– Поиск похожих документов**.** Технология основана на выявлении ограниченного числа наиболее значимых слов в исходном документе и поиску по их списку похожих документов в системе (так называемый поиск с учетом расстояния между ключевыми словами – насколько близко они друг от друга в тексте).

– Нечеткий поиск. Обеспечивается выполнение поиска в текстовых данных, содержащих орфографические ошибки или опечатки. На практике такой подход позволяет, например, выявлять ситуации преднамеренной замены символов, например, буквы «О» на цифру «0» с целью усложнения поиска информации.

– Извлечение сущностей. Обеспечивается возможность распознавания и извлечения из текста определенной значимой информации – сущностей (люди, места, организации, номера телефонов, продукты или услуги и т.п.). В основе используемой технологии лежит частичный синтаксический анализ и лингвистические шаблоны, содержащие лексическую, морфологическую и синтаксическую информацию.

4. ВЫЗОВ И ЗАГРУЗКА ПРОГРАММЫ

Службы, необходимые для функционирования 3i Search Platform 3.x, запускаются автоматически при старте серверов.

4.1. Установка L6HDUFK3ODWIRUP[

Установка 3i Search Platform 3.x осуществляется переносом (копированием) архива с исполняемыми файлами с загрузочного диска в требуемую директорию на жестком диске сервера. После переноса (копирования) архива исполняемые файлы 3i Search Platform 3.x разархивируются из него в требуемую директорию на жестком диске сервера.

Дальнейшие действия по установке выполняются из требуемой директории на жестком диске сервера из папки 3iSearchPlatform:

1) Проверка наличия java:

#java –version

Если java не установлена или имеет версию ниже, чем 1.6, то необходимо установить версию java, входящую в комплект поставки:

#cd /java_jre

#dpkg –i openjdk-7-jre-headless_7u95-2.6.4-0ubuntu0.12.04.1_amd64.deb

Далее необходимо добавить переменные JAVA_HOME и JRE_HOME в переменные среды операционной системы, если их нет, или изменить их, прописав пути к новой установленной версии java.

2) Установка ElasticSearch.

Для установки пользователь должен обладать root правами в операционной системе.

#sudo su

Далее нужно перейти в папку elastic и запустить скрипт установки:

#cd /elastic

#chmod +x elasticsearchInstall.sh

#./elasticsearchInstall.sh

3) Установка ActionDistributor.

Для пользователя ({username}), который будет работать с ActionDistributor, необходимо создать соответствующую папку для его размещения:

#sudo su

#mkdir /home/{username}/bin

Далее, необходимо скопировать каталог с actiondistributor в созданную папку:

cp -avr /actiondistributor /home/{username}/bin

#chown –R {username}:{username} /home/{username}/bin/actiondistributor

После этого нужно отредактировать файл конфигурации, который находится по адресу /home/{username}/bin/actiondistributor/conf/ActionDistributorConf, задав в нем корректный IP-адрес на котором работает ElasticSearch (по умолчанию в файле конфигурации задан 127.0.0.1).

Далее нужно скопировать библиотеки, необходимые для работы ActionDistributor:

#cp –avr /libs/poco /usr/local/lib

#cp –avr /libs/icu /usr/local/lib

После этого необходимо добавить пути к папкам библиотек (/usr/local/lib/poco /usr/local/lib/icu) и запустить команду обновления списка доступных библиотек:

#ldconfig

Далее, нужно скопировать файл сервиса в папку /etc/init.d/ :

#cp –f /servicescript/actiondistributor /etc/init.d/actiondistributor

В файле /etc/init.d/actiondistributor переменной USER_NAME нужно присвоить имя пользователя, под которым будет запускаться ActionDistributor:
USER_NAME={username}

4) Прописать установленные сервисы в автозагрузку операционной системы.
#sudo su

#update-rc.d /etc/init.d/actiondistributor defaults
#usesserv –v /etc/init.d/actiondistributor
#service start actiondistributor

#update-rc.d /etc/init.d/elasticsearch defaults
#usesserv –v /etc/init.d/elasticsearch
#service start elasticsearch

4.2. Конфигурирование компонент 3i Search Platform 3.x

Конфигурирование компонент 3i Search Platform 3.x выполняется пользователем путем внесения изменений в соответствующие файлы конфигураций каждой установленной компоненты. Настройки компонент представлены в Таблице 4.1. Некоторые настройки приложений скрыты от пользователя и осуществляются автоматически.

Таблица 4.1 Настройки компонент 3i Search Platform 3.x

№ п/п	Наименование компоненты	Перечень настроек
1.	ActionDistributor	Задается порт компоненты. Задается и IP-адрес и порт поискового сервера ElasticSearch.

5. ВЫПОЛНЕНИЕ ПРОГРАММЫ

5.1. Описание функционала API

Функции API 3i Search Platform 3.x условно представлены специализированным сервером поиска, обеспечивающим возможность индексации и поиска разнородной текстовой информации.

Далее приведено отдельное описание функций, типов полей, обработок и ошибок API.

5.1.1. Сервер поиска

Функционал сервера поиска 3i Search Platform 3.x базируется на API Elasticsearch и расширяет его за счет предоставления функциональных возможностей в области лингвистической обработки данных (морфологический анализ), в том числе поиска по синонимам. Детальная информация по API Elasticsearch представлена в руководстве пользователя Elasticsearch для версии 1.7 (Elasticsearch Reference 1.7 https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html).

5.1.1.1. Выполнение поисковых запросов

API позволяет выполнять поисковый запрос и получать результат, соответствующий параметрам этого запроса. Запрос может быть сформулирован для выполнения в виде простой строки (URI Search) или представлен телом запроса (Request Body Search).

Поиск с использованием простой строки (URI Search).

URI (англ. Uniform Resource Identifier) – унифицированный идентификатор ресурса. URI – это последовательность символов, идентифицирующая абстрактный или физический ресурс.

Поисковый запрос может быть выполнен с использованием URI через соответствующие параметры запроса. Не все параметры поиска могут быть доступны при использовании этого способа, но URI удобен для выполнения быстрых тестов запросов.

Пример запроса с использованием URI:

```
$ curl -XGET 'http://localhost:9200/twitter/tweet/_search?q=user:kimchy'
```

Пример ответа на запрос:

```
{
  "_shards":{
    "total" : 5,
    "successful" : 5,
    "failed" : 0
  },
  "hits":{
    "total" : 1,
    "hits" : [
      {
        "_index" : "twitter",
        "_type" : "tweet",
        "_id" : "1",
        "_source" : {
          "user" : "kimchy",
          "postDate" : "2009-11-15T14:12:12",
          "message" : "trying out Elasticsearch"
        }
      }
    ]
  }
}
```

Таблица 5.1.1.1 Доступные параметры запроса с использованием URI

№ п/п	Наименование параметра	Описание параметра
1	q	Параметр, обозначающий саму строку запроса.

№ п/п	Наименование параметра	Описание параметра
2	df	Параметр обозначает значение поля по умолчанию, когда никакой префикс не определен в пределах запроса.
3	analyzer	Название анализатора, которое будет использоваться при разборе строки запроса.
4	lowercase_expanded_terms	Параметр для управления возможностью автоматического приведения термов к нижнему регистру. По умолчанию значение true.
5	analyze_wildcard	Параметр управления возможностью анализа запросов по префиксам или отдельным символам. По умолчанию значение false.
6	default_operator	Параметр содержит используемый оператор по умолчанию. Может быть AND (И) или OR (ИЛИ). По умолчанию устанавливается значение OR (ИЛИ).
7	lenient	Параметр для управления возможностью обработки ошибок форматирования (например, указание текста в числовом поле). По умолчанию значение false.
8	explain	Параметр содержит дополнительную информацию о параметрах расчета значений результатов выполненного поискового запроса.
9	_source	Параметр для управления доступностью получения значения поля _source документа. По умолчанию значение true.
10	fields	Параметр для отображения требуемых полей документа в результатах поискового запроса, перечисляемых через запятую. В случае пустого значения не отображает никаких полей документа.
11	sort	Параметр содержит значения требуемого варианта

№ п/п	Наименование параметра	Описание параметра
		сортировки. Может указываться в форме fieldName, orfieldName:asc/ fieldName:desc. fieldName может конкретным полем в документе или специальным именем _score для указания сортировки, выполняемой по числовому параметру.
12	track_scores	Параметр для управления возможностью вывода значения числовой сортировки для каждого результата поискового запроса. Применяется при использовании сортировки, значение по умолчанию false.
13	timeout	Параметр для управления таймаутом поиска. Приостанавливается выполнение запросов в определенный период времени и накапливается пул запросов к исполнению по факту завершения таймаута. По умолчанию значение no timeout.
14	from	Параметр управления начальным значением числа результатов поискового запроса. По умолчанию значение 0.
15	size	Параметр управления числом возвращаемых результатов поискового запроса. По умолчанию значение 10.
16	search_type	Параметр типа выполняемой поисковой операции. Может быть представлен переменными: bedfs_query_then_fetch, dfs_query_and_fetch, query_then_fetch, query_and_fetch, count, scan. По умолчанию значение параметра установлено как переменная query_then_fetch.

Поиск по параметрам в теле запроса (Request Body Search).

В теле запроса, основанном на JSON, параметры поиска формулируются на Query DSL (Unified Queries for Java). Базовые запросы составляются из термов или префиксов. Также используются составные запросы, например булевские (bool) – запросы, включающие другие запросы. У запросов могут также быть фильтры, связанные с ними. Например, constant_score с определенными параметрами.

Пример тела запроса с параметрами поиска:

```
$ curl -XGET 'http://localhost:9200/twitter/tweet/_search' -d '{
   "query" : {
      "term" : { "user" : "kimchy" }
   }
}
'
```

Пример ответа на запрос:

```
{
   "_shards":{
      "total" : 5,
      "successful" : 5,
      "failed" : 0
   },
   "hits":{
      "total" : 1,
      "hits" : [
         {
            "_index" : "twitter",
            "_type" : "tweet",
            "_id" : "1",
            "_source" : {
               "user" : "kimchy",
               "postDate" : "2009-11-15T14:12:12",
               "message" : "trying out Elasticsearch"
            }
         }
      ]
   }
}
```

Таблица 5.1.1.2 Доступные параметры в теле запроса

№ п/п	Наименование параметра	Описание параметра
1	timeout	Параметр для управления таймаутом поиска. Приостанавливается выполнение запросов в определенный период времени и накапливается пул запросов к исполнению по факту завершения таймаута. По умолчанию значение no timeout.
2	from	Параметр управления начальным значением числа результатов поискового запроса. По умолчанию значение 0.
3	size	Параметр управления числом возвращаемых результатов поискового запроса. По умолчанию значение 10.
4	search_type	Параметр типа выполняемой поисковой операции. Может быть представлен переменными: bedfs_query_then_fetch, dfs_query_and_fetch, query_then_fetch, query_and_fetch, count, scan. По умолчанию значение параметра установлено как переменная query_then_fetch.
5	query_cache	Параметр управления кешированием результатов поискового запроса для where ?search_type=count. По умолчанию значение false.

Вышеупомянутые параметры search_type и query_cache должны быть переданы как параметры URI Search. Остальная часть поискового запроса должна быть передана в пределах самого тела запроса. Содержимое тела запроса может также быть передано как REST параметр source.

5.1.1.2. Выполнение агрегаций

Агрегации предоставляют возможность группировки и извлечения статистической информации из данных. API обеспечивает выполнение поиска, возвращающего в одном и том же запросе совпавшие документы и, независимо от них, результаты агрегации. Благодаря этому появляется возможность выполнять запросы со множественной агрегацией и получать результаты обеих операций за один шаг, избегая лишней передачи данных по сети, используя краткий и простой функционал API.

Возможности агрегации данных представлены двумя типами операций:

– Обобщение документов – агрегации, которые формируют группы документов по заданным критериям и ключам. В процессе выполнения агрегации осуществляется проверка каждого документа на совпадение с требуемыми критериями. В случае успешного совпадения документ перемещается в назначенную группу.

– Расчет метрик – агрегации, которые отслеживают и вычисляют метрики по ряду документов.

Общий синтаксис запроса для агрегации:

```
"aggregations" : {
  "<aggregation_name>" : {
    "<aggregation_type>" : {
      <aggregation_body>
    }
    [,"aggregations" : { [<sub_aggregation>]+ } ]?
  }
  [,"<aggregation_name_2>" : { ... } ]*
}
```

Объект «aggregations» в вышеуказанном теле файла JSON содержит выполняемые агрегаты. Каждая агрегация ассоциируется с логическим именем заданным пользователем. Данное уникальное логическое имя может быть

использовано при формировании запроса. Каждая агрегация имеет свой специфический тип <aggregation_type>, определяющий ее содержание <aggregation_body>.

Одновременно на уровне задания типа агрегации может быть определен набор дополнительных вложенных агрегатов. Агрегатов может быть сколь угодно много, и у каждого элемента может быть вложенный элемент без ограничений по глубине.

5.1.1.3. Расширение функционала

Функционал сервера поиска 3i Search Platform 3.x расширяется с помощью плагинов (от. англ. plugin – вставка, сменный блок) для Elasticsearch. Детальная информация по плагинам Elasticsearch представлена в руководстве пользователя Elasticsearch для версии 1.7 (Elasticsearch Reference 1.7 https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-plugins.html).

Используются следующие плагины:

– Elasticsearch-morphology-plugin – расширение функциональных возможностей ElasticSearchCluster в области лингвистической обработки (морфологический анализ).

Плагины устанавливаются с помощью типовых команд и интегрируются посредством стандартных интерфейсов Elasticsearch. Пример типовых команд для подключения плагина морфологии:

```
systemctl stop elasticsearch.service
NODE1=/usr/share/elasticsearch/bin
MORPHOLOGY_PLUGIN_URL=file:///home/admin/Downloads/plugin/elasticsearch-morphology-plugin-1.0-SNAPSHOT.zip
$NODE1/plugin --url $ MORPHOLOGY_PLUGIN_URL --install elasticsearch-morphology-plugin
systemctl start elasticsearch.service
```

где

NODE1 – путь к размещению Elasticsearch на жестком диске;

PLUGIN_URL – путь к файлу плагина на жестком диске;

команда «$NODE1/plugin» – подключение плагина к конкретному экземпляру Elasticrearch на NODE1.

Пример запроса для выполнения морфологического анализа текста:

```
http://10.250.83.242:9200/_analyze?analyzer=russian_dss&text=110 км у Клары украли кораллы
```

где параметр «_analyze» обозначает запрос на выполнение морфологического анализа требуемого текста «text».

Пример результата морфологического анализа текста, где отображается смещение токена, его тип, преобразованный токен:

```
{"tokens":[{"token":"110","start_offset":0,"end_offset":3,"type":"<NUM>","position":1},{"token":"километр","start_offset":4,"end_offset":6,"type":"<ALPHANUM>","position":2},{"token":"у","start_offset":7,"end_offset":8,"type":"<ALPHANUM>","position":3},{"token":"клара","start_offset":9,"end_offset":14,"type":"<ALPHANUM>","position":4},{"token":"украсть","start_offset":15,"end_offset":21,"type":"<ALPHANUM>","position":5},{"token":"коралл","start_offset":22,"end_offset":29,"type":"<ALPHANUM>","position":6}]}
```

Благодаря использованию лингвистической обработки для токена

```
"token":"украсть","start_offset":15,"end_offset":21,"type":"<ALPHANUM>","position":5
```

было значение «украли», а системой распознано «украсть», а не «украл».

5.2. Описание языка запросов

Язык запросов – структурированный набор правил, позволяющий использовать возможности поисковой системы. Описание языка запросов разбито на группы, например, простой поиск и логический поиск. Группы имеют схожее назначение и синтаксис. Запросы, сформулированные на языке запросов, поступают на обработку поисковой системы посредством программного интерфейса. Для пользователя запрос, как правило, создается графическим интерфейсом и не требует запоминания конкретных операторов языка. Далее под запросом будет подразумеваться значение специальной переменной queryText программного интерфейса.

5.2.1. Язык запросов сервера поиска

5.2.1.1. Простой поиск

Выполняется поиск по заданным пользователям ключевым словам. Например, queryText=Путин или queryText=Россия. Поиск производится с учетом всех грамматических форм введенных слов. Таким образом, для запроса *Россия* будут найдены также: *России, Россию, Россией* и т.д. Если в простой форме запроса несколько слов, т.е. *Российская Федерация,* то по умолчанию будет произведен поиск с использованием логического оператора «ИЛИ» (см. п. 5.2.1.2).

5.2.1.2. Логический поиск

При формировании поисковых запросов поддерживаются следующие логические операторы: &&, ||, !. Им соответствуют «И», «ИЛИ», «НЕ». Для создания запросов с применением нескольких операторов поддерживается операция группировки с использованием символов скобок "(" и ")".

Примеры:

1) Россия && Китай

Данному запросу будут соответствовать документы, содержащие все слова в запросе.

2) Нефть || Украина

Данный запрос находит документы, содержащие слова *Россия* и *Украина* как по отдельности, так и оба слова.

3) Путин !Обама

Данный запрос находит документы, содержащие только слово *Путин* без упоминания *Обама*.

4) (Обама || Меркель) && Россия

Данный комбинированный запрос находит документы, содержащие только слова *Обама* и *Россия* или только *Меркель* и *Россия*.

В логическом поиске также поддерживаются булевы операторы:

«+» – требуемый терм обязательно должен присутствовать в результатах;

«-» – требуемый терм обязательно должен отсутствовать в результатах.

5) Санкции && (+Россия -Европа -США)

Данный комбинированный запрос находит документы, содержащие слова *Санкции* и обязательно *Россия*, но исключая слова *Европа* и *США*.

5.2.1.3. Поиск с использованием расстояния

Для поиска с использованием расстояния используется оператор " "~N, где N – целое число (расстояние, меньше либо равно N). Под N подразумевается расстояние между искомыми словами в тексте документа вне зависимости от порядка их следования.

Примеры:

1) "Украина Порошенко"~5

Данному запросу будут соответствовать документы, содержащие слова *Украина* и *Порошенко* в пределах расстояния 5 вне зависимости от порядка их следования.

2) ("Украина Порошенко"~3) && Берлин

Данному запросу будут соответствовать документы, содержащие слова *Украина* и *Порошенко* в пределах расстояния 3 вне зависимости от порядка их следования, а также слово *Берлин* без ограничения на расстояние.

5.2.1.4. Нечеткий поиск

Для нечеткого поиска доступен оператор ~N, где N целое число (редакционное расстояние). По умолчанию N=2 и может не указываться рядом с оператором ~. Данное расстояние используется, чтобы найти все термы с максимумом двумя изменениями (вставка, замена, удаление, перемещение двух смежных символов). Указание расстояния N=1 позволяет обработать около 80% различных человеческих опечаток в одном символе искомого терма.

Примеры:

1) эксперт~1

Данному запросу будут соответствовать документы, содержащие слово *Эксперт* с разными его вариациями, включая потенциальные «ошибки» в одном символе, например, *Экспорт*.

2) образовать~

Данному запросу будут соответствовать документы, содержащие слово *Образовать* с разными его вариациями, включая возможные два изменения символов, например, *Обжаловать, Обрисовать*.

5.2.1.5. Поиск с использованием специальных символов

Допустимы следующие символы: “**?”, “*”**. Необходимо иметь ввиду, что при использовании “*” и “?” в множество, которое будет получено в результате обхода индексированных термов, попадет определенное количество термов, указанное в конфигурации, в противном случае поиск может быть бесконечным.

“**?”** – поиск термов с подстановкой одного любого символа.

“*” – поиск термов с подстановкой любого количества символов. Оператор ***** без префикса находит все документы.

Примеры:

1) SS*

Данному запросу будут соответствовать документы, содержащие любые слова, начинающиеся с SS*: SSJ-100, SSAB, SS-18.

2) ?олото

Данному запросу будут соответствовать документы, содержащие любые слова, состоящие из любого первого символа и заканчивающиеся на «олото»: золото, болото, долото и др.

3) золот* && побер*

Данному запросу будут соответствовать документы, содержащие любые слова, начинающиеся с «золот» и «побер», т.е. каждый найденный документ должен содержать хотя бы одно слово из каждой группы.

Необходимо отметить, что для использования зарезервированных знаков, которые функционируют как операторы (+ - = && || > < ! ( ) { } [ ] ^ " ~ * ? : \ /), нужно использовать обратную косую черту. Например, для поиска выражения (1+1) =2, запрос должен выглядеть как \(1 \+ 1 \) \= 2.

6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Входными данными являются структурированные данные в формате JSON.

Выходными данными являются результаты выполнения заданий над поисковым индексом (совокупность релевантных документов, отфильтрованное множество документов и др.). В качестве основного формата представления выходных данных также используется JSON (англ. JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript (происходит от подмножества языка стандарта ECMA-262 1999 года). Формат считается независимым от языка и может использоваться практически с любым языком программирования.

JSON-текст представляет собой (в закодированном виде) одну из двух структур:

– Набор пар ключ: значение. В различных языках это реализовано как объект, запись, структура, словарь, хэш-таблица, список с ключом или ассоциативный массив. Ключом может быть только строка, значением — любая форма.

– Упорядоченный набор значений. Во многих языках это реализовано как массив, вектор, список или последовательность.

В качестве значений в JSON используются структуры:

– Объект – это неупорядоченное множество пар ключ: значение, заключённое в фигурные скобки «{ }». Ключ описывается строкой, между ним и значением стоит символ «:». Пары ключ-значение отделяются друг от друга запятыми.

– Массив (одномерный) – это упорядоченное множество значений. Массив заключается в квадратные скобки «[ ]». Значения разделяются запятыми.

– Значение может быть строкой в двойных кавычках, числом, объектом, массивом, одним из литералов: true, false или null. Таким образом структуры могут быть вложены друг в друга.

– Строка – это упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки. Символы могут быть указаны с использованием escape-последовательностей, начинающихся с обратной косой черты «\».

7. СООБЩЕНИЯ ОПЕРАТОРУ

Обращение к REST API происходит посредством HTTP-запросов. Тело запросов и ответов представлено в JSON формате. Ошибки выполнения функций предоставляются оператору в виде соответствующих кодов состояния HTTP – (англ. HTTP status code) – часть первой строки ответа сервера при запросах по протоколу HTTP. Он представляет собой целое число из трёх десятичных цифр. Первая цифра указывает на класс состояния. За кодом ответа обычно следует отделённая пробелом поясняющая фраза на английском языке, которая разъясняет человеку причину именно такого ответа (см. Таблицу 7.1).

Таблица 7.1 Коды состояния HTTP, используемые для обозначения ошибок

Код	**Описание**	**Рекомендуемые действия для устранения ошибки**
400	«Bad Request». Сервер обнаружил в запросе клиента синтаксическую ошибку. Относится к группе ошибок со стороны клиента.	Проверить содержание требуемого запроса на наличие синтаксических ошибок и внести соответствующие правки.
404	«Not Found». Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL. Относится к группе ошибок со стороны клиента.	Проверить содержание требуемого запроса на правильность запрашиваемого адреса URL и внести соответствующие правки.
500	«Internal Server Error». Внутренняя ошибка сервера.	Требуется детальная диагностика системы (анализ логов, инспекция

Код	Описание	Рекомендуемые действия для устранения ошибки
	Относится к группе ошибок со стороны сервера.	данных и др.).
503	«Service Unavailable». Сервер временно не имеет возможности обрабатывать запросы по техническим причинам (обслуживание, перегрузка и прочее). Относится к группе ошибок со стороны сервера.	Рекомендуется повторить запрос через некоторое время.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

В настоящем документе приняты следующие условные обозначения:

СПО	Специальное программное обеспечение
СУБД	Система управления базами данных
3i Search Platform 3.x	СПО «3i Search Platform 3.x»
API	сокр. англ. Application Programming Interface, интерфейс программирования приложений – набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением для использования во внешних программных продуктах
IP-адрес	сокр. англ. Internet Protocol Address — уникальный сетевой адрес узла в компьютерной сети, построенной по протоколу IP
JSON	сокр. англ. JavaScript Object Notation – текстовый формат обмена данными, основанный на JavaScript (происходит от подмножества языка стандарта ECMA-262 1999 года)
HTTP	сокр. англ. HyperText Transfer Protocol, протокол передачи гипертекста — протокол прикладного уровня передачи данных на технологии «клиент-сервер»
REST API	сокр. англ. Representational State Transfer, передача репрезентативного состояния – метод взаимодействия компонентов распределённого приложения в сети Интернет, при котором вызов удаленной процедуры представляет собой обычный HTTP-запрос

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	№ документа	Входящий № сопроводительного документа и дата	Подпись	Дата
	Измененных	замененных	новых	аннулированных					