Поисковый сервис контактной информации на основе системы мониторинга пользователей социальных сетей

Студент: Андриишин А.И.
Научный руководитель: Власов А.И.

Цель работы

Цель работы:

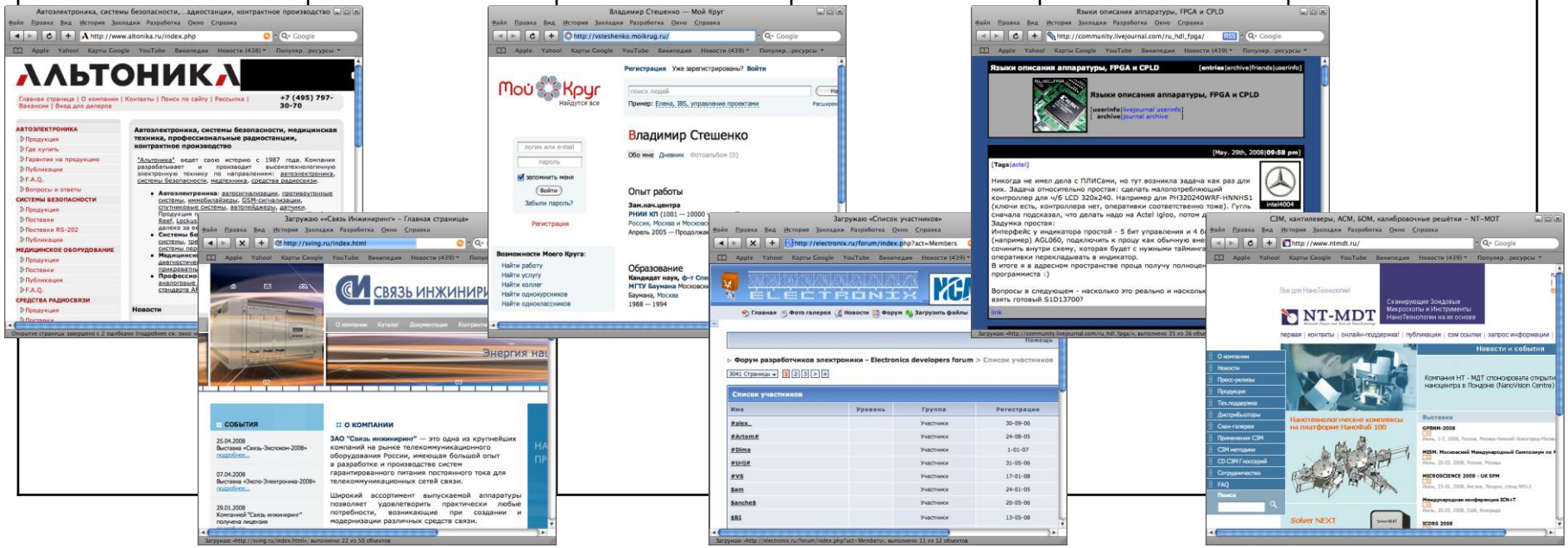
- Создание сервиса поиска контактной информации на основе системы мониторинга пользователей социальных сетей.

Решаемые задачи:

- Исследование и классификация информационных ресурсов, посвященных разработке ЭС
- Исследование и анализ принципов построения распределённых вычислительных систем
- Проектирование архитектуры поискового сервиса
- Разработка алгоритмов агрегации данных
- Разработка программного обеспечения на основе разработанной архитектуры
- Развёртывание системы, тестирование и отладка

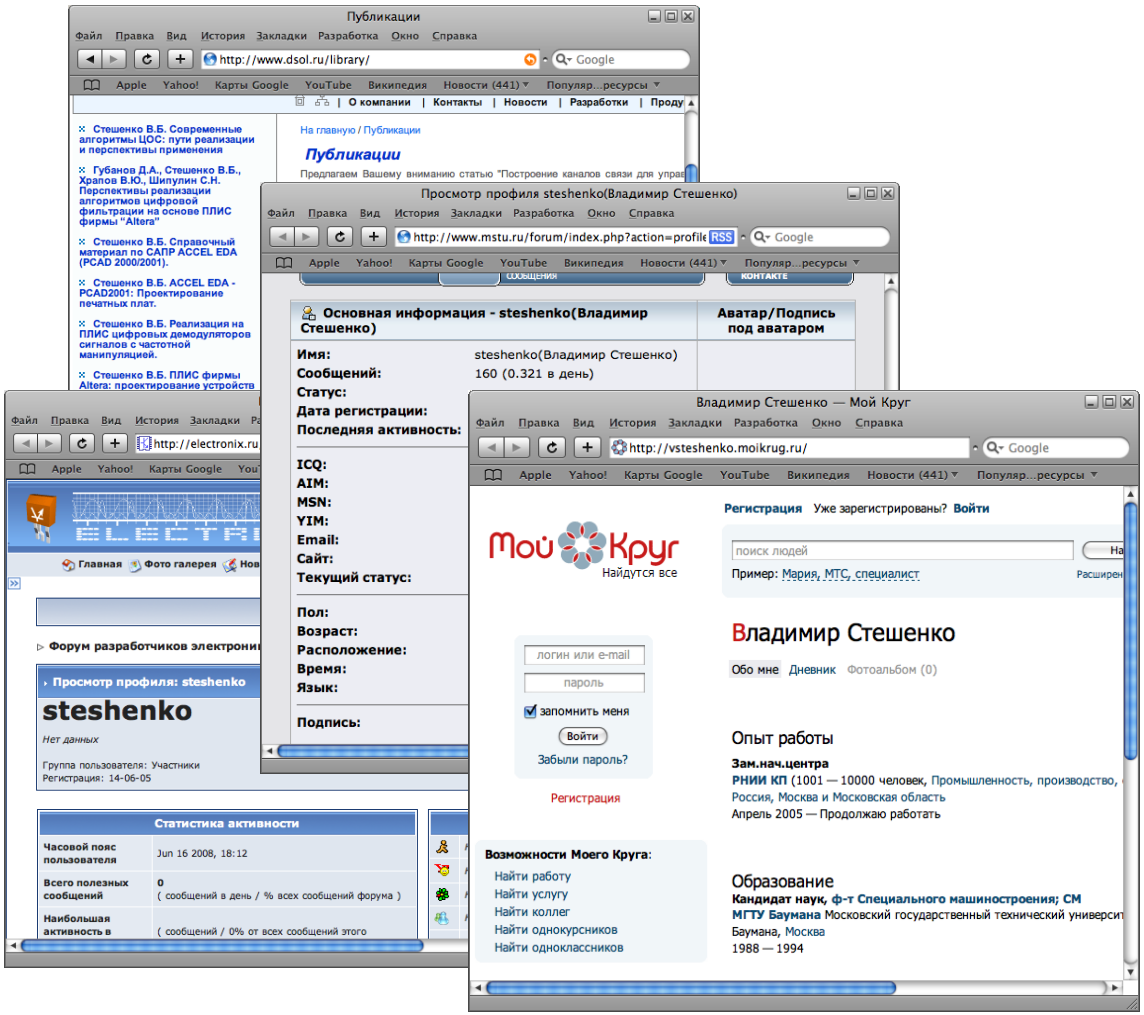
Классификация информационных систем, посвящённых разработке ЭС

Представительства компаний и организаций	Персональные страницы разработчиков	Социальные сети	Блоги	Форумы	Сайты поиска работы
www.altonika.ru www.niiin.ru www.sving.ru www.ntmdt.ru и т.д.	library.tsc.ru/~oleg/ www.yanson.ru/?part=contact&page=me и т.д.	www.moikrug.ru www.linkedin.com и т.д.	www.elementy.ru www.community.livejournal.com/ru_hdl_fpga/ и т.д.	forum.ixbt.com www.electronix.ru/forum и т.д.	www.job.ru www.hh.ru www.freelance.ru www.personal.ru и т.д.



Описание проблемы

Разрозненность информации об участниках различных технических сообществ и, как следствие, отсутствие единого информационного пространства



Предпосылки:

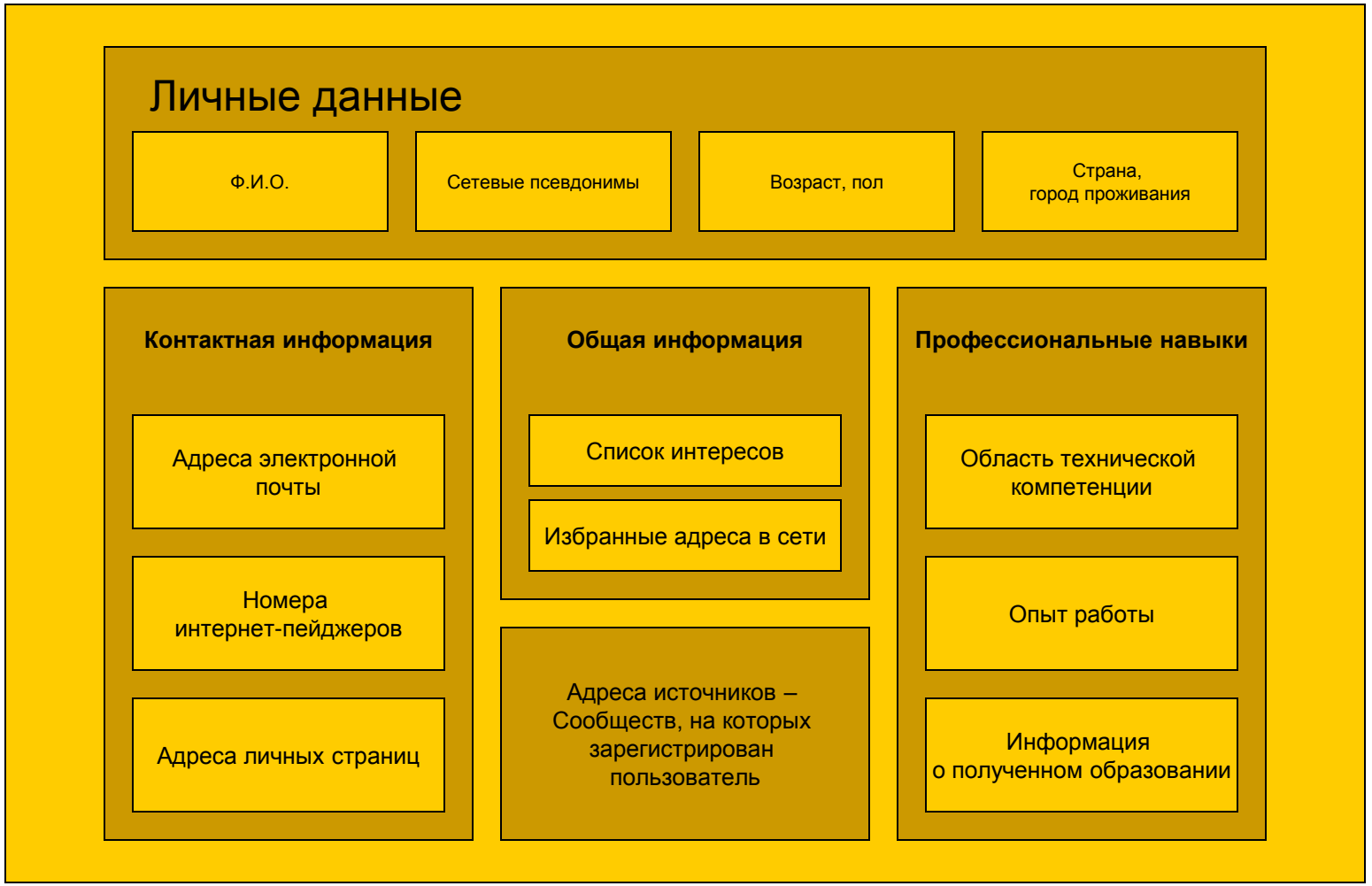
1. Большое количество различных информационных ресурсов
2. Необходимость регистрации на каждой из ИС
3. Отсутствие взаимной коммуникации и централизованного поиска

Постановка задачи:

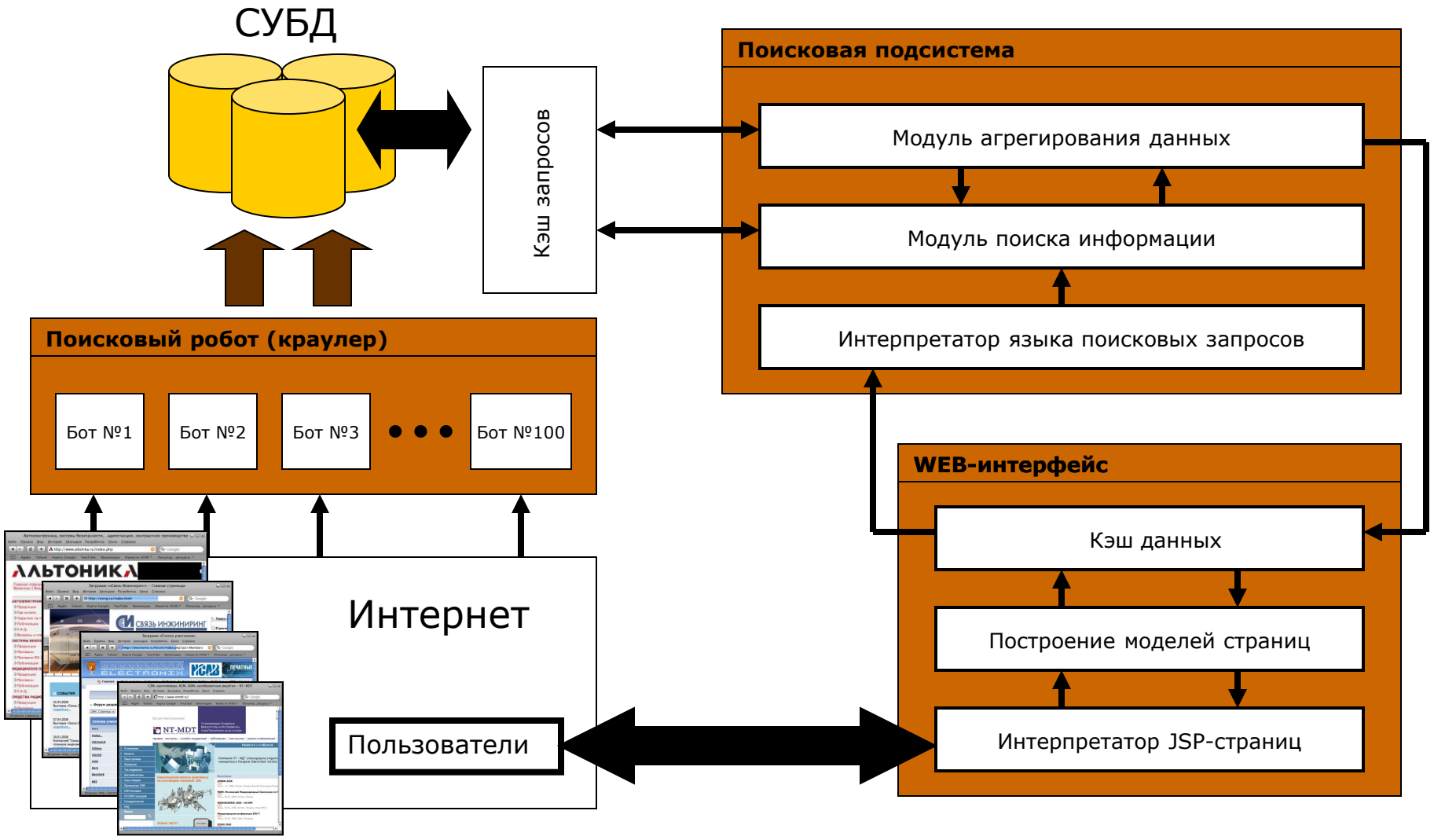
Разработать систему личностноориентированного поиска, способную объединить профили пользователей из различных технических сообществ и обладающую следующими возможностями:

- Осуществление поиска данных среди профилей пользователей различных информационных ресурсов
- Возможность нахождения всех профилей, принадлежащих одному и тому же человеку
- Отображение найденной информации в едином окне, при этом:
  - Должны отображаться источники информации
  - Должна указываться вероятность, с которой найденная информация действительно принадлежит одному и тому же человеку
  - Заведомо ложная или неверная информация должна отфильтровываться

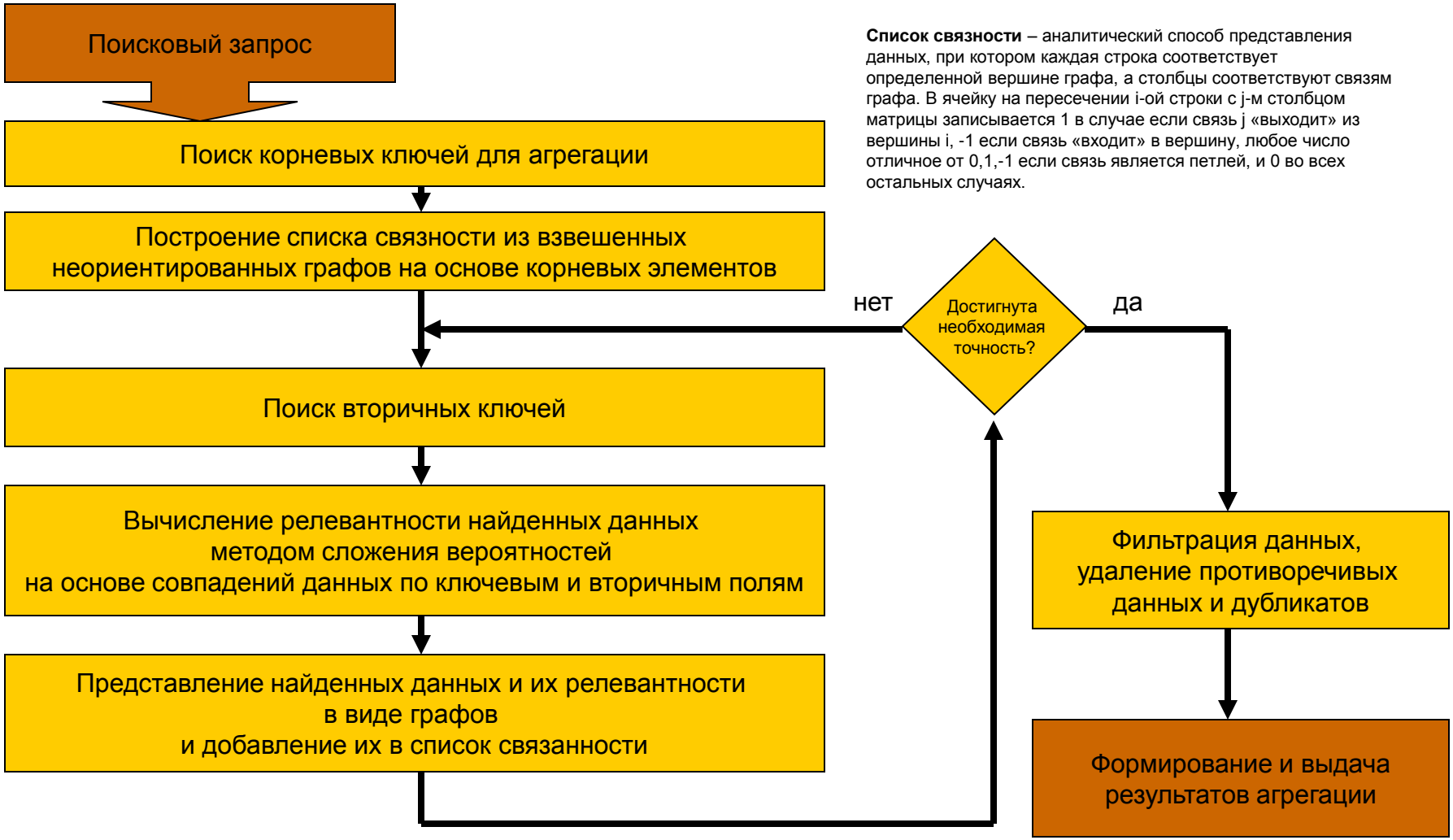
Результат личностноориентированного поиска – интегрированная анкета



Архитектура системы

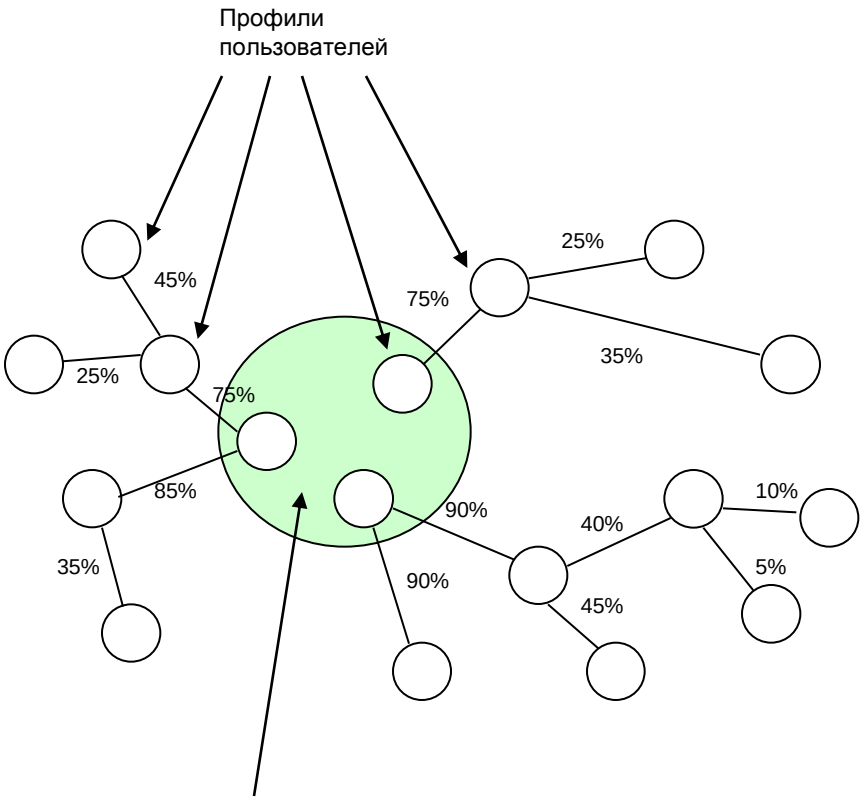


Алгоритм агрегации данных



Список связности – аналитический способ представления данных, при котором каждая строка соответствует определенной вершине графа, а столбцы соответствуют связям графа. В ячейку на пересечении i-ой строки с j-м столбцом матрицы записывается 1 в случае если связь j «выходит» из вершины i, -1 если связь «входит» в вершину, любое число отличное от 0,1,-1 если связь является петлей, и 0 во всех остальных случаях.

Алгоритм обхода списка связности



Профили, найденные в первой итерации алгоритма связывания (корневые профили)

Присвоим $d[a] \leftarrow 0,\ p[a] \leftarrow a$

Для всех $u \in V$ отличных от a

присвоим $d[u] \leftarrow \infty$

Пока $\exists v \notin U$ с $d[v] < \infty$

Пусть $v \notin U$ – вершина с минимальным $d[v]$

Добавим вершину v к U

Для всех $u \notin U$ таких, что $vu \in E$

если $d[u] > d[v] + w[vu]$ то

изменим $d[u] \leftarrow d[v] + w[vu]$

изменим $p[u] \leftarrow p[v], u$

- ***Список условных обозначений:***
- *V — множество вершин графа*
- *E — множество ребер графа*
- *w[ij] — вес (длина) ребра ij*
- *a — вершина, расстояния от которой ищутся*
- *U — множество посещенных вершин*
- *d[u] — по окончанию работы алгоритма равно длине кратчайшего пути из a до вершины u*
- *p[u] — по окончании работы алгоритма содержит кратчайший путь из a в u*

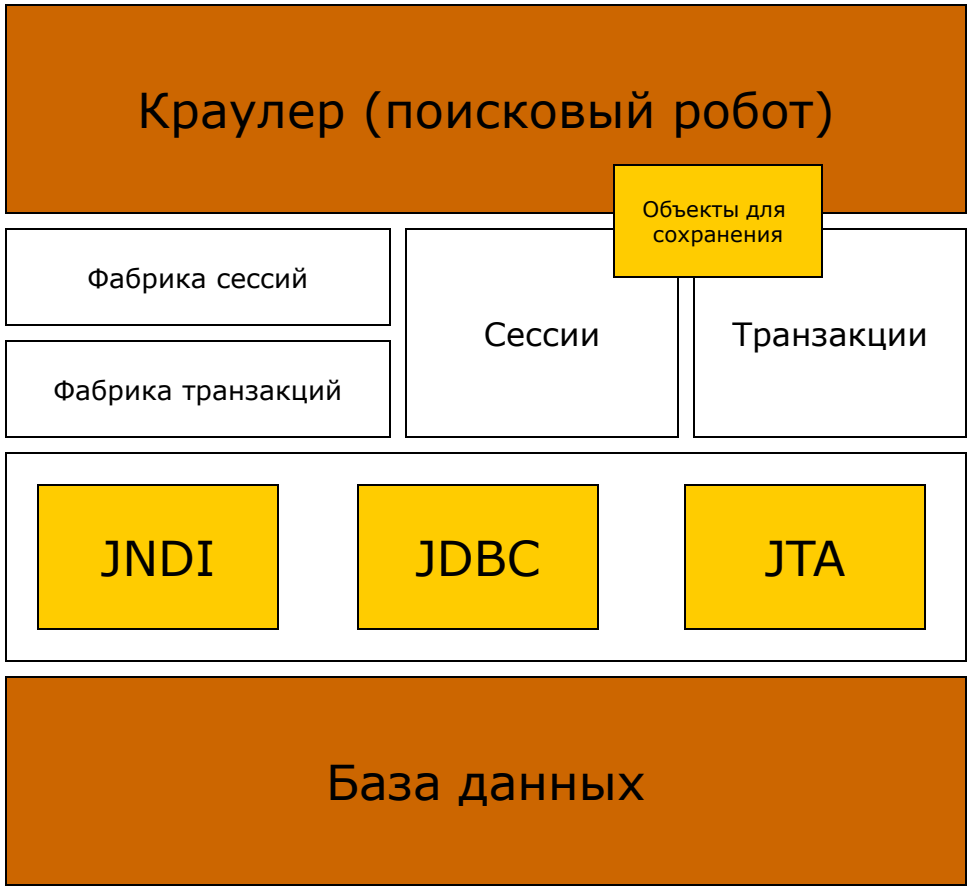
Алгоритмы индексации

Прямой индекс - записи отсортированы по значениям данных

Обратный индекс - записи отсортированы по элементам данных

Тип индекса	**Особенности работы**
B-tree	■ самый быстрый индекс по критериям поиска и обновления данных ■ поддерживает малое количество операций ■ Использует бинарное дерева для хранения индексированной информации
Hash	■ Скорость создания индекса медленнее, чем у B-tree в 5 раз ■ Скорость поиска аналогична индексу B-tree ■ Выгодно применять в очень редких ситуациях для типов данных, построение хеш-функции для которых меньше чем создание бинарного дерева
GiST	■ Возможно реализовать любую операцию, написав собственный модуль к PostgreSQL ■ Скорость обновления зависит от реализации, но в целом ниже чем у B-tree и Hash из-за необходимости соблюдать определённую структуру алгоритма
GIN	■ Поддерживает операции полнотекстового поиска и поиска вхождения в массив, скорость выборки данных быстрее чем у GiST в 3 раза ■ Скорость обновления индекса медленнее чем скорость обновления GiST в 10 раз ■ Скорость создания медленнее чем скорость создания GiST в два раза ■ Поддерживает групповое обновление ■ Выгоднее использовать массовую вставку данных с последующим построением индекса, нежели построчное обновление

Механизм сохранения данных



Краулер

Комплекс программ, который занимается сбором, анализом и индексированием информации.

Java Naming and Directory Interface (JNDI)

Набор программных интерфейсов для работы со всеми существующими системами именования и службами каталогов.

Java Database Connectivity (JDBC)

Промышленный стандарт взаимодействия между Java и широким спектром баз данных. JDBC предоставляет основанный на SQL API для доступа к базам данных.

Java Transaction API (JTA)

Высокоуровневый API, независящий от реализации и используемых протоколов, который позволяет приложениям и серверам приложений осуществлять доступ к транзакциям.

Интерфейс интегрированной анкеты

Строка поискового запроса

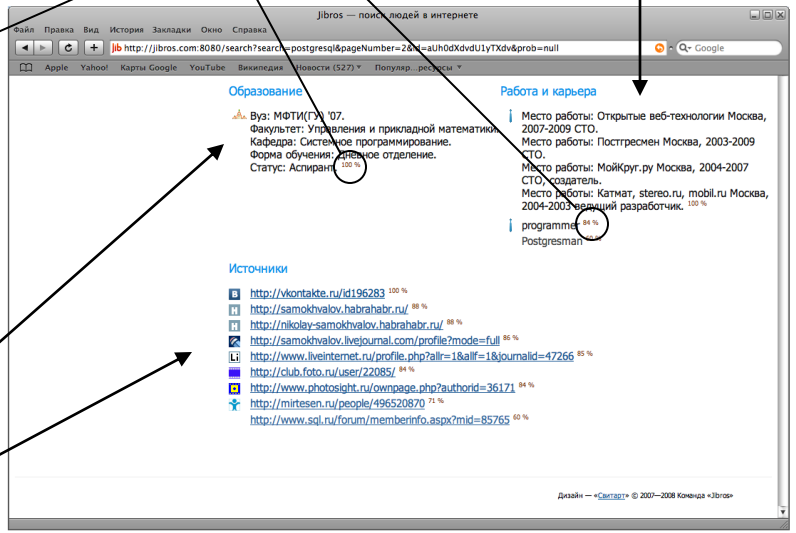
Личные данные

Адреса электронной почты и номера интернет-пейджеров

Личные страницы пользователя

Уровень достоверности показываемой информации

Опыт работы

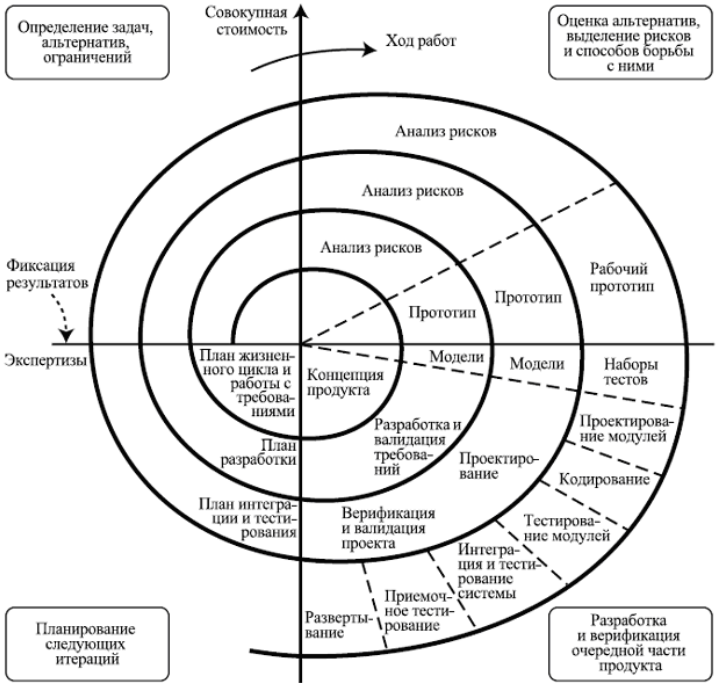


Список найденных пользователей

Образование

Перечень источников

Методика тестирования



Уровни тестирования

Модульное тестирование (юнит-тестирование) — тестируется Минимально возможный для тестирования компонент, например, отдельный класс или функция

Интеграционное тестирование — проверяет, есть ли какие-либо проблемы в интерфейсах и взаимодействии между интегрируемыми компонентами

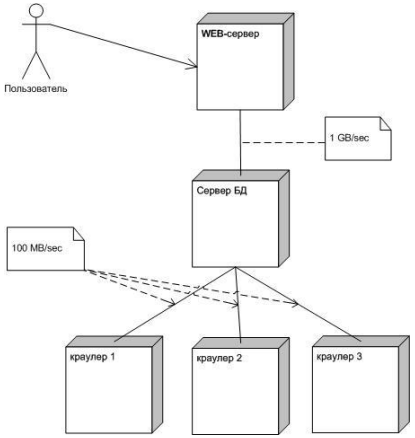
Системное тестирование — тестируется интегрированная система на её соответствие исходным требованиям

Альфа-тестирование — имитация реальной работы с системой штатными разработчиками

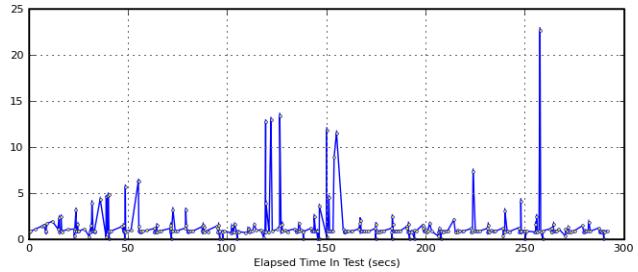
Бета-тестирование — выполняется распространение версии с ограничениями, с тем чтобы убедиться, что продукт содержит достаточно мало ошибок.

Проведение нагрузочного тестирования

Структура тестового стенда



Результаты нагрузочного тестирования для 30ти конкурирующих пользователей (измерение время отклика)



НАЗВАНИЕ КОМПЬЮТЕРА	*КОНФИГУРАЦИЯ*
WEB-сервер	Материнская плата MB Intel S5000PAL Корпус Intel SR2500 Процессор CPU Intel Xeon 5130 Dual Core 2,0GHz Винчестер 750,0 Gb HDD Seagate (ST3750330NS) Barracuda (4 штуки) Оперативная память FB-DIMM 2048Mb PC2-5300(667Mhz) ECC (8 штук) DVD+-RW slim Nec AD-5540A
Сервер БД	Материнская плата MB Intel S5000PAL Корпус Intel SR2500 Процессор CPU Intel Xeon 5130 Dual Core 2,0GHz Винчестер 750,0 Gb HDD Seagate (ST3750330NS) Barracuda (6 штук) Оперативная память FB-DIMM 2048Mb PC2-5300(667Mhz) ECC (8 штук) Комплект для крепежа 6-го Hotswap HDD ASR2500SIXDRV DVD+-RW slim Nec AD-5540A
краулер 1 краулер 2 краулер 3	Процессор Intel Core2 6320 @ 1.86 Материнская плата ASUS P5B SE Socket775,intel® P965, 4*DDR2,PCI-E16x,ATA133,SATA+RAID,ADI HDA 6ch,GigLan,ATX Винчестер 250,0 Gb HDD Seagate Barracuda Корпус InWin C588T ATX 350W DVD-RW NEC ND-4550A

Расчёт динамики роста объема сохраненной информации

Хранение профилей
По результатам опытной эксплуатации размер одного сегмента партиционированной таблицы размером 10 миллионов профилей составляет **11154 Мб**, из них:

Размер данных: **2412 Мб**
Размер индекса: **8742 Мб**

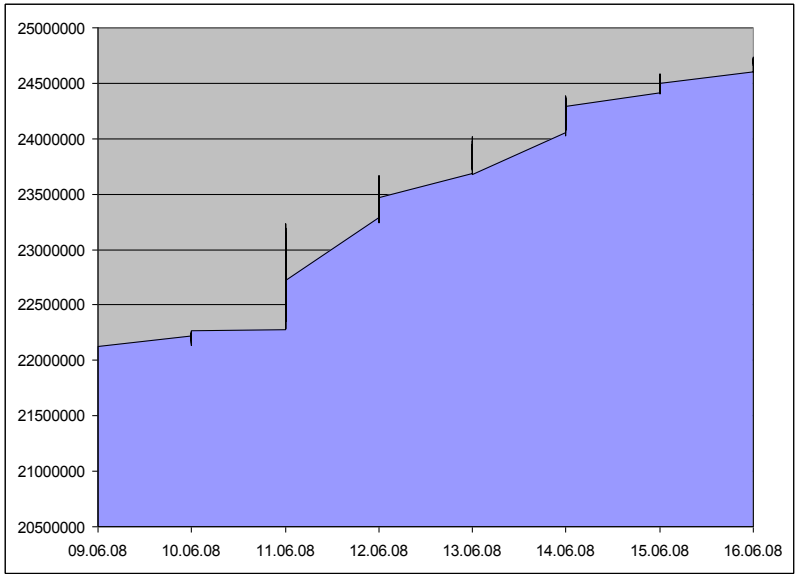
Хранение информации для агрегирования
Размер одного сегмента партиционированной таблицы размером 100 миллионов записей составляет **28 Гб**, из них:

Размер данных: **11 Гб**
Размер индекса: **17 Гб**

Статистика отношения количества сохраненных единиц информации для агрегирования к количеству сохраненных профилей:

420000000 / 24500000 = **17.4** – среднее количество единиц информации на один профиль пользователя.

Измерение общего количества найденных и проиндексированных профилей за неделю



При сохранении текущей динамики скорости обновления базы данных (10 миллионов профилей в месяц) ежемесячное увеличение объема данных составит:

11Гб + 28Гб * (100/10) * 17.4 = 60 Гб.

Результаты работы

- Проведен анализ принципов построения поисковых систем
- Разработана архитектура личностноориентированной поисковой системы
- Проведена разработка программной части с использованием технологий Java Enterprise Edition и PostgreSQL 8.3 в качестве СУБД
- Произведено развёртывание системы и запуск в опытную эксплуатацию

- С помощью разработанной методики было произведено тестирование поисковой системы, в результате которого были определены:
  - Скорость заполнения базы данных, которая составила 60Гб в месяц.
  - Скорость обновления базы данных (10 млн. профилей в месяц)
  - Максимальное количество поисковых запросов в сутки (2,6 млн.)