Сравнительное исследование технологий доступа к реляционным БД на основе нестандартных моделей данных

Харичкин Александр
МФТИ(ГУ)
16 июня 2011

Введение

- Характеристики информационных систем
  - Объем кода
  - Сложность модели данных
  - Масштабируемость
  - Возможность повторного использования функциональности
  - Конфигурируемость приложения
  - Возможность изменять логику в режиме реального времени
- Задачи, решаемые при проектировании системы
  - Выбор подхода к хранению данных
  - Разработка объектной модели приложения
  - Описание процессов и бизнес-правил

Постановка задачи

- I. Сравнение технологий доступа и моделей данных по производительности, гибкости и простоте реализации
  - Подходы к хранению данных в БД
    - «Плоские» модели
    - «Метамодели»
      - Адаптивные ОМ
      - ММ типа NetCracker
      - Нестандартные объектные модели
  - Технологии доступа
    - JDBC
    - EJB 2.x (Entity)
    - ORM (JPA etc)
- II. Исследование «нестандартной» однотабличной модели данных

Реализованные подходы и технологии

Технология\ Подход	Relational DB	MetaModel («NetCracker»)	MetaModel (Other)
JDBC	+	–	+/-
EJB 1.x, 2.x	–	+	–
JPA	+	+	-

- Дополнительные возможности
  - Кэширование (на стороне Application-сервера)
  - Перенесение части нагрузки на сервер БД
  - Денормализация

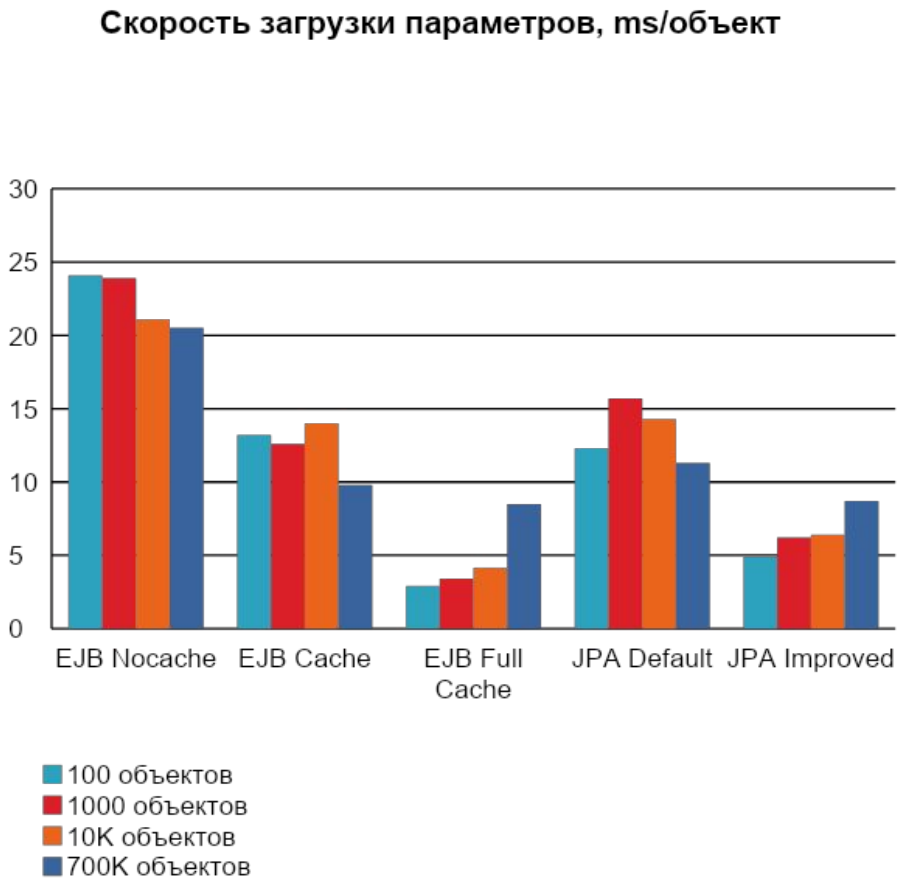
Методика тестирования производительности

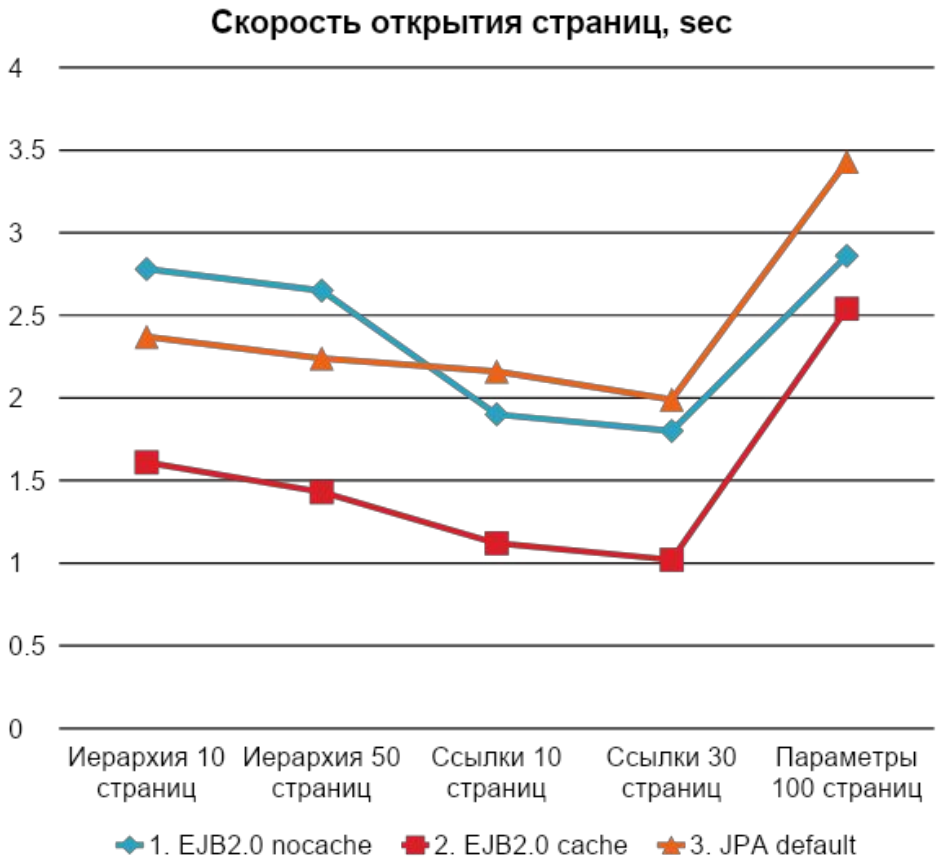
- Субъективные тесты (1 пользователь)
  - WEB-приложение, сценарий типа «выполнить следующие действия с данными»
- Формализованные нагрузочные тесты
  - Поиск объектов и отображение страницы параметров
    - Поиск по идентификатору объекта
    - Обход дерева иерархий
    - Переходы между объектами по ссылкам
- Результаты
  - Время загрузки параметров и открытия страницы
  - Количество обращений в базу при выполнении «стандартных» операций

Исследуемые реализации

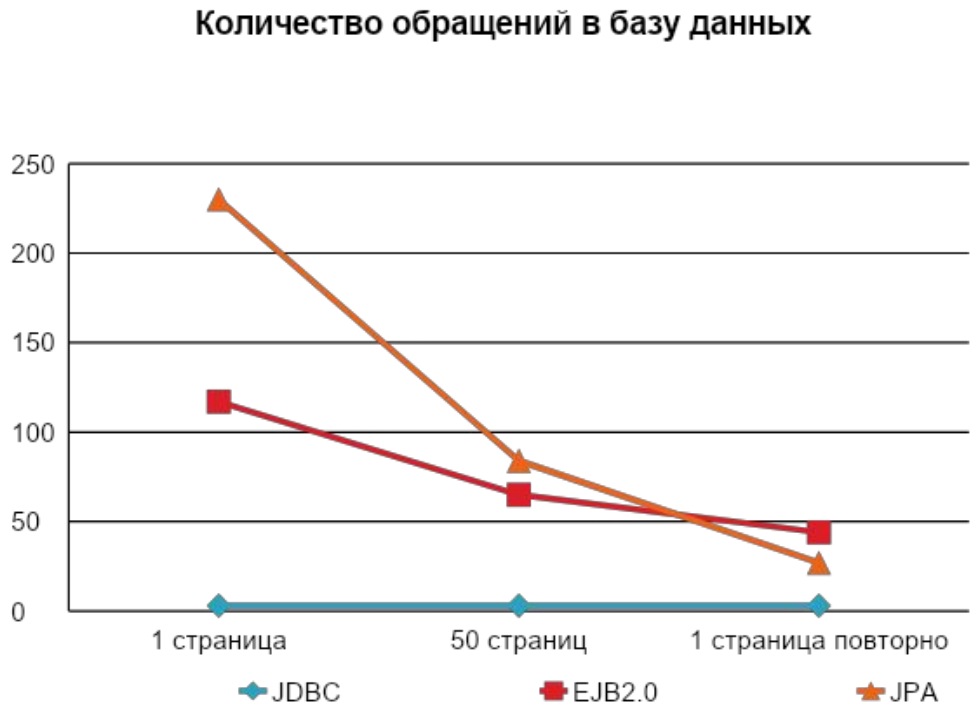
- Enterprise JavaBeans 2.0
  - Стандартная реализация
  - Усовершенствование в виде кэширования метаданных на стороне Application-сервера
  - Полное кэширование данных на сервере
- Java Persistence API
  - Реализация без дополнительных настроек
  - Усовершенствование настроек для обеспечения наибольшей производительности

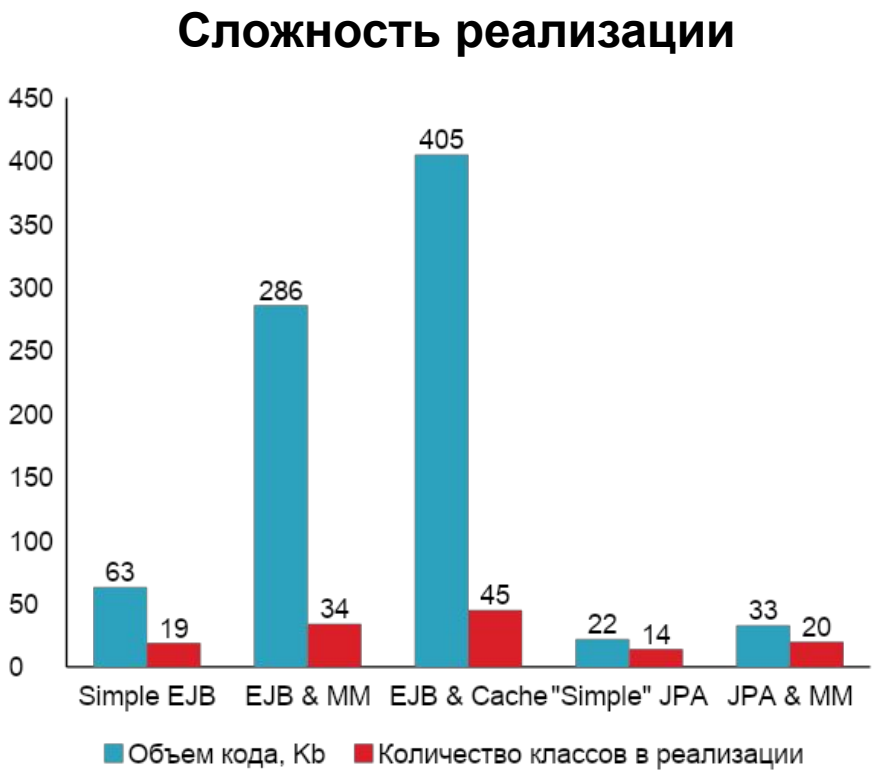
Результаты


Скорость загрузки параметров, ms/объект
30
25
20
15
10
5
0
EJB Nocache
EJB Cache
EJB Full Cache
JPA Default
JPA Improved
100 объектов
1000 объектов
10K объектов
700K объектов


Скорость открытия страниц, sec
4
3.5
3
2.5
2
1.5
1
0.5
0
Иерархия 10 страниц
Иерархия 50 страниц
Ссылки 10 страниц
Ссылки 30 страниц
Параметры 100 страниц
1. EJB2.0 nocache
2. EJB2.0 cache
3. JPA default

Результаты


Количество обращений в базу данных
250
200
150
100
50
0
1 страница
50 страниц
1 страница повторно
JDBC
EJB2.0
JPA


Сложность реализации
450
400
350
300
250
200
150
100
50
0
63
19
286
34
405
45
22
14
33
20
Simple EJB
EJB & MM
EJB & Cache
"Simple" JPA
JPA & MM
Объем кода, Kb
Количество классов в реализации

Выводы

- В среднем технология JPA обеспечивает несколько меньшую производительность по сравнению с EJB2.0
  - Однако в большинстве случаев это различие *несущественно*
  - В некоторых обстоятельствах *JPA превосходит EJB* по быстродействию
- Реализация технологии EJB намного более затратна в терминах объема и сложности кода
  - JPA «выигрывает» засчет *простоты реализации* и множества *утилит*, упрощающих разработку
- EJB позволяет вносить усовершенствования на уровне кода *программируемых модулей*
  - В то время как в JPA все это делается исключительно *настройками*

«Нестандартные» метамодели

- Модель с единственной таблицей
  - Любой объект, тип, атрибут, бизнес-правило представлены одной записью в таблице
- Цель
  - Введение дополнительных уровней абстракции для повышения гибкости приложения и Service Availability
  - «Вложение» моделей друг в друга, исследование совместимости моделей данных
  - Влияние степени нормализации данных на быстродействие и сложность приложения

Типичные примеры ИС и предлагаемые решения

- Plain DB + JDBC
  - Data Warehouse
- Plain DB + JPA
  - Social Networks
- Metamodels + JPA
  - Bookstore
  - E-Shop
- Metamodels + …
  - Telecom OSS (NetCracker)
  - Document Workflow Framework (Adaptive Object Model)
  - Search Engine (Key-Value model)

Заключение

- Разработано несколько реализаций ИС на основе EJB 2.x, JPA и JDBC
  - Введены усовершенствования реализаций
    - Кэширование EJB
    - Дополнительные настройки JPA
- Проведено испытание производительности реализаций автоматическими нагрузочными тестами
  - Результат - данные для многокритериального анализа в задаче принятия решения
  - Сделаны выводы о применимости технологий для различных классов ИС
- Рассмотрена на уровне идеи абстрактная однотабличная модель данных
  - Рассмотрены методические аспекты таких моделей
  - Направление дальнейшего исследования – получение численных результатов испытаний подобных моделей

Thank you!