[1. 引言 2](#_Toc490074331)

[1.1 编写目的 2](#_Toc490074332)

[1.2 系统描述 2](#_Toc490074333)

[1.3版本功能简述 2](#_Toc490074334)

[3. 术语 4](#_Toc490074335)

[4. 用户需求定义 6](#_Toc490074336)

[5. 系统体系结构 7](#_Toc490074337)

[6. 系统需求描述 11](#_Toc490074338)

[7. 系统模型 12](#_Toc490074339)

[8. 系统进化 13](#_Toc490074340)

[9. 附录 14](#_Toc490074341)

[10. 界面设计 15](#_Toc490074342)

[10.1查词功能页面 15](#_Toc490074343)

[10.2翻译功能页面 16](#_Toc490074344)

[10.3单词本功能页面 16](#_Toc490074345)

[10.4文章功能页面 17](#_Toc490074346)

[10.5个人设置功能页面 17](#_Toc490074347)

# 引言

## 编写目的

本需求说明书描述项目需求细节和实现方式，供开发人员、测试人员和系统用户使用。

## 系统描述

本系统为日文

常见文档信息搜索方式包括：网络搜索、企业搜索和P2P搜索。一个具体的应用场景为：用户向搜索引擎输入一个包含目标关键字的搜索请求并由后者以表单形式返回一个有序的文档列表。不同于数据库中的结构化查询[[1]](#footnote-1)，文档内容通常没有固定结构，在其上执行查找作业时需要按字符比较方式从中提取与关键字匹配的信息片断。文本索引能够提高文档信息搜索效率，被广泛运用于商业、学术和生物医疗领域。例如：谷歌[[2]](#endnote-1)和百度[[3]](#endnote-2)等常用搜索引擎以及BWA和SOAP等经典生物基因对齐软件分别采用倒排索引和后缀索引来提高搜索效率。

1. 倒排索引和后缀索引是全文索引[[4]](#footnote-2)，两者均能够在O(nlogn+occ)的时间内返回指定关键字在文档中的所有出现。本项目采用全文索引构建一个用于日文翻译的垂直型信息检索系统，对给定的输入关键字、词、句，返回指定上下文中（词库、句库、原文、译文）的相关内容（解释、示例）。

## 1.3版本功能简述

# 术语

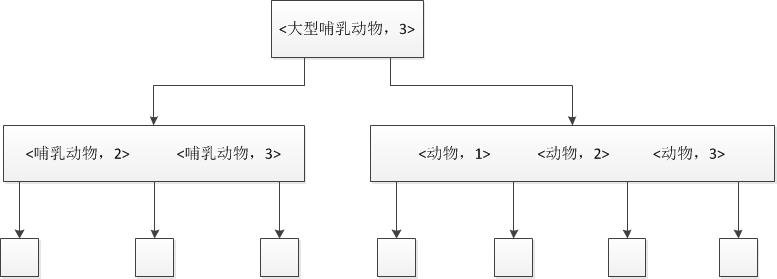
定义文档中的技术术语和词汇 。假设文档读者是不具有专业知识和经验的人。

倒排索引可将一个原子搜索项（例如：单词）映射到一个包含该项的被索引单元集合（例如：文件集）。可使用B树构建倒排索引，树中节点包含<word, location>并以此二段优先权排序。下面以表1数据为输入，构造并查找关键字“动物”：

|  |  |
| --- | --- |
| 表1 | |
| 文件集 | {这是一只动物、这是一只哺乳动物、这是一只大型哺乳动物} |
| 词袋 | {动物、哺乳动物、大型哺乳动物} |

索引构造过程包括以下3步：

1. 生成元组。组合无序元组序列，得到<动物，1>，<动物，2>，<动物，3>，<哺乳动物，2>，<哺乳动物，3>，<大型哺乳动物，3>。
2. 排序元组。依次按字典序和整数序对各元组的首字段和次字段排序，得到有序元组序列：<哺乳动物，2>，<哺乳动物，3>，<大型哺乳动物，3>，<动物，1>，<动物，2>，<动物，3>。
3. 构建B树。构建3阶B树如下。



索引查找过程包括以下2步：

1. 判定存在。从B树根节点出发，访问编号为1的节点，得到所有包含“动物”的三个元组。
2. 获取位置。从步骤1中查找得到的元组中取出包含关键字的文件编号，使用字符串匹配算法（例如：KMP）在指定文件中查找“动物”的出现。

# 用户需求定义

用例1：

用例2：

# 系统体系结构

这一部分要对待建系统给出体系结构框架，该体系结构要给出功能在各个模块中的分布。能被复用的结构中组件要用醒目方式示意出来。

应用范围划分：网络、企业、垂直、桌面、

词频，话题相关度，用户相关度

评估：查全率，查对率；响应时间，查询吞吐量和索引速度，新鲜度（合并、更新？）

搜索引擎体系结构：体系结构的设计是用于确保系统能够满足应用的需求或目标。一个搜索引擎的两个主要目标是：（1）高效（质量）。根据查询请求从文件集合中找出与之相关度最高的子集。（2）效率（速度）。尽快处理用户请求。

搜索引擎的部件包含两个主要功能模块： （1）索引进程和查询进程。

索引进程构造用于搜索的数据结构，查询进程使用构造的结构和用户的查询输入来生成排序的文件集合（结果）。

索引进程的构件：文本获取、文本转换和索引创建。

1. 文本获取：例如爬去网络信息。该阶段创建一个文本数据库，其中包含所有文档的元数据（文件类型、结构和其他特征）和文本（文档内容）。
2. 文本转换：将文件转换为索引项或特征。索引项是从文本中取出的将被索引的内容，其用于后续的查找。最简单的索引项是词。特征则是文本文件中的一段数据，其可用于表现该文件的内容。其他的索引项或特征包括：成语，人名，日期和网络连接。索引词汇表是定义在一个文件集上的包含其中所有索引项的集合。
3. 索引创建：将文本转换后的输出作为输入，构建用于快速搜索的索引和数据结构。倒排索引和倒排文件是搜索引擎最常用的索引结构。一个倒排索引为每个索引项维持一个列表，列表中的每个文件包含该索引项。

查询进程的构件：用户交互，排序和评估

（1） 用户交互：提供用户与搜索引擎之间的接口。一个任务是接收用户的查询请求输入并将之转换为索引项。另一个任务是将搜索引擎返回的排序好的查询结构向用户显示。这包括：生成查询结果的片段向用户显示。

（2） 排序：搜索引擎的核心。将用户交互接口转换得到的索引项作为输入，基于一个信息获取模型来排序索引查找结果。注重高效性和效率。

（3） 评估：测量和监视效率和有效性。一个重要部分是记录用户的查询行为并根据日志分析其行为。评估结构被用于优化排序的结果。评估往往是线下行为。

文本获取构件：

1. 爬虫（crawler）：爬虫部件的主要功能是为搜索引擎区分和获取文档。
2. 信息提供源(feeds)：例如：新闻RSS。搜索引擎只需要监控该输入流的更新状况。
3. 转换：将爬虫和信息提供源所提供的非纯文本（HTML，XML，PDF等）转换为格式一致的文本并为其生成元数据。
4. 文档数据存储：用于管理大量文档和关联的有结构数据的数据库。文档的内容通常以要格式存储，有结构数据包括元数据和其他从文档中提取的信息（包括链接和锚文本）。

文本转换构件：

1. 分词（paser）：用于处理文本中的令牌序列，识别结构化元素（例如：题目，图片，链接和标题）。分词是重要的第一步。文件和查询文本需要被转换为令牌序列，以便比较。
2. 停止词过滤（stopping）：从成为索引项的令牌流中移除公共词汇。例如：“the, of, to, for”，因为这些词汇使用频度很高，将他们移除后能够大幅减少索引的规模。
3. 词干提取（stemming）：将从一个公共词根派生得到的词汇归纳为一组。
4. 链接抽取和分析（link extraction and analysis）：从文本中抽取链接，这些链接的相关信息通畅存储在数据库中另行索引。
5. 信息抽取（information extraction）：用于识别比单词更为复杂的索引项。这通常需要句法分析器或者词性标签的支持。
6. 分类器：分类技术将预定义的类标签赋予文档。这些标签通常表示文档的话题范畴，例如：“体育，政治或商业”。另外两个重要的分类类型是：垃圾内容和广告等非内容部分的分类。

索引创建构件：

1. 文本统计：收集关于词、特征和文档的统计信息。这些信息被排名部件用来对文档打分。
2. 赋权重：索引项的权反应其在文本中的相对重要性，并被用于排名打分。在老式获取模型中常用的一个权重函数是tf-idf（term frequency & inverse document frequency）
3. 倒置：该部件的任务时将文本转换部件生成的文件-索引项信息流转换为索引项-文件信息流，用来创建倒排索引。
4. 索引分布：将索引分布到多台计算机和可能的位于网络中的多个站点上。（根据文档划分索引存储，根据索引项划分索引存储或者将多个索引副本分布存储）

用户交互

1. 查询输入：提供一个接口和一个用于解析查询语言的分析器。最简单的查询语言定义了一组操作符，每个操作符用于指出查询输入中需以特定方式处理的文本。
2. 查询转换：改进用户输入的初始查询语句。最简单的技术包括用于文本转换的分词、停止词、词干提取等等。拼写检查和查询建议是查询转换技术，其输出较为类似。两种技术都向用户提供一组用于替换初始查询语句的提示语句，它们通常利用从互联网取得的历史查询日志来生成提示语句。查询扩展技术建议或添加额外的索引项至查询请求，但通常是基于索引项在文本中的出现频率。
3. 结果输出：结果输出部件负责展示排名部件输出的已排序文件。主要任务包括：生成各文件的描述摘要，高光重要词汇和短语，根据相关度聚类已排序文件，将适当的建议添加到查询结果显示中。

排名构件

1. 打分：基于检索模型，使用排名算法为各文件打分。
2. 性能优化：包括优化排名算法和相关索引，减少响应时间和增加查询吞吐量。
3. 分布：配合索引的分布存储方式，将排名构件的功能分布化。

评估构件

日志记录：使用用户的查询日志和他们与搜索引擎的交互信息来优化查询效率和效能。

排名分析：日志数据或者对大量（查询，文件）对的显式相关度判断可以用于对排名算法的效能进行比较分析。

性能分析：响应时间、吞吐量

# 系统需求描述

这一部分要对功能和非功能需求进行详细描述。如有必要，对非功能需求要再进一步描述，例如，定义与其他系统间的接口。

# 系统模型

这一部分要提出一个或多个系统模型，以表达系统组件、系统以及系统环境之间的关系。这些模型可以是对象模型、数据流模型和语义数据模型。

# 系统进化

这一部分要描述系统基于的基本设想和定位以及硬件和用户需求改变时所要做的改变。这部分对系统设计人员来说是有用的，因为这有助于他们避免一些设计决策，这些决策可能会限制未来系统的变更。

# 附录

这一部分要提供与开发的应用有关的详细、专门的信息。该附录的例子是硬件和数据库的描述。硬件需求定义了系统最小和最优配置，数据库需求定义了系统所用的数据的逻辑结构和数据之间的关系。

# APP程序设计

|  |  |
| --- | --- |
| SearchActivity | 查词功能界面 |
| TranslateActivity | 翻译功能界面 |
| WordActivity | 单词本功能界面 |
| ArticleActivity | 文章功能界面 |
| SettingActivity | 设定功能界面 |

# 界面设计

## 10.1查词功能页面

|  |  |  |
| --- | --- | --- |
| IMG_0081.PNG | IMG_0086.PNG | 查词-返回界面.PNG |
| 初始 | 查询 | 返回（精确匹配） |
| （必）输入查找内容  （必）今日热词 | （必）建议查询 | （必）释义  （必）句例 |
|  |  |  |
| 查词-返回界面（模糊匹配）.PNG |  |  |
| 返回（模糊匹配） |  |  |
| （必）网络释义  （选）联想词汇 |  |  |

## 10.2翻译功能页面

|  |  |  |
| --- | --- | --- |
| 翻译-返回.PNG | 翻译-返回.PNG | 翻译-查找.PNG |
| 初始（复用） | 查找（上半页面） | 返回（下半页面） |
|  | （必）输入查找内容 | （必）网络释义 |

## 10.3单词本功能页面

|  |  |  |
| --- | --- | --- |
| 词句本.PNG | 词句本.PNG |  |
| 收缩式词列表界面 | 收缩式句列表界面 |  |
| （必）词列表展示 | （必）句列表展示 |  |

## 10.4文章功能页面

|  |  |  |
| --- | --- | --- |
| 词句本.PNG | 词句本.PNG | 词句本.PNG |
| 文章源列表界面 | 文章列表界面 | 文章界面 |
| （必）文章源列表展示 | （必）文章列表展示 | （必）文章展示 |

## 10.5个人设置功能页面

|  |  |  |
| --- | --- | --- |
| 个人设置-初始.PNG |  |  |
| 初始界面 |  |  |
| （必）获取网络释义选项  （必）离线文库下载选项  （必）关于 |  |  |

1. 不包含数据库中支持的文本属性的精确/模糊查询。 [↑](#footnote-ref-1)
2. [] google search engine, http://www.google.com. [↑](#endnote-ref-1)
3. [] baidu search engine, https://www.baidu.com. [↑](#endnote-ref-2)
4. 严格地说，倒排索引不是全文索引。其假设输入关键字来自一个预定义词袋并对词袋做预处理。 [↑](#footnote-ref-2)