ARENADATA

Корпоративная платформа хранения
и обработки больших данных

Миграция с зарубежных СУБД: нюансы планирования

Антон Балагаев

Директор департамента консалтинга

В данной презентации разберём:

Данный обзорный материал создавался как пособие для компаний, планирующих миграцию с инженерных систем, то есть с преднастроенной совокупности оборудования и СУБД. Однако он в полной мере покрывает и задачи, которые будут стоять при миграции обычных, не инженерных СУБД. При любой миграции следует помнить, что оборудование, необходимое для работы разных СУБД – тоже разное, не всегда удаётся переиспользовать имеющиеся мощности, и не всегда можно обойтись без дозакупки оборудования. Эта мысль зачастую не находится в фокусе внимания в проектах срочной миграции, и за невнимание к этой детали можно заплатить немалую цену, как деньгами, так и временем. В связи с этим крайне важно не игнорировать аппаратный фактор, затронутый в данном материале.

1. Каким образом достигается продолжение функционирования прикладных разработок, созданных в рамках эксплуатации инженерных систем, таких как Teradata, Exadata, Netezza на ПО Arenadata.
2. Какова структура трудозатрат миграции данных и процессов, варианты структуры проекта миграции. Рассмотрим два ключевых варианта со своими плюсами и минусами.
3. Особенности ресурсного плана под трудозатраты миграции и компетенцию вовлекаемых специалистов и команд.
4. Порядок подбора commodity-оборудования, покрывающего производительность действующей инженерной системы.

Взгляд на миграцию кода с разных сторон

Где лежит то, что мы мигрируем

1

Объекты СУБД
Кодогенерация в ETL
Хардкод в приложениях

Что именно мы мигрируем

2

Скрипты
Функциональный код
Метаданные объектов

Кто какие объекты мигрирует

3

Аналитики
Разработчики СУБД
Разработчики ППО

Куда мы сложим результат

4

ELT/ETL
Процедуры СУБД
Вызовы в приложениях

ARENADATA

Источники прикладного кода

Прикладные разработки, специфичные для используемой инженерной системы, необходимо локализовать и изменить так, чтобы их можно было использовать после миграции.

Обычно их можно найти в следующих системах и конструкциях (с удельным весом в % строк):

Код в СУБД:

Процедуры, представления, пакеты и другие объекты СУБД инженерной системы.

ETL/ELT-процессы:

Процессы загрузки данных, выполненные в ETL-инструментах, зачастую содержат исполняемый код, написанный на языке СУБД инженерной системы.

Код приложений:

Работающие над СУБД инженерной системы программы: обычно они не содержат встроенный код и хранят только ссылку на объект СУБД, к которому обращаются, однако риск прямого встраивания кода есть, и к нему следует относиться серьёзно.

Типы прикладного кода

Изменение кода, необходимое для функционирования разработок, делится на **3** части:

1. **Изменение метаданных, то есть служебных сведений, описывающих как хранятся данные и объекты, на них влияющие.**

   Это самая простая часть, так как подавляющая часть кода метаданных может быть сконвертирована автоматически. В ряде случаев всё же необходимо участие разработчиков, конвертирующих уникальные конструкции СУБД инженерной системы в аналогичные конструкции, используемые в замещающей СУБД.

2. **Изменение скриптового кода запросов и представлений, в том числе содержащихся в хранимых процедурах.**

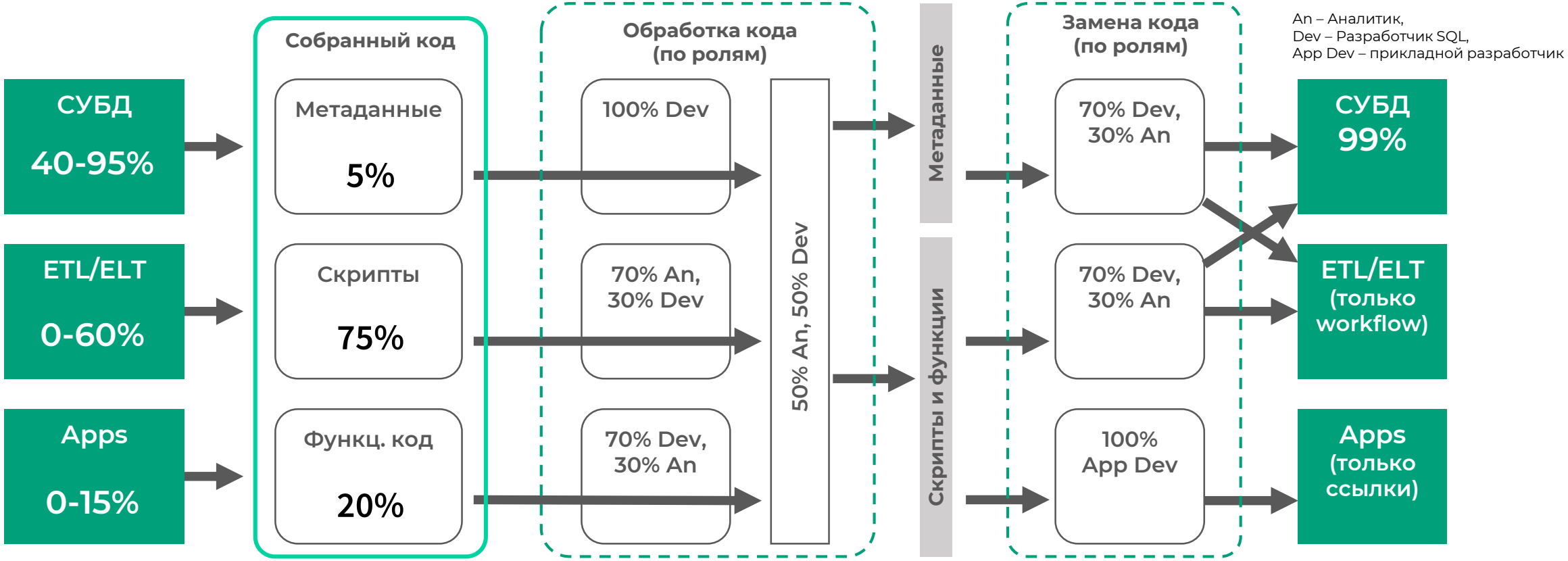
   Это код, описывающий алгоритмы преобразования данных перед их конечным представлением в слое витрин или в конкретных отчётах, сервисах или приложениях. Большая часть трудозатрат миграции заключена именно здесь, несмотря на то, что от 90% до 98% такого кода будет работать на замещающей системе вообще без каких-либо изменений. Оставшиеся же 2-10% кода будут требовать от команды миграции понимания логики алгоритмов запросов и знания различий синтаксиса СУБД.

3. **Изменение функционального кода процедур, пакетов, функций и триггеров.**

   Когда мы говорим о любом диалекте SQL, мы на самом деле говорим про два языка, работающих вместе: скриптовый и функциональный. Циклы, объявления переменных, динамический SQL и т.п. – всё это также необходимо переписать в процессе миграции

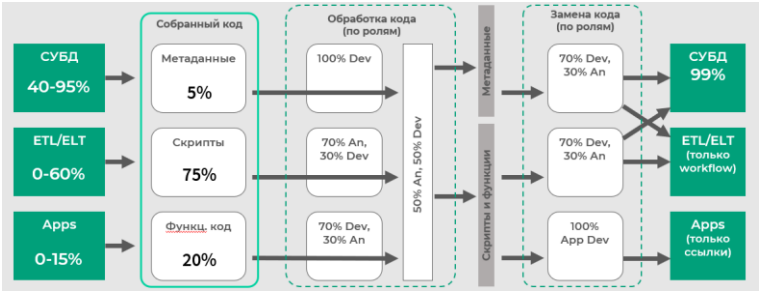
Диаграмма точек приложения усилий

Общая картина процесса миграции кода с разделением по ролям участников миграции выглядит следующим образом:

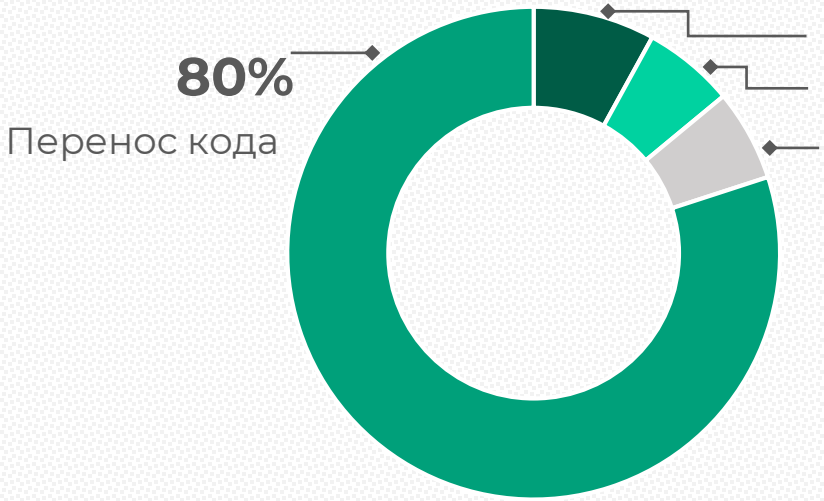


Трудозатраты вне рамок переноса кода

Основная часть трудозатрат миграции – это перенос кода. Структуру переноса мы концептуализировали ранее. В этом же разделе мы выделим конкретные задачи в рамках и за рамками переноса, и предложим индикативы по весам этих задач.



Работы вендора СУБД и системного интегратора



8% Архитектурный контроль

6% Настройка ИБ

6% Адаптация регламентной документации

\+ Непрямые зависимости от объёма сметы:

- Обучение сотрудников **4 чел/дня** на сотрудника;
- Пусконаладка СУБД **10 чел/дней.**

Стратегическая развилка миграции

Есть два сценария миграции: offload и полная миграция. Состав работ в их рамках примерно одинаков. Однако, в случае схемы Offload, добавляется работа по созданию внутренних и внешних федераций данных.

Offload

- ✓ Доступность данных как из старой системы, так и из новой (замещающей);
- ✓ Плавный переход вместо «переключения рубильником» со старой на новую системы;
- ✓ Минимизируется риск плохой адаптации пользователей.
- × Разовые затраты выше, чем при миграции;
- × Возможно лишь для систем, допускающих федеративный доступ к данным;
- × Часть проведённых работ окажется бесполезной после завершения миграции.

Миграция

- ✓ Производятся только необходимые работы, поэтому проект – короче, а разовые затраты – меньше.
- ✓ Возможна для любых замещаемых систем;
- ✓ Проект становится прозрачней за счёт отсутствия полумер.
- × Есть риск плохой адаптации пользователей (последствия: текучка, плохое качество кода);
- × Риск технического простоя при переключении на новую систему.

Декомпозиция ресурсов (пример)

No.	Point	Drivers desc	Boundaries	Results & actions	Drivers values					Labor costs, w/d						Client's labor costs, w/d	Price, RUR w/o WAT
					D1	D2	D3	D_cl	f(Di)	AN	D	SAN	SD	AR/PM	Total, w/d		
0.	OLAP Tech Platform																
1.	Quick Win 2018																
1.1.	Tech Platform Proof of Concept - Stage 1																
1.1.0.	PoC prerequisites																
1.1.0.1.	QuickWin process shortlist: to be tested within PoC																
1.1.0.1.1.	Online delivery to Data Lake / ODS (MVP0+1+4)																
	Online interfaces capabilities (REST, clickstream, etc)																
	Plain delivery of 1-2 stream types predefined by customer (through PubSub interfaces)																
	PoC: Stream analytics example (negotiated scenario)																
1.1.0.1.2.	Batch delivery to DDS/DMs (MVP0+2)																
	Batch interface capabilities (parallel dblinks, external tables, etc)																
	ETL to DDS through ODS (two landing points; MVP0+2)																
1.1.0.1.3.	Cloud capabilities (MVP3)																
	Google cloud access (data virtualization)																
1.1.0.1.4.	Self-service BI (MVP0)																
	Data preparation capabilities example video																
	Report compilation capabilities example video																
1.1.0.2.	Hardware/Cloud sizing and allocation																
1.1.1.	Deployment and setup of basic clusters																
	Arenadata Hadoop (MVP0 PoC)																
	Arenadata Streaming (MVP0+1+4 PoC) - based on Kafka + NiFi																
	Arenadata DB (MVP2 PoC) - based on OSS Greenplum Database																
	Arenadata Analytical Workspace (MVP0) - based on Zeppelin																
1.1.2.	PoC in accordance with prerequisites, leading to deployed MVP0 (to be detailed according to 1.1.0.)																
1.1.3.	Training for administrators of Arenadata Hadoop, Arenadata DB, Arenadata Streaming																
2.	Main Project																
2.1.	Tech platform PoC - Stage 2																
2.1.0.	PoC prerequisites																
2.1.0.1.	Upgrade to MVP4 process shortlist: to be tested within PoC																
2.1.0.1.1.	Batch delivery to DDS/DMs (MVP0+2)																
	Calc formulas management through Glossary (MVP2) within ETL: ODS->DDS step																
2.1.0.1.2.	Cloud capabilities (MVP3)																
	Hybrid capabilities verification																
	Billing example & PoC																
2.1.0.1.3.	Self-service BI (MVP4)																
	Report marketplace access management example																
2.1.0.1.4.	Collaboration and Data Governance capabilities (MVP2+)																
	Glossary capabilities (formulas management, tagging, search)																
	Data Lineage creation PoC																
	ER-model management tool PoC																
	Bonus DG capabilities if any																
2.1.0.5.	Hardware/Cloud sizing and allocation																
2.1.1.	Software deployment and setup																
	Data Governance software (Arenadata or OEM Informatica)																
	Glamorous BI-Tool with report marketplace feature (Tableau or smth)																
	Collaborative ER-tool (SAP PowerDesigner or smth)																
2.1.2.	PoC in accordance with prerequisites, leading to deployed MVP4 (to be detailed according to 2.1.0.)																
2.2.	Data Lake & Detail Data Store																
2.2.1.	Analysis stage																
	Target ER-model definition (will form the DDS)																
	Sources profiling (defining entities and atributes required for target model)																
	Development & negotiation of SQL-prototypes for relational entities (SQL queries formed of source's data producing datasets, identical to DDS entities in terms of both metadata and data)																
	Defining events & complex events for streaming analytics FR																
	Fixation of delivery algorithms for non-relational and streaming data																
	Data quality checks lists preparation for ODS & DDS entities																
	Data unification rules stating for entities of DDS																

Для каждой строки декомпозиции нужно определить:

- Результат (артефакт, созданный в рамках исполнения строки);
- Ограничения (что не делается);
- Измеримые метрики, изменение значений которых влияет на объём трудозатрат (драйверы), и формулы такой зависимости.

Порядок подбора оборудования

Раньше жили в такой парадигме. Если удастся остаться в ней – это идеально.

1

Пилот на типовой конфигурации

Небольшой кластер, сбалансированный по нагрузке на:

- CPU
- Память
- Диски
- Сеть

2

Адаптация конфигурации

- Выявление выполняемых часто и ёмких запросов.
- Определение профиля нагрузки инженерной системы.
- Коррекция соотношений количества ядер CPU, оперативной памяти и типов и объёма дисков.

3

Оптимизация TCO

- Выбор: максимальные по ёмкости и мощности узлы или оптимальные по стоимости на объём.
- Выбор: контрактование с сопровождением на горизонт TCO или ежегодная закупка.
- Запросы предложений к поставщикам оборудования.

Возможные меры

если нового железа нет и не предвидится

1. Сжать всё, что возможно, максимально эффективным из доступных алгоритмов сжатия. На сжатие расходуется ресурс CPU, так что процессы, завязанные на сжатые данные, замедлятся, однако так получится создать запас для роста или высвободить часть оборудования. Как и прежде, следует находить баланс между расходом CPU и силой сжатия, но точка баланса теперь сильно смещена в сторону повышения компрессии.

2. Провести проверки соответствия политик резервирования и катастрофоустойчивости требованиям бизнеса в новое время. Если требования снизились, то часть мощностей хранения под оперативные бэкапы (не на лентах) можно высвободить.

3. Рассмотреть возможность объединения ранее разделённых контуров/сред (разработка, тестирование, UAT, preprod). Решение, принятое в сытые годы, может поменяться.

4. Освободившееся оборудование использовать под СУБД, обладающие следующими свойствами:
   - Колоночное хранение – сильно уменьшает объём места, занимаемого данными;
   - Эффективное сжатие данных (например, z_std);
   - Возможность федерализации запросов – как входящих, так и исходящих.

При этом нельзя забывать про стандартные enterprise-требования: наличие инструментов резервного копирования, наличие визуальных средств администратора (мониторинг, расширение, сопровождение), коннекторов к другим СУБД и, конечно, поддержки с SLA и готовности вендора исправлять код ПО в соответствии с заведёнными тикетами заказчика по багам.

Риски переиспользования оборудования

Если у клиента есть инженерная система или иное оборудование, адаптированное под конкретный софт, и из-за тех или иных причин софтом пользоваться более возможности нет, то у клиента неизбежно возникнет большой соблазн его переиспользовать. Что в этом случае учитывать:

1. Проверить, будут ли работоспособны сервера в случае зачистки и установки другого ПО. Нужно исключить риск неработоспособности физических компонент при смене софтверных. Например, если ранее использованный софт содержит в себе драйверы, без которых оборудование бесполезно, и получить эти драйверы иным путём нельзя.

2. Если оборудование всё же неработоспособно в комплексе, то можно проверить, переиспользуемы ли отдельные его компоненты: CPU, RAM, диски, RAID-контроллеры, сетевые карты.

3. Вопрос пере-используемости оборудования нужно поднимать в контексте архитектуры нового ПО, под которое будут использованы компоненты. У разного ПО разные узкие места и разные аппаратные потребности.

Если достать оборудование – не проблема

1. Жить в старой парадигме и давать сайзинги, заточенные на максимальную производительность, уже никто не может себе позволить. Если вы сможете достать для клиента новое оборудование, то почти наверняка кратно дороже, чем раньше. В такой ситуации нужно оптимизировать стоимость терабайта хранения.

2. Всегда рассматривайте опцию температурного хранения данных. В текущих реалиях может оказаться выгоднее выделять холодные зоны в основных кластерах Arenadata DB или Arenadata Quick Marts, а не выносить эти данные в Arenadata Hadoop, так как ADB и ADQM дают более эффективное сжатие при колоночном хранении. Экономия на лицензиях из-за использования более дешёвого ПО теперь несущественна по сравнению с экономией на оборудовании из-за сжатия и колоночного хранения.

3. Предлагайте варианты с бОльшим объёмом дисков или общей ёмкости на тот же объём CPU и RAM.

4. Могут возникнуть ситуации, в которых клиент не может использовать ничего кроме имеющихся мощностей, и эти мощности никак не бьются с лучшими практиками подбора оборудования. Раньше в таких случаях клиенту отказывали. Сейчас для клиента это может быть вопросом выживания и нужно не отказываться, а сконцентрироваться на damage control от нецелевой конфигурации. В чём следует убедиться:
   - Правдами и неправдами обеспечить равномерную нагрузку на узлы распределённых систем.
   - Всеми силами избегать использования общих с другими системами коммутаторов, сетевых каналов или находить какие-то варианты лимитирования нагрузки на ваше оборудование от сторонних систем.
   - Клиент должен понимать, что нецелевая конфигурация – всё равно риск, и вендор на себя этот риск взять не сможет. Arenadata будет проактивно идти навстречу в спорных ситуациях.

Матрица соответствия программных продуктов

Мигрируемый продукт	Тип нагрузки / задача	Куда мигрируем
Oracle DB, MS SQL, SAP Sybase ASE, MS Access, IBM DB/2	OLTP	Arenadata Postgres
Oracle (< 1000 TPS)	OLAP / RAC / Exadata (DWH)	Arenadata DB
Oracle, MS SQL, Teradata	Витрины данных	Arenadata Quick Marts
Teradata, Netezza, Vertica, Exasol, Impala, Presto, Tanzu/Pivotal Greenplum, SAP Sybase IQ	DWH, Реляционная MPP СУБД	Arenadata DB
MongoDB, Cassandra, DynamoDB	Хранение документов, JSON	Arenadata Hadoop: Hbase
Redis, Hazelcast	In-memory Key-Value	Picodata Grid
Elasticsearch	Поиск, анализ логов	Arenadata Log Search
Snowflake, RedShift	Реляционная облачная MPP, DWH	Arenadata DB
Azure SQL, Google Big Query	Реляционная облачная СУБД	Arenadata DB, Arenadata Hadoop
SAP HANA	In-memory RDBMS	Arenadata Quick Marts + Arenadata DB
Продукты Cloudera / Hortonworks	Экосистема Hadoop	Arenadata Hadoop
Confluent Kafka	Pub/Sub брокер сообщений	Arenadata Streaming: Kafka
IBM Streams, SAS Event Streams Processing, Azure Stream Analytics, TIBCO Streaming, Cloudera DataFlow	Потоковая аналитика	Arenadata Streaming: Kafka (KSQL)
Amazon Timestream, HCL Informix, InfluxDB Enterprise, Kdb+	СУБД для анализа временных рядов	Arenadata QuickMarts или TimescaleDB

ARENADATA

Корпоративная платформа хранения
и обработки больших данных

Оцените потенциал Big Data и Open Source вместе с Arenadata Enterprise Data Platform

- Узнайте больше на **arenadata.tech**
- Скачайте бесплатно на **store.arenadata.io**
- Получите консультацию по почте **info@arenadata.io**