблему целиком. Например, улучшение эмбеддингов может повысить качество ретривала, но не восстановит разрушенные границы смысла. Улучшение чанкинга может уменьшить эффект конфетти, но не решит неполноту данных и отсутствие версий. Следовательно, требуется архитектурная стратегия, которая снижает шум у источника и обеспечивает детерминированность и структуру до начала конвейера.

Теоретический пробел в текущих подходах

Современные практики оптимизации информационных систем накопили множество частных методов. Однако в контексте машинного чтения проявляется пробел: отсутствует единый уровень, который связывает редакторскую практику, формат публикации и требования машинного потребителя.

SEO и структурированные данные улучшают доступность и частично повышают извлекаемость фактов. Но эти инструменты ориентированы на отдельные атрибуты и не решают проблему процедур, аргументов, исключений и контекста истинности. Более того, они не гарантируют, что модель будет использовать разметку правильно, если остальной контекст шумен.

Оптимизация эмбеддингов, ретривала и ранжирования является «вниз по течению» решением. Она предполагает, что источник неизменен и что задача сводится к поиску релевантных фрагментов. Но если фрагменты уже разрушены сегментацией, если версии смешаны, если исключения отделены от правил, то улучшение поиска лишь ускоряет извлечение неполного знания.

Попытки решать проблему на уровне модели также огра-

ничены. Увеличение контекста позволяет вместить больше текста, но оно не увеличивает долю релевантного. При низком коэффициенте релевантности увеличение контекста часто означает увеличение шума. Более «умная» модель может лучше фильтровать, но при высоком шуме фильтрация превращается в угадывание, а не в детерминированное извлечение.

Недостающее звено можно обозначить как архитектура контента. Требуется слой, который делает знание представимым в форме, пригодной для машинного чтения: определяет семантические сегменты, фиксирует идентификаторы, связывает факты с источниками, задает версии и временные метки, отделяет нарратив от оболочки. В рамках данной монографии этот слой будет формализован через концепцию контент-конверта и через различение человекоцентричной и машиноцентричной архитектур.

Резюме главы

В главе был введен единый взгляд на проблему шума. Шум определен как контекстно-зависимая составляющая входа, увеличивающая затраты без повышения вероятности корректного решения задачи. Показано различие синтаксического и семантического шума и описаны их проявления в веб-источниках и в конвейерах потребления. Введены ко-

эффициент релевантности и налог на внимание как метрики, связывающие качество источника с экономикой вычислений и с вероятностью ошибок. Показано, что шум имеет каскадный характер: он усиливается на стадиях извлечения, очистки, сегментации, индексирования, ретривала и генерации. Наконец, обозначен теоретический пробел текущих подходов: при всей полезности SEO, структурированных данных и улучшений RAG отсутствует архитектурный слой, который бы обеспечивал детерминированное и проверяемое машинное чтение.

Следующая глава вводит таксономию парадигм информационной архитектуры, где человекоцентричная и машиноцентричная формы описываются как разные режимы организации контента. Эта таксономия необходима, чтобы перейти от описания проблемы шума к формальному проектированию решений и к измерению враждебности источников к машинному чтению.

Глава 3. Таксономия парадигм информационной архитектуры

Введение. Зачем нужна таксономия

Предыдущая глава рассматривала шум как центральную переменную машинного чтения и показала его каскадную природу. Однако само понятие шума остается неполным, если оно не встроено в более широкую типологию архитектурных режимов, в которых производится и потребляется цифровой контент. На практическом уровне индустрия часто описывает различие источников через формат (HTML, PDF, API) или через инструмент (скрейпер, парсер, RAG-конвейер). Эти классификации полезны, но не раскрывают фундаментального различия, определяющего воспроизводимость, проверяемость и стоимость извлечения смысла.

В данной главе вводится таксономия парадигм информационной архитектуры, основанная на дихотомии человекоцентричной архитектуры и машиноцентричной архитектуры. Эта таксономия служит двум целям. Первая цель – дать формальный язык для описания того, почему один и тот же контент может быть одновременно удобен для человека и враждебен для машины. Вторая цель – показать путь проектирования, при котором человеческий интерфейс

и машинное представление не конкурируют, а сосуществуют в двухуровневой модели, позволяющей постепенный переход.

Принципиально важно подчеркнуть, что речь не идет о замене одного режима другим. Человекоцентричный веб является социальной и экономической реальностью, и он не исчезнет. Таксономия нужна не для того, чтобы объявить существующие практики ошибочными, а для того, чтобы отделить функции представления от функций передачи знания машине и тем самым уменьшить энтропию канала, по которому машина получает смысл.

Дихотомия HCA/MCA: формальное определение

Под человекоцентричной архитектурой (HCA) понимается такой режим организации цифрового контента, при котором первичным адресатом является человек, а смысл и навигация реализуются главным образом через слой представления. В HCA структура документа оптимизирована под восприятие, внимание и интерактивность. Смысл часто выражается не только через текст, но и через визуальные контейнеры, позиционирование, типографику, микровзаимодействия, композицию и культурные ожидания читателя.

Под машиноцентричной архитектурой (MCA) понимает-

ся режим, при котором первичным адресатом является машинный потребитель, а смысл передается через явные семантические структуры, пригодные для детерминированного извлечения. В MCA структура задается так, чтобы основные утверждения, сущности, отношения, условия, исключения, версии и источники были доступны без реконструкции по визуальным артефактам.

Формальные различия между HCA и MCA можно описать через четыре оси.

Ось адресата. HCA оптимизирует когнитивный и поведенческий цикл человека: обнаружение, сканирование, интерпретация, доверие, действие. MCA оптимизирует вычислительный и верификационный цикл машины: обнаружение, загрузка, разбор, сопоставление, проверка целостности, извлечение, цитирование.

Ось семантики. HCA допускает неявную семантику и компенсирует ее человеческими способностями к контекстуализации. MCA требует явной семантики, поскольку для машины контекст не является устойчивым: он меняется при конвертации, чанкинге, ранжировании и сборке контекста.

Ось детерминированности. HCA допускает вариативность представления и динамические элементы, поскольку

человек интерпретирует результат как опыт. MCA стремится к детерминированности: один и тот же запрос к источнику должен воспроизводимо возвращать одну и ту же полезную нагрузку с фиксированными идентификаторами и версиями.

Ось соотношения сигнал/шум. HCA часто производит высокий шум в машинном смысле, потому что значимая информация окружена интерфейсной и маркетинговой оболочкой. MCA стремится к высокой доле сигнала и к минимизации примесей, которые не увеличивают вероятность решения машинной задачи.

Эти оси не образуют бинарного выбора. В реальности существует спектр, где источники занимают промежуточные позиции. Например, техническая документация может быть ближе к MCA по формальности терминов, но оставаться HCA по структуре и по доминированию интерфейса. И наоборот, внутренний API может быть MCA по данным, но HCA по описанию условий в виде нарратива без явных контрактов.

Важным следствием спектральной природы является возможность измерения: если существует шкала, на которой можно описать степень человекоцентричности и машиноцентричности, то можно оценивать прогресс, сравнивать подходы и управлять переходом без разрушения пользова-

тельского опыта.

Человекоцентричная архитектура (HCA) в деталях

HCA исторически возникла как эволюция печатной культуры в цифровую среду. Печатная страница уже была ориентирована на человека: она использовала композицию, иерархию заголовков, колонтитулы, сноски и визуальные маркеры. Цифровая среда добавила интерактивность, гиперссылки, персонализацию и поведенческую оптимизацию. В результате современные веб-страницы стали не столько документами, сколько интерфейсами, в которых текст является лишь одним из элементов.

Доминирование слоя представления

В HCA слой представления выполняет сразу несколько функций: навигацию, доверие, объяснение, убеждение и конверсию. Именно поэтому он доминирует над смысловым слоем. Для человеческого читателя это рационально: дизайн сокращает время ориентации, выделяет важное и поддерживает мотивацию. Для машины доминирование представления означает, что канал передачи знания не отделен от канала управления вниманием.

Доминирование представления проявляется в том, что

многие смысловые связи выражаются через контейнеры и расположение, а не через формальные отношения. Таблица тарифов на странице часто является визуальной таблицей, но в исходном HTML может быть реализована как набор вложенных блоков, порядок которых не гарантирует логическое чтение. Визуальные подсказки, такие как подчеркивание, выделение цветом, иконки, становятся частью смысла для человека, но не обязательно переводятся в машинно-распознаваемые атрибуты.

Неявная семантика

Неявность семантики является ключевым свойством HCA. Человек способен восстановить смысл из неполной или неоднозначной формы, используя общий фон знаний, культурные конвенции и восприятие контекста. Машина в типовом конвейере получает линейный текст, в котором контекст выражен слабо. Даже если используются современные модели, их способность компенсировать неявность является статистической и зависит от корпуса и от формы запроса. Следовательно, неявная семантика HCA не является проблемой качества текста; она является несоответствием адресата.

Неявная семантика особенно заметна в случаях, когда важные ограничения представлены как мелкий текст, всплы-

вающая подсказка или блок внизу страницы. Человек видит структуру страницы и понимает, что это оговорка, относящаяся к конкретному элементу. Машина может увидеть только последовательность предложений и чисел без принадлежности. Это приводит к систематическим ошибкам, когда исключения отделяются от правил, а условия применимости теряются.

Высокий коэффициент шума

В HCA шум возникает не случайно, а как продукт оптимизации под внимание и экономику взаимодействия. Навигационные блоки, рекомендации, повторяющиеся элементы, юридические вставки, элементы доверия и маркетинговые слои создают значительный объем текста и метаданных, которые не участвуют в решении машинной задачи извлечения. На практике это приводит к низкому коэффициенту релевантности и к росту налога на внимание в машинном смысле.

Важная деталь состоит в том, что в HCA шум не является чисто внешней примесью. Он часто интегрирован в основное повествование. Маркетинговая фраза может стоять рядом с фактом. Призыв к действию может быть частью предложения. Ссылки на другие материалы могут быть встроены в смысловую линию. Поэтому механическое удаление «лиш-

него» часто удаляет и часть сигнала.

Историческая эволюция HCA

Эволюцию HCA можно описать как движение от статического текста к интерактивному опыту. Печатный текст был относительно стабильным. Ранний веб сохранял некоторые черты статичности, даже если содержал гиперссылки. Современный веб является динамическим: контент рендерится на клиенте, персонализируется, тестируется в экспериментах, подстраивается под устройство и поведение. Для человека это улучшает опыт, но для машины увеличивает вариативность источника и снижает воспроизводимость. В машинном чтении вариативность является видом шума, потому что она мешает надежно ссылаться на конкретные утверждения и версии.

Машиноцентричная архитектура (MCA) в деталях

Если HCA можно рассматривать как архитектуру опыта, то MCA следует рассматривать как архитектуру знания. Ее задача состоит в том, чтобы обеспечить устойчивый и проверяемый канал передачи смысла машине, отделенный от интерфейсных и коммерческих слоев. Это не означает отказ от нарратива как такового. Это означает, что нарратив и структура разведены по слоям, а ключевые утвержде-

ния доступны в форме, пригодной для извлечения без реконструкции.

Доминирование семантического слоя

В МСА ключевым является семантический слой, который задает сущности, отношения и ограничения. Доминирование семантики означает, что текстовые фрагменты привязаны к структурам: определение термина связано с идентификатором термина, правило связано с областью применимости, цена связана с тарифом, дата связана с версией. Это резко снижает вероятность ошибочных ассоциаций при извлечении.

Конец ознакомительного фрагмента.