

VISVESVARAYA TECHNOLOGICAL UNIVERSITY  
JNANASANGAMA, BELAGAVI - 590018



# “SOCIRANK: IDENTIFYING AND RANKING PREVALENT NEWTOPICS USING SOCIAL MEDIA FACTORS”

Thesis submitted in partial fulfillment of the curriculum prescribed for  
the award of the degree of Bachelor of Engineering in  
Computer Science & Engineering by

1CR14CS098 Pramod Singh  
1CR14CS175 Keerthana J  
1CR14CS093 Prabhat Sharma  
1CR13CS086 Sadgayanam

Under the Guidance of

Ms. Navaneetha  
Assistant Professor  
Department of CSE, CMRIT, Bengaluru



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
#132, AECS LAYOUT, IT PARK ROAD, BENGALURU - 560037

2017-18

VISVESVARAYA TECHNOLOGICAL UNIVERSITY  
JNANASANGAMA, BELAGAVI - 590018



## Certificate

This is to certify that the project entitled “**SOCIRANK-IDENTIFYING AND RANKING PREVALENT NEWS TOPICS USING SOCIAL MEDIA FACTORS**” is a bonafide work carried out by Pramod Singh bearing USN : 1CR14CS098 ,Keerthana J bearing USN: 1CR14CS175, Prabhat Sharma bearing USN: 1CR14CS093 and Sadgayanam bearing USN : 1CR13CS086 in partial fulfillment of the award of the degree of Bachelor of Engineering in Computer Science & Engineering of Visvesvaraya Technological University, Belgaum, during the year 2017-18. It is certified that all corrections / suggestions indicated during reviews have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

-----  
Signature of Guide  
**Ms. Navaneetha**  
Assistant Professor  
Department of CSE  
CMRIT, Bengaluru - 37

-----  
Signature of HoD  
**Dr. Jhansi Rani P**  
Professor & Head  
Department of CSE  
CMRIT, Bengaluru - 37

-----  
Signature of Principal  
**Dr. Sanjay Jain**  
Principal  
CMRIT,  
Bengaluru - 37

### External Viva

Name of the Examiners	Institution	Signature with Date
1. -----	-----	-----
2. -----	-----	-----

# Acknowledgement

We take this opportunity to thank all of those who have generously helped us to give a proper shape to our work and complete our BE project successfully. A successful project is fruitful culmination efforts by many people, some directly involved and some others indirectly, by providing support and encouragement.

We would like to thank **Dr. SANJAY JAIN** , Principal , CMRIT , for providing excellent academic environment in the college.

We would like to express our gratitude towards **Dr. JHANSI RANI P** , Professor & HOD , Dept of CSE , CMRIT , who provided guidance and gave valuable suggestions regarding the project.

We consider it a privilege and honour to express our sincere gratitude to our Internal Guide **Ms. NAVANEETHA** , Asst. Professor , Department of Computer Science & Engineering , CMRIT , for her valuable guidance throughout the tenure of this project work.

Pramod Singh  
Keerthana J  
Prabhat Sharma  
Sadgayanam

# Table of Contents

Table of Contents	ii
List of Figures	iv
List of Tables	v
Abstract	vi
<b>1 PREAMBLE</b>	<b>1</b>
1.1 INTRODUCTION . . . . .	2
1.2 PROBLEM STATEMENT . . . . .	4
<b>2 LITERATURE SURVEY</b>	<b>5</b>
2.1 INTRODUCTION . . . . .	6
2.2 LITERATURE SURVEY . . . . .	6
2.3 PAPER 1 . . . . .	7
2.4 PAPER 2 . . . . .	7
2.5 PAPER 3 . . . . .	8
2.6 PAPER 4 . . . . .	8
<b>3 THEORETICAL BACKGROUND</b>	<b>9</b>
3.1 INTRODUCTION . . . . .	10
3.2 TOPIC IDENTIFICATION . . . . .	10
3.3 TOPIC RANKING . . . . .	10
3.4 SOCIAL NETWORK ANALYSIS . . . . .	10
3.5 KEYWORD EXTRACTION . . . . .	11
3.6 CO-OCCURRENCE SIMILARITY . . . . .	12
3.7 STANFORD CoreNLP-NATURAL LANGUAGE SOFTWARE . . . . .	12
<b>4 SYSTEM REQUIREMENT SPECIFICATION</b>	<b>13</b>
4.1 INTRODUCTION . . . . .	14
4.2 FUNCTIONAL REQUIREMENTS . . . . .	15

4.3	NON-FUNCTIONAL REQUIREMENTS . . . . .	15
4.4	HARDWARE REQUIREMENTS . . . . .	18
4.5	SOFTWARE REQUIREMENTS . . . . .	18
4.6	SOFTWARE QUALITY ATTRIBUTES . . . . .	19
<b>5</b>	<b>SYSTEM ANALYSIS</b>	<b>20</b>
5.1	INTRODUCTION . . . . .	21
5.2	FEASIBILITY STUDY . . . . .	21
5.3	OPERATIONAL FEASIBILITY . . . . .	21
5.4	ECONOMICAL FEASIBILITY . . . . .	22
5.5	TECHNICAL FEASIBILITY . . . . .	22
5.6	SOCIAL FEASIBILITY . . . . .	22
<b>6</b>	<b>SYSTEM DESIGN</b>	<b>23</b>
6.1	INTRODUCTION . . . . .	24
6.2	SYSTEM DEVELOPMENT METHODOLOGY . . . . .	24
6.3	DESIGN USING UML . . . . .	26
6.4	DATA FLOW DIAGRAM . . . . .	26
6.5	CLASS DIAGRAM . . . . .	27
6.6	USE CASE DIAGRAM . . . . .	28
6.7	ACTIVITY DIAGRAM . . . . .	29
6.8	SEQUENCE DIAGRAM . . . . .	30
<b>7</b>	<b>IMPLEMENTATION</b>	<b>31</b>
7.1	INTRODUCTION . . . . .	32
7.2	SOCIRANK CODE . . . . .	32
7.3	CONNECTING ANGULAR AND SPRING . . . . .	36
7.4	CONNECTING SPRING AND MYSQL . . . . .	37
<b>8</b>	<b>TESTING AND RESULTS</b>	<b>39</b>
8.1	TESTING . . . . .	40
8.2	VALIDATION AND SYSTEM TESTING . . . . .	40
8.3	RESULTS . . . . .	43
<b>9</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>45</b>
9.1	CONCLUSION . . . . .	46
9.2	FUTURE SCOPE . . . . .	46
	<b>References</b>	<b>47</b>

# List of Figures

4.1	Software Quality Attributes . . . . .	19
6.1	Waterfall Model . . . . .	26
6.2	Class Diagram of Components . . . . .	28
6.3	Use Case Diagram . . . . .	28
6.4	Activity Diagram . . . . .	29
6.5	Sequence Diagram . . . . .	30
8.1	search by location . . . . .	43
8.2	news feeds . . . . .	43
8.3	twitter data . . . . .	44
8.4	success screen . . . . .	44
8.5	socirank . . . . .	44

# List of Tables

6.1 Symbols used in UML . . . . .	27
-----------------------------------	----

# Abstract

Mass media sources, specifically the news media, have traditionally informed us of daily events. In modern times, social media services such as Twitter provide an enormous amount of user-generated data, which have great potential to contain informative news-related content. For these resources to be useful, we must find a way to filter noise and only capture the content that, based on its similarity to the news media, is considered valuable. However, even after noise is removed, information overload may still exist in the remaining data; hence, it is convenient to prioritize it for consumption. To achieve prioritization, information must be ranked in order of estimated importance considering three factors. First, the temporal prevalence of a particular topic in the news media is a factor of importance, and can be considered the media focus (MF) of a topic. Second, the temporal prevalence of the topic in social media indicates its user attention (UA). Last, the interaction between the social media users who mention this topic indicates the strength of the community discussing it, and can be regarded as the user interaction (UI) toward the topic. We propose an unsupervised framework, SociRank, which identifies news topics prevalent in both social media and the news media, and then ranks them by relevance using their degrees of MF, UA, and UI. Our experiments show that SociRank improves the quality and variety of automatically identified news topics.



# Chapter 1

## PREAMBLE

## 1.1 INTRODUCTION

Today, online social media such as Twitter have served as tools for organizing and tracking social events. Understanding the triggers and shifts in opinion driven mass social media data can provide useful insights for various applications in academia, industry, and however, there remains a general lack of finding of what causes the hot spots in social media. Typically, the reasons behind the rapid spread of information can be summarized in terms of two categories: exogenous and endogenous factors. Growing factors are the results of information diffusion inside the social network itself, namely, users obtain information primarily from their online social network. In contrast, exogenous factors mean that users get information from outside sources first, for example, traditional news media, and then bring it into their social network.

Although previous works have explored both the social media and external news data datasets, few researchers have looked at the endogenous and exogenous factors based on semantical or topical knowledge. They have either sought to identify relevant tweets based on news articles or simply correlated the two data sources through similar patterns in the changing data volume. Still within the same data source, there could be various factors that drive the evolution of information over time. Exogenous factors across multiple datasets make analyzing the evolution and relationship among multiple data streams more difficult. Watching social media and outside news data streams in a united frame can be a practical way of solving this problem.

In this paper, we propose a novel topic model, News and Twitter Interaction Topic model (NTIT), that jointly learns social media topics and news topics and subtly capture the influences between topics. The intuition behind this approach is that before a user posts a message, he/she may be influenced either by opinions from his/her online friends or by articles from news agencies. In our new framework, a word in a tweet can be responsive to the topical influences coming either from endogenous factors (tweets) or from exogenous factors (news). A straightforward approach for identifying topics from different social and news media sources is the application of topic modeling. Many methods have been proposed in this area, such as latent Dirichlet allocation (LDA) and probabilistic latent semantic analysis (PLSA). Topic modeling is, in essence, the discovery of topics in text corpora by clustering together frequently co-occurring words. This approach, however, misses out in the temporal component of prevalent topic detection, that is, it does not take into account how topics change with time. Furthermore, topic modeling and other topic detection techniques do not rank topics according to their popularity by taking into account their prevalence in both news media and social media. We introduce unsupervised system SociRank which

---

effectively identifies news topics that are prevalent in both social media and the news media, and then ranks them by relevance using their degrees of MF, UA and UI.

Even though this paper focuses on news topics, it can be easily adapted to a wide variety of fields, from science and technology to culture and sports. To the best of our knowledge, no other work attempts to employ the use of either the social media interests of users or their social relationships to aid in the ranking of topics. Moreover, SociRank undergoes an empirical framework, comprising and integrating several techniques, such as keyword extraction, measures of similarity, graph clustering, and social network analysis. The effectiveness of our system is validated by extensive controlled and uncontrolled experiments.

## 1.2 PROBLEM STATEMENT

- social media news describes only the interest of the audience and may thus provides insight into their popularity.
- It is hard to find a way to filter news from noisy.
- It requires high computation to prioritize the news.
- Over 60% news in web doesn't provide important news that viewer should be aware of and which is happening in their respective locations.

## **Chapter 2**

# **LITERATURE SURVEY**

## 2.1 INTRODUCTION

Literature survey is mainly carried out in order to analyze the background of the current project which helps to find out flaws in the existing system and guides on which unsolved problems we can work out. So, the following topics not only illustrate the background of the project but also uncover the problems and flaws which motivated to propose solutions and work on this project.

## 2.2 LITERATURE SURVEY

Literature survey is the documentation of a comprehensive review of the published and unpublished work from secondary sources data in the areas of specific interest to the researcher. The library is a rich storage base for secondary data and researchers used to spend several weeks and sometimes months going through books, journals, newspapers, magazines, conference proceedings, doctoral dissertations, master's theses, government publications and financial reports to find information on their research topic. Reviewing the literature on the topic area at this time helps the researcher to focus further interviews more meaningfully on certain aspects found to be important is the published studies even if these had not surfaced during the earlier questioning. So the literature survey is important for gathering the secondary data for the research which might be proved very helpful in the research. The literature survey can be conducted for several reasons. The literature review can be in any area of the business.

## 2.3 PAPER 1

**Title:** Towards a Followee Recommender System for Information Seeking Users in Twitter

**Context:**

Micro-blogging activity taking place in sites such as Twitter gains everyday more importance as a source of real-time information and news spreading medium. Finding relevant information sources among the increasing number of Twitter members is essential for users needing to cope with real-time information. In this paper we study Twitter aiming at generating a set of recommendations to a target user consisting in people who publish tweets that might be interesting to him/her. We evaluate and compare two recommendation approaches: the first selects a set of candidate recommendations using only the network topology and the second exploits the user-generated content available in their tweets. We report the results of a set of controlled experiments with real users carried out to evaluate and compare the performance of both algorithms.

## 2.4 PAPER 2

**Title:** Comparing Twitter and Traditional Media using Topic Models

**Context:**

Twitter as a new form of social media can potentially contain much useful information, but content analysis on Twitter has not been well studied. In particular, it is not clear whether as an information source Twitter can be simply regarded as a faster news feed that covers mostly the same information as traditional news media. In This paper we empirically compare the content of Twitter with a traditional news medium, New York Times, using unsupervised topic modeling. We use a Twitter-LDA model to discover topics from a representative sample of the entire Twitter. We then use text mining techniques to compare these Twitter topics with topics from New York Times, taking into consideration topic categories and types. We also study the relation between the proportions of opinionated tweets and retweets and topic categories and types. Our comparisons show interesting and useful findings for downstream IR or DM applications

## 2.5 PAPER 3

**Title:** : A Speech-Based Just-In-Time Retrieval System

**Context:**

This paper addresses the problem of key-word extraction from conversations, with the objective of utilizing those watchwords to get better, for each quick discussion piece, a bit number of conceivably pertinent reviews, which may be prescribed to members. Anyways, even a brief piece contains a mixed bag of phrases, using a programmed discourse acknowledgment (ASR) framework offers slips amongst them. along those lines, it is hard to surmise efficaciously the statistics wishes of the discussion members. We first recommend a calculation to take away decisive phrases from the yield of an ASR framework, which makes usage of topic demonstrating techniques and of a sub modular prize capability which supports differing qualities inside the magic phrase set, to coordinate the potential differing characteristics of topics and reduce ASR commotion. At that point, we recommend a method to deduce numerous topically remote inquiries from this decisive word set, preserving in mind the stop goal to expand the possibilities of creating at any fee one pertinent proposal while making use of those inquiries to searching for over the English.

## 2.6 PAPER 4

**Title:** : Language- and Domain Independent Automatic Indexing Terms for Abstracting

**Context:**

A method of drawing index terms from text is presented. The approach uses no stop list, stemmer, or other language- and domain-specific component, allowing operation in any language or domain with only trivial modification. The method uses n-gram counts, achieving a function similar to, but more general than, a stemmer. The generated index terms, which the author calls highlights, are suitable for identifying the topic for perusal and selection. An extension is also described and demonstrated which selects index terms to represent a subset of documents, distinguishing them from the corpus. Some experimental results are presented, showing operation in English, Spanish, German, Georgian, Russian, and Japanese.



## Chapter 3

# THEORETICAL BACKGROUND

## 3.1 INTRODUCTION

## 3.2 TOPIC IDENTIFICATION

Much research has been carried out in the field of topic identification referred to more formally as topic modeling. Two traditional methods for detecting topics are LDA and PLSA. LDA is a generative probabilistic model that can be applied to different tasks, including topic identification. PLSA, similarly, is a statistical technique, which can also be applied to topic modeling. In these approaches, however, temporal information is lost, which is paramount in identifying prevalent topics and is an important characteristic of social media data. Furthermore, LDA and PLSA only discover topics from text corpora; they do not rank based on popularity or prevalence.

## 3.3 TOPIC RANKING

There are several means by which this task can be accomplished, traditionally being done by estimating how frequently and recently a topic has been reported by mass media. We made use of Twitter to discover news-related content that might be considered important. There are many ways by which ranking can be achieved:

- By estimating the amount of times they read stories related to that particular topic.
- By utilizing a clustering approach for tweet mining.
- Ranking based on the co-occurrence of popular terms within the user's RSS and Twitter feeds.

All of these systems aim to identify emerging topics, but give no insight into their popularity over time. Nevertheless, these works provide us with a basis for extending the premise of User Attention. Their work, however, does not integrate or collaborate with other data sources, as accomplished by SociRank.

## 3.4 SOCIAL NETWORK ANALYSIS

website usage statistics provide initial proof of attention, additional data are needed to corroborate it. We employ the use of social media, specifically Twitter, as a means to estimate UA. When a user tweets about a particular topic, it signifies that user is interested in the topic and it has captured her attention than visiting a website.

In summary, visiting a website might be the initial stimulus, but taking the additional step of discussing a topic via social media signifies genuine attention. Additionally, we believe that the relationship between social media users who discuss the same topics also plays a key role in topic relevance. Kwan et al. proposed a measure referred to as reciprocity, which attempts to detect the interaction between social media users and perceive their engagement in relation to a particular topic. Higher reciprocity means greater interaction between users, and thus topics with higher reciprocity should be considered more important because of their underlying community structure. We can inherently identify the power and influence of a well-structured community as opposed to a decentralized and unstructured one. Our method applies this logic to support the idea that higher reciprocity signifies greater importance.

### 3.5 KEYWORD EXTRACTION

Concerning the field of keyword or informative term extraction, many unsupervised and supervised methods have been proposed. Unsupervised methods for keyword extraction rely solely on implicit information found in individual texts or in a text corpus. Supervised methods, on the other hand, make use of training datasets that have already been classified. Among the unsupervised methods, there are those that employ statistical measures of term informativeness or relevance, such as term specificity, TFIDF, word frequency, n-grams, and word co-occurrence. Other unsupervised approaches are graph-based, where a text is converted into a graph whose nodes represent text units (e.g., words, phrases and sentences) and whose edges represent the relationships between these units. The graph is then recursively iterated and relevance scores are assigned to each node using different approaches. A popular example of a graphbased keyword extraction method is TextRank, proposed by Mihalcea and Tarau; it utilizes the premise of the popular PageRank algorithm. Due to its simple implementation, we use n-grams to extract keywords from the news media sources. Furthermore, n-grams does not require training or any document corpus for its operation.

## 3.6 CO-OCCURRENCE SIMILARITY

Since establishing the importance of the word-pair co-occurrence distribution in the actual corpus of tweets is of more interest to us, we tested other similarity measures, and found that the Dice similarity measure provided the best results.

## 3.7 STANFORD CoreNLP-NATURAL LANGUAGE SOFTWARE

Stanford CoreNLP provides a set of human language technology tools. It can give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, get the quotes people said, etc. Choose Stanford CoreNLP if we need:

- An integrated NLP toolkit with a broad range of grammatical analysis tools.
- A fast, robust annotator for arbitrary texts, widely used in production.
- Support for a number of major (human) languages.
- A modern, regularly updated package, with the overall highest quality text analytics

## Chapter 4

# SYSTEM REQUIREMENT SPECIFICATION

## 4.1 INTRODUCTION

This chapter describes about the requirements. It specifies the hardware and software requirements that are required in order to run the application properly. The Software Requirement Specification (SRS) is explained in detail, which includes overview of dissertation as well as the functional and non-functional requirement of this dissertation.

A SRS document describes all data, functional and behavioral requirements of the software under production or development. SRS is a fundamental document, which forms the foundation of the software development process. Its the complete description of the behavior of a system to be developed. It not only lists the requirements of a system but also has a description of its major feature. Requirement Analysis in system engineering and software engineering encompasses those tasks that go into determining the need or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, such as beneficiaries or users. Requirement Analysis is critical to the success to a development project. Requirement must be documented, measurable, testable, related to in identified business needs or opportunities, and defined to a level of detail sufficient for system design.

The SRS functions as a blueprint for completing a project. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specification, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and non-functional requirements only.

Thus the goal of preparing the SRS document is to

- To facilitate communication between the customer, analyst, system developers, maintainers.
- To serve as a contrast between purchaser and supplier.
- To firm foundation for the design phase.
- Support system testing facilities.
- Support project management and control.
- Controlling the evolution of the system.

## 4.2 FUNCTIONAL REQUIREMENTS

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

- Input test case must not have compilation and runtime errors.
- The application must not stop working when kept running for even a long time.
- The application must function as expected for every set of test cases provided.
- The application should generate the output for given input test case and input parameters.
- The application should generate on-demand services.

## 4.3 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non-functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

- Product Requirements
- Organizational Requirements
- User Requirements
- Basic Operational Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. This should be contrasted with functional requirements that define specific behaviour or functions. The plan for implementing non-functional requirements is detailed in the system architecture. Broadly, functional requirements define what a system is supposed to do and non- functional requirements define how a system is supposed to be. Functional requirements are usually in the

form of system shall do requirement $i$ , an individual action of part of the system, perhaps explicitly in the sense of a mathematical function, a black box description input, output, process and control functional model or IPO Model. In contrast, non-functional requirements are in the form of system shall be requirement $i$ , an overall property of the system as a whole or of a particular aspect and not a specific function. The systems' overall properties commonly mark the difference between whether the development project has succeeded or failed.

Non-functional requirements of our project include:

- **Response time** The time the system takes to load and the time for responses on any action the user does.
- **Processing time** - How long is acceptable to perform key functions or export / import data?
- **Throughput** The number of transactions the system needs to handle must be kept in mind.
- **Storage** - The amount of data to be stored for the system to function.
- **Growth Requirements** As the system grows it will need more storage space to keep up with the efficiency.
- **Locations of operation** - Geographic location, connection requirements and the restrictions of a local network prevail.
- **Architectural Standards** The standards needed for the system to work and sustain.

### 4.3.1 PRODUCT REQUIREMENTS

- **Portability:** Since the SLR system is designed to run using Arduino (whose library is written in C), the system is portable.
- **Correctness:** It follows a well-defined set of procedures and rules to compute and also rigorous testing is performed to confirm the correctness of the data.
- **Ease of Use:** The front end is designed in such a way that it provides an interface which allows the user to interact in an easy manner.
- **Modularity:** The complete product is broken up into many modules and well-defined interfaces are developed to explore the benefit of flexibility of the product.



- **Robustness:** This software is being developed in such a way that the overall performance is whereas evolution quality involves testability, maintainability, extensibility or scalability.

#### 4.3.1.1 ORGANIZATIONAL REQUIREMENTS

**Process Standards:** IEEE standards are used to develop the application which is the standard used by the most of the standard software developers all over the world.

**Design Methods:** Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

#### 4.3.1.2 USER REQUIREMENTS

The user requirements document (URD) or user requirements specification is a document usually used to software engineering that specifies the requirements the user expects from software to be constructed in a software project. Once the required information is completely gathered it is documented in a URD, which is meant to spell out exactly what the software must do and becomes part of the contractual agreement. A customer cannot demand feature not in the URD, whilst the developer cannot claim the product is ready if it does not meet an item of the URD. The URD can be used as a guide to planning cost, timetables, milestones, testing etc. The explicit nature of the URD allows customers to show it to various stakeholders to make sure all necessary features are described. Formulating a URD requires negotiation to determine what is technically and economically feasible. Preparing a URD is one of those skills that lies between a science and economically feasible. Preparing a URD is one of those skills that lies between a science and an art, requiring both software technical skills and interpersonal skills.

#### 4.3.1.3 BASIC OPERATIONAL REQUIREMENTS

Operational requirement is the process of linking strategic goals and objectives to tactic goals and objectives. It describes milestones, conditions for success and explains how, or what portion of, a strategic plan will be put into operation during a given operational period, in the case of, a strategic plan will be put into operation during a given operational period, in the case of commercial application, a fiscal year or another given budgetary term. An operational plan is the basis for, and justification of an annual operating budget request. Therefore, a five-year strategic plan would typically require five operational plans funded by five operating budgets. Operational plans should establish the activities and budgets for each part of the organization for

the next 1-3 years. They link the strategic plan with the activities the organization will deliver and the resources required to deliver them. An operational plan draws directly from agency and program strategic plans to describe agency and program missions and goals, program objectives, and program activities. Like a strategic plan, an operational plan addresses four questions:

- Where are we now?
- Where do we want to be?
- How do we get there?

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:

- Mission profile or scenario: It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.
- Performance and related parameters: It points out the critical system parameters to accomplish the mission
- Utilization environments: It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.
- Operational life cycle: It defines the system lifetime

## 4.4 HARDWARE REQUIREMENTS

- Micro-Processors : Pentium Dual Core..
- Hard Disk : 120 GB.
- Monitor : 15 LED
- Input Devices : Keyboard, Mouse
- Ram : 1 GB

## 4.5 SOFTWARE REQUIREMENTS

- Operating System : Windows 7.

- Coding Language : Java,JavaScript,HTML,CSS,SQL.
- Tools : Spring and MYsqli Browser
- Database : MYSQL

## 4.6 SOFTWARE QUALITY ATTRIBUTES

- **Functionality:** the capability of the software to provide functions which meet stated and implied needs when the software is used under specified conditions.
- **Reliability:** the capability of the software to maintain its level of performance when used under specified conditions.

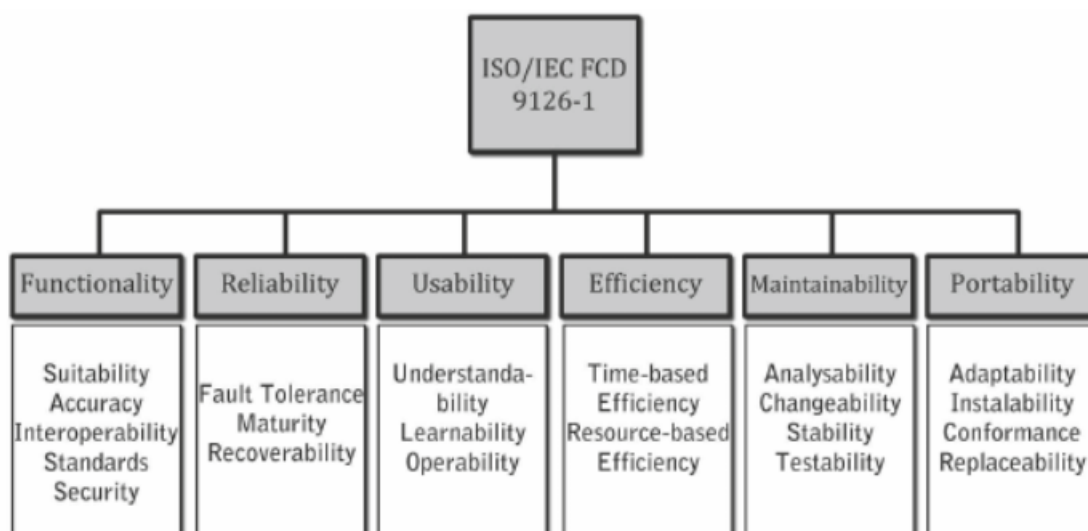


Figure 4.1: Software Quality Attributes

- **Usability:** the capability of the software to be understood, learned, used and liked by the user, when used under specified conditions.
- **Efficiency:** the capability of the software to provide the required performance, relative to the amount of resources used, under stated conditions.
- **Maintainability:** the capability of the software to be modified. Modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications.
- **Portability:** the capability of software to be transferred from one environment to another.

# Chapter 5

## SYSTEM ANALYSIS

## 5.1 INTRODUCTION

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translated into a representation of software. Design is a place where design is fostered in software Engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. Design is the perfect way to accurately translate a customer's requirement in the finished software product. Design creates a representation or model, provides details about software data structure, architecture, interfaces and components that are necessary to implement a system. The logical system design arrived at as a result of systems analysis is converted into physical system design.

## 5.2 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

- Operational Feasibility
- Economical Feasibility
- Technical Feasibility
- Social Feasibility

## 5.3 OPERATIONAL FEASIBILITY

- This system can be implemented in the organization because there is adequate support from various user groups to know the top trending news content and the authenticity on the same.
- Being developed in JAVA so that the necessary operations are carried out automatically and can be run with any platform.
- It is mainly related to human organization and political aspects. These points

considered are: What organizational structures are distributed? What new skills will be required? Do the existing system staff members have these skills? If not, can they be trained in the course of time?

## 5.4 ECONOMICAL FEASIBILITY

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More commonly known as cost / benefit analysis. The procedure is to determine the benefits and savings are expected from a proposed system and compare them with costs. If benefits outweigh costs; a decision is taken to design and implement the system will have to be made if it is to have a chance of being approved. There is an ongoing effort that improves in accuracy at each phase of the system life cycle.

## 5.5 TECHNICAL FEASIBILITY

This is considered with specifying equipment and software that will successfully satisfy the user requirements. The technical needs of the system may vary considerably but might include:

- The facility to produce outputs in a given time.
- Response time under certain conditions.
- Ability to process a certain volume of transaction at a particular speed.

## 5.6 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

# Chapter 6

## SYSTEM DESIGN

## 6.1 INTRODUCTION

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translated into a representation of software. Design is a place where design is fostered in software Engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. Design is the perfect way to accurately translate a customer's requirement in the finished software product. Design creates a representation or model, provides details about software data structure, architecture, interfaces and components that are necessary to implement a system. The logical system design arrived at as a result of systems analysis is converted into physical system design.

## 6.2 SYSTEM DEVELOPMENT METHODOLOGY

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

### 6.2.1 MODEL PHASES

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design, Implementation, Testing and maintenance.

- **Requirement Analysis:** This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.
- **System Design:** Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on:-algorithm, data structure, software architecture etc.
- **Coding:** In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in to machine readable compute code.



- **Implementation:** The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation
- **Testing:** In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.
- **Maintenance:** The maintenance phase is the longest phase in which the software is updated to fulfill the changing customer need, adapt to accommodate change in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

### 6.2.2 REASON FOR CHOOSING WATERFALL MODEL AS DEVELOPMENT METHOD

- Clear project objectives.
- Stable project requirements.
- Progress of system is measurable.
- Strict sign-off requirements.
- Helps you to be perfect.
- Logic of software development is clearly understood.
- Production of a formal specification
- Better resource allocation.
- Improves quality. The emphasis on requirements and design before writing a single line of code ensures minimal wastage of time and effort and reduces the risk of schedule slippage.
- Less human resources required as once one phase is finished those people can start working on to the next phase.

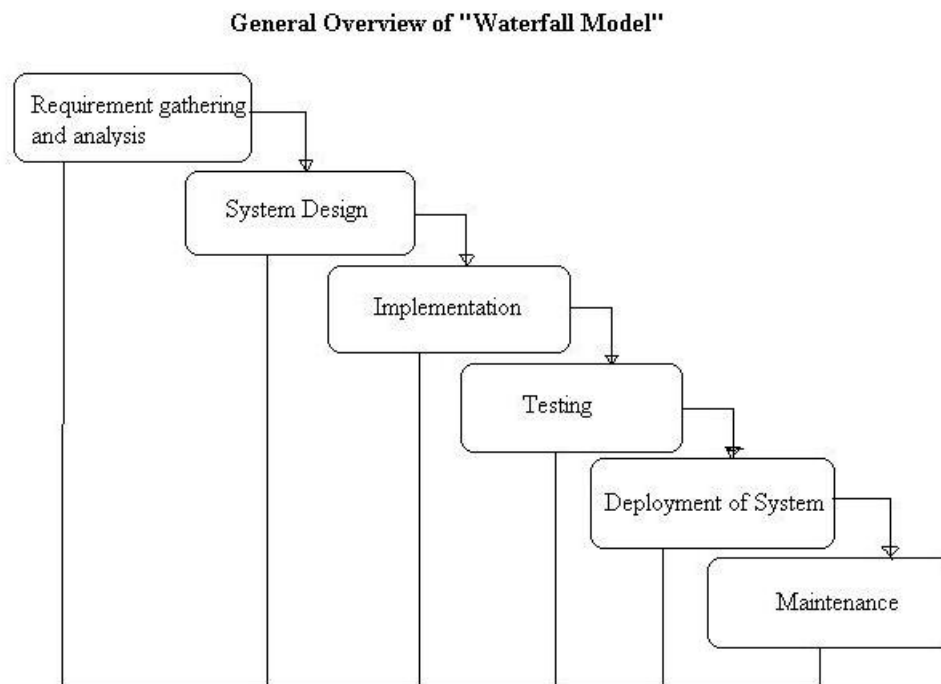


Figure 6.1: Waterfall Model

### 6.3 DESIGN USING UML

Designing UML diagram specifies, how the process within the system communicates along with how the objects within the process collaborate using both static as well as dynamic UML diagrams since in this ever-changing world of Object Oriented application development, it has been getting harder and harder to develop and manage high quality applications in reasonable amount of time. As a result of this challenge and the need for a universal object modeling language every one could use, the Unified Modeling Language (UML) is the Information industries version of blue print. It is a method for describing the systems architecture in detail. Easier to build or maintains system, and to ensure that the system will hold up to the requirement changes.

### 6.4 DATA FLOW DIAGRAM

A data flow diagram (DFD) is graphic representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. DFDs show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage. There are only four











<i>Line</i>	<i>Symbol</i>
Association	<u>AssociationName</u>
Aggregation	
Generalization	
Dependency	
Activity edge	
Event, transition	event[guard]/action 
Link	<u>.inkName</u>
Composition	
Realization	
Assembly connection	
Message	some code 
Control flow	

Table 6.1: Symbols used in UML

symbols: 1. Squares representing external entities, which are sources and destinations of information entering and leaving the system. 2. Rounded rectangles representing processes, in other methodologies, may be called 'Activities', 'Actions', 'Procedures', 'Subsystems' etc. which take data as input, do processing to it, and output it. 3. Arrows representing the data flows, which can either, be electronic data or physical items. It is impossible for data to flow from data store to data store except via a process, and external entities are not allowed to access data stores directly. 4. The flat three-sided rectangle is representing data stores should both receive information for storing and provide it for further processing.

## 6.5 CLASS DIAGRAM

UML class diagram shows the static structure of the model. The class diagram is a collection of static modeling elements, such as classes and their relationships, connected as a graph to each other and to their contents. The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects and or interactions in the application and the objects to be programmed.

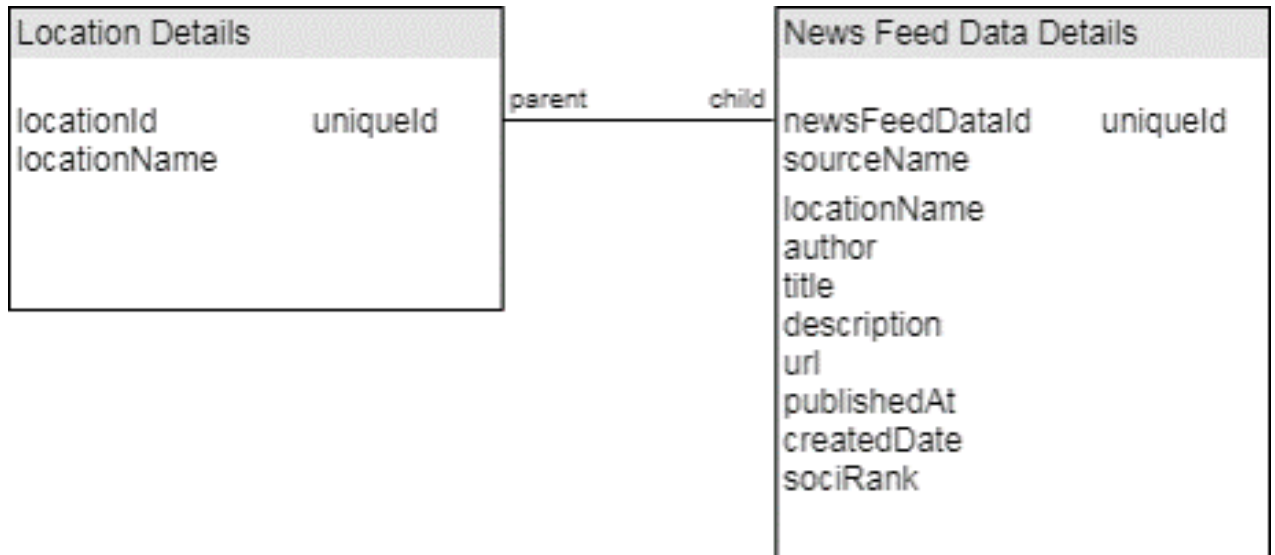


Figure 6.2: Class Diagram of Components

## 6.6 USE CASE DIAGRAM

A use case defines a goal-oriented set of interactions between external entities and the system under consideration. The external entities which interact with the system are its actors. A set of use cases describe the complete functionality of the system at a particular level of detail and it can be graphically denoted by the use case diagram.

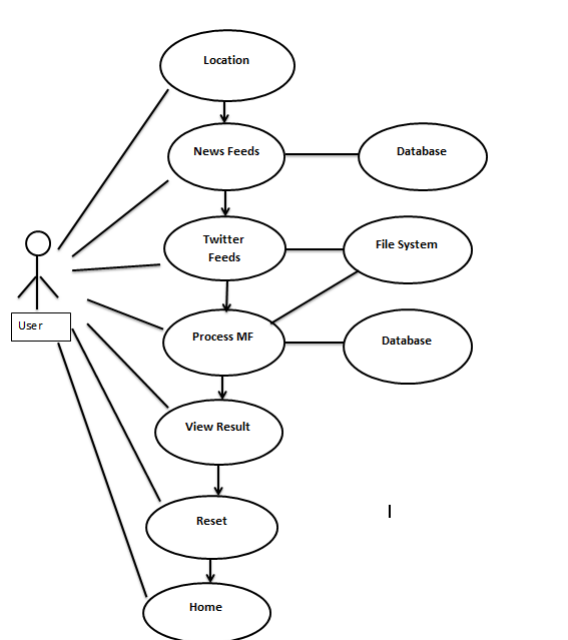


Figure 6.3: Use Case Diagram

## 6.7 ACTIVITY DIAGRAM

An activity diagram shows the sequence of steps that make up a complex process. An activity is shown as a round box containing the name of the operation. An outgoing solid arrow attached to the end of the activity symbol indicates a transition triggered by the completion.

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

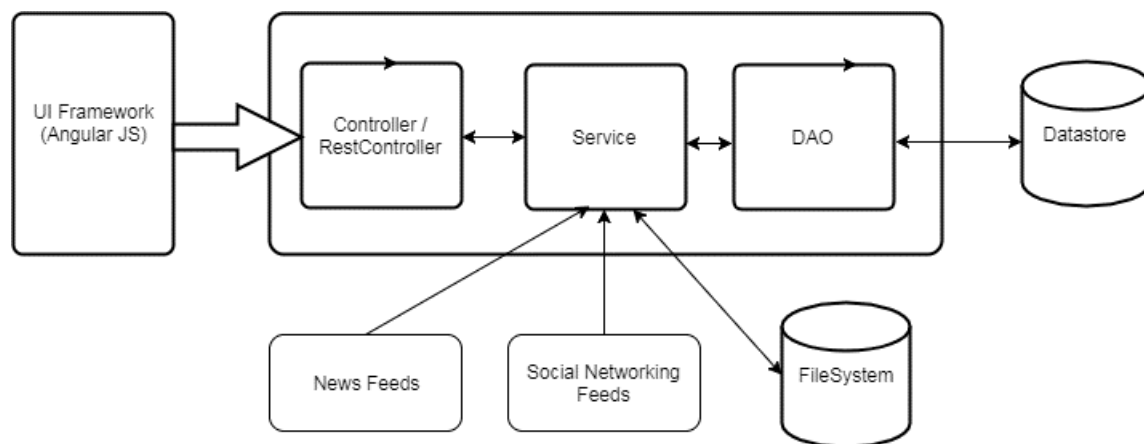


Figure 6.4: Activity Diagram

## 6.8 SEQUENCE DIAGRAM

Sequence diagram are an easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram has two dimensions: vertical dimension represents time, the horizontal dimension represents the objects existence during the interaction. **Basic elements:**

- **Vertical rectangle:** Represent the object is active (method is being performed).
- **Vertical dashed line:** Represent the life of the object.
- **X:** represent the life end of an object. (Being destroyed from memory)
- **Horizontal line with arrows:** Messages from one object to another.

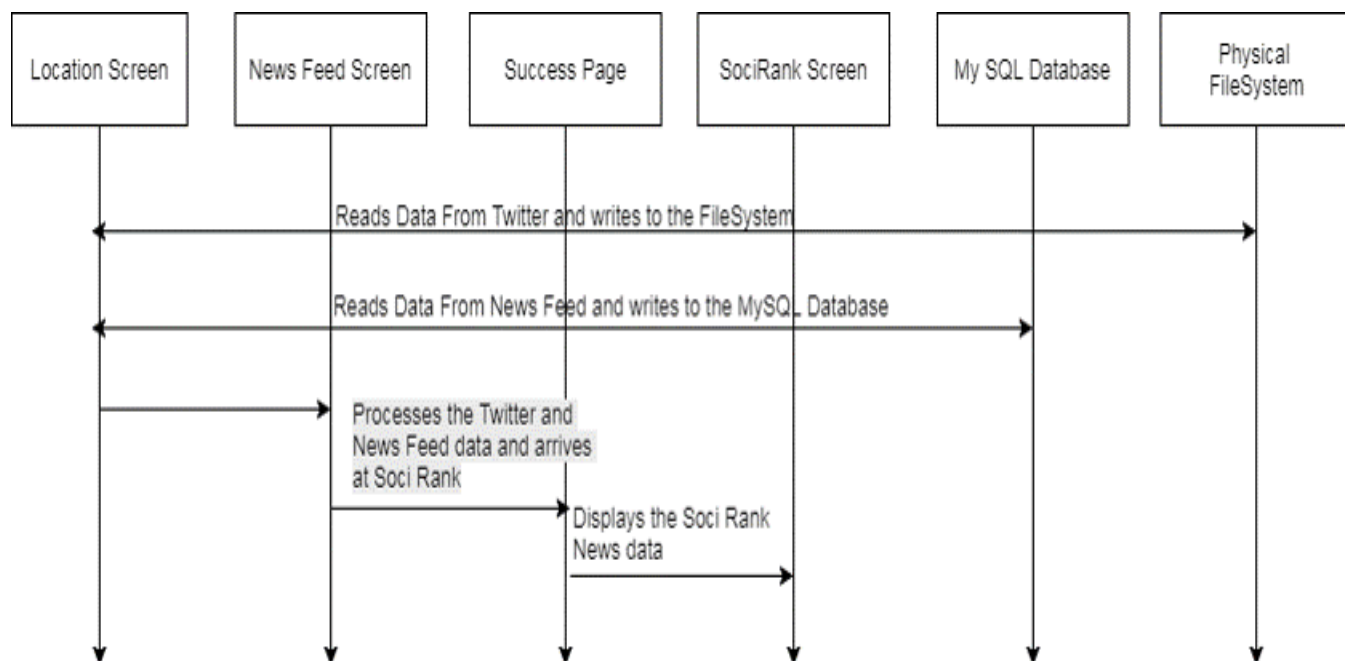


Figure 6.5: Sequence Diagram

# Chapter 7

## IMPLEMENTATION

## 7.1 INTRODUCTION

The implementation phase of the project is where the detailed design is actually transformed into working code. Aim of the phase is to translate the design into a best possible solution in a suitable programming language. This chapter covers the implementation aspects of the project, giving details of the programming language and development environment used. It also gives an overview of the core modules of the project with their step by step flow. The implementation stage requires the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.
- Evaluation of the changeover method.
- Correct decisions regarding selection of the platform.
- Appropriate selection of the language for application development.

## 7.2 SOCIRANK CODE

```
fetch news topics
```

```
public List<NewsAPI> readNewsFeedByLocation(String locationName) {
    DefaultHttpClient httpClient = new DefaultHttpClient();
    logger.info(" _NewsFeedMain_:: _main_:: _STARTS_");
    List<NewsAPI> newsFeedList = null;
    try{
        newsFeedURL=newsFeedURL.replace("CITY" , locationName);
        HttpGet getRequest = new HttpGet(newsFeedURL);
        getRequest.addHeader("accept" , "application/json");
        HttpResponse response = httpClient.execute(getRequest);
        if (response.getStatusLine().getStatusCode() != 200) {
            throw new RuntimeException(" Failed _: _HTTP_error_code_: _"+
        }
        BufferedReader reader = new BufferedReader(new InputStreamReader
        ((response.getEntity().getContent())));
        newsFeedList = JasonParserUtils.convertNewsAPIJason(reader , locationName
```



```

logger.info(" NewsFeedList:: " + newsFeedList);
} catch (Exception e) {
    logger.info(" Exception:: Message:: " + e.getMessage());
    e.printStackTrace();
} finally {
    httpClient.getConnectionManager().shutdown();
}

        return newsFeedList;
}

```

fetch twitter data

```

public void processNewsFeedDataReadByLocation(String locationName) {
    logger.info(" SocIRankTwitterServiceImpl::
    getNewsFeedDataReadByLocation:: STARTS ");
    NewsFeedReadByLocation newsFeedUtility = new NewsFeedReadByLocation();
    List<NewsAPI> newsList =
        newsFeedUtility.readNewsFeedByLocation(locationName);
    if(newsList != null && newsList.size() > 0) {
        socialCommonDAO.deleteNewsFeedData(locationName);
        socialCommonDAO.insertNewsFeedData(newsList);
    }
}

```

comparing news and twitter data

```

public static int calculateSentenceSimilarity(
    String newsfeed, String sentences) {

    WS4JConfiguration.getInstance().setMFS(true);
    String [] words1 = newsfeed.split(" ");
    String [] words2 = sentences.split(" ");
    RelatednessCalculator rc = new Resnik(db);
    int score = getSimilarityMatrix(words1, words2, rc);
    return score;
}

```

```

public static int getSimilarityMatrix(String [] words1,
    String [] words2, RelatednessCalculator rc) {

```

```

        double [][] result = new double[words1.length][words2.length];
        int count = 0;
        for (int i = 0; i < words1.length; i++) {
        for (int j = 0; j < words2.length; j++) {
        double score = rc.calcRelatednessOfWords(words1[i], words2[j]);
        if(score > 1.0) count++;
        }
    }
return count;
}

```

### Processing Twitter Data

```

public Set<String> processLDAForKeywordExtraction(String filePath ,
    List<String> tweetsData) {
    logger.info("FileHandlerUtility_::
    processLDAForKeywordExtraction_::_STARTS" + filePath);
    Set<String> sentenceSet = new HashSet<String>();
    String line = null;
    Path path = Paths.get(filePath);
    try(BufferedReader reader = Files.newBufferedReader(path,
    Charset.forName("ISO-8859-1"))) {
    String folderPath = filePath.substring(0, filePath.lastIndexOf("\\"));
    String fileName = filePath.substring(filePath.lastIndexOf("\")+1);
    Path ldaPath = Paths.get(folderPath + "/LDA");
    if(!Files.exists(ldaPath)) {
    Files.createDirectories(ldaPath);
    }
    Path ldaFilePath = Paths.get(ldaPath.toString(), fileName);
    if(Files.exists(ldaFilePath)) {
    Files.deleteIfExists(ldaFilePath);
    }
    Files.createFile(ldaFilePath);
    for(String tweet : tweetsData) {
    try {
        line = CommonsUtility.getOnlyStringsDigits(tweet);
        line = CoreNLPUtililty.applyStopWords(line);
        System.out.println(line+"\n");
        sentenceSet.add(line);
    }
    }
}

```

```

        Files.write(ldaFilePath, (line + "\n").getBytes(),
StandardOpenOption.APPEND);
    } catch(Exception e) {
        logger.error("Exception:: Extract Keywords:: " + e.getMessage());
    }
}
logger.info("Keywords Size:: " + sentenceSet.size());
} catch(Exception e) {
logger.error(" Exception in processLDAForKeywordExtraction:: "
+ e.getMessage());
}
logger.info(" FileHandlerUtility:: processLDAForKeywordExtraction:: END");
return sentenceSet;
}

```

### Saving Twitter Data to File System

```

public synchronized String writeTwitterDataToFileSystem(List<Status>
twitterData, String locationName, List<String> tweetData) {
logger.info(" FileHandlerUtility::
writeTwitterDataToFileSystem:: STARTS");
Path filePath = null;
try {
String currentDate = CommonsUtility.getCurrentDate();
Path basePath = Paths.get(baseFolderPath + currentDate);
if(!Files.exists(basePath)) {
Files.createDirectories(basePath);
}
Path locationPath = Paths.get(basePath.toString() + "/" + locationName);
if(!Files.exists(locationPath)) {
Files.createDirectories(locationPath);
}
String fileName = "twitterdata_" +
CommonsUtility.getCurrentDateTimeForFileName() + ".txt";
filePath = Paths.get(locationPath.toString(), fileName);
if(!Files.exists(filePath)) {
Files.createFile(filePath);
}
}

```

```

for(Status status : twitterData ) {
Files.write(filePath , (status.getText() + "\n").getBytes() ,
StandardOpenOption.APPEND);
tweetData.add(status.getText());
}
} catch (Exception e) {
logger.error(" _Exception_in_writeTwitterDataToFileSystem_::_" + e.getMes
}
logger.info(" FileHandlerUtility_::_writeTwitterDataToFileSystem_::_ENDS"
return filePath.toString();
}
}

```

Sending Mail to Users

```

private void sendNotificationToUser(String locationName) {
logger.info(" SocialCommonServiceImpl_::_sendNotificationToUser_::
STARTS_");
String subject = " Notification_on_Data_Generated_For_The
News_Feed_Comparison";
StringBuffer stbBody = new StringBuffer();
stbBody.append(" Hi_\n");
stbBody.append(" Please_use_the_URL_below_to_view_the_Detailed_comparison
on_the_News_Feed_data_over_the_Social_Data_" + locationName + "\n\n");
stbBody.append(
"<a_href='http://localhost:9090/socirank/socirankdata/' +
locationName + "'>_Click_Here_<a>\n\n\n");
stbBody.append(" Thanks,_\n");
stbBody.append(" Administrator\n");
String body = stbBody.toString();
EmailUtility email = new EmailUtility();
email.sendEmail(subject , body);
logger.info(" TwitterServiceImpl_::_sendNotificationToUser_::_STARTS_");
}
}

```

### 7.3 CONNECTING ANGULAR AND SPRING

```

angular.module('myApp').controller('SocialController', ['$scope',
'$window', 'NgTableParams', 'SocialService',
function($scope, $window, NgTableParams, SocialService) {

```

```

var self = this;
self.location={locationId:null, locationName:''};
self.newsFeed={newsFeedDataId: null, sourceName: '', author: '', title:
'', description: '',
url: '', urlToImage: '', publishedAt: '', sociRankScore: null }
self.locations = [];
self.newsFeeds = [];
self.submit = submit;
self.reset = reset;
$scope.init = function (value) {
if(value === 'location') fetchLocationDetails();
if(value === 'newsfeed') fetchNewsFeedDetails();
if(value === 'socirank') fetchSociRankNewsDetails();
};

```

## 7.4 CONNECTING SPRING AND MYSQL

Accessing data from database

```

public class SocialCommonDAOImpl implements SocialCommonDAO {
private Logger logger = Logger.getLogger(SocialCommonDAOImpl.class);
private static final String UPDATE_NEWSFEED_DATA =
    "update NEWSFEED_DATA
set SOCIRANK_=? where NEWSFEED_DATA_ID=?";
@Autowired
@Qualifier("jdbcTemplate")
JdbcTemplate jdbcTemplate;
@Override
@SuppressWarnings({ "unchecked", "rawtypes" })
public List<Location> getLocationDetails() {
logger.info("CreditCardDetailsDAOImpl:: getLocationDetails:: STARTS")
List<Location> locationList = jdbcTemplate.query(SELECT_LOCATION_DATA,
new RowMapper() {
public Location mapRow(ResultSet rs, int rowNum) throws SQLException {
Location location = new Location();
location.setLocationId(rs.getInt("LOCATION_ID"));
location.setLocationName(rs.getString("LOCATION_NAME"));
return location;
}
}

```

```
});  
logger.info("CreditCardDetailsDAOImpl::_getLocationDetails::_ENDS_");  
return locationList;  
}  
}
```

## Chapter 8

# TESTING AND RESULTS

## 8.1 TESTING

Unit testing is a development procedure where programmers create tests as they develop software. The tests are simple short tests that test functionally of a particular unit or module of their code, such as a class or function. Using open source libraries like cunit, oppunit and nun it (for C, C++ and C#) these tests can be automatically run and any problems found quickly. As the tests are developed in parallel with the source unit test demonstrates its correctness.

## 8.2 VALIDATION AND SYSTEM TESTING

Validation testing is a concern which overlaps with integration testing. Ensuring that the application fulfils its specification is a major criterion for the construction of an integration test. Validation testing also overlaps to a large extent with System Testing, where the application is tested with respect to its typical working environment. Consequently for many processes no clear division between validation and system testing can be made. Specific tests which can be performed in either or both stages include the following.

- **Regression Testing:** Where this version of the software is tested with the automated test harness used with previous versions to ensure that the required features of the previous version are still working in the new version.
- **Recovery Testing:** Where the software is deliberately interrupted in a number of ways off, to ensure that the appropriate techniques for restoring any lost data will function.
- **Security Testing:** Where unauthorized attempts to operate the software, or parts of it, attempted it might also include attempts to obtain access the data, or harm the software installation or even the system software. As with all types of security determined will be able to obtain unauthorized access and the best that can be achieved is to make this process as difficult as possible.
- **Stress Testing:** Where abnormal demands are made upon the software by increasing the rate at which it is asked to accept, or the rate at which it is asked to produce information. More complex tests may attempt to create very large data sets or cause the software to make excessive demands on the operating system.
- **Performance testing:** Where the performance requirements, if any, are checked. These may include the size of the software when installed, type amount of main



memory and/or secondary storage it requires and the demands made of the operating when running with normal limits or the response time.

- **Usability Testing:** The process of usability measurement was introduced in the previous chapter. Even if usability prototypes have been tested whilst the application was constructed, a validation test of the finished product will always be required.
- **Alpha and beta testing:** This is where the software is released to the actual end users. An initial release, the alpha release, might be made to selected users who be expected to report bugs and other detailed observations back to the production team. Once the application changes necessitated by the alpha phase can be made to larger more representative set users, before the final release is made to all users. The final process should be a Software audit where the complete software project is checked to ensure that it meets production management requirements. This ensures that all required documentation has been produced, is in the correct format and is of acceptable quality. The purpose of this review is: firstly to assure the quality of the production process and by implication construction phase commences. A formal hand over from the development team at the end of the audit will mark the transition between the two phases.
- **Integration Testing:**

Integration Testing can proceed in a number of different ways, which can be broadly characterized as top down or bottom up. In top down integration testing the high level control routines are tested first, possibly with the middle level control structures present only as stubs. Subprogram stubs were presented in section2 as incomplete sub programs which are only present to allow the higher. Level control routines to be tested.

Top down testing can proceed in a depth-first or a breadth-first manner. For depth-first integration each module is tested in increasing detail, replacing more and more levels of detail with actual code rather than stubs. Alternatively breadth-first would processed by refining all the modules at the same level of control throughout the application .in practice a combination of the two techniques would be used. At the initial stages all the modules might be only partly functional, possibly being implemented only to deal with non-erroneous data. These would be tested in breadth-first manner, but over a period of time each would be replaced with successive refinements which were closer to the full functionality. This allows depth-first testing of a module to be performed

simultaneously with breadth-first testing of all the modules.

The other major category of integration testing is Bottom Up Integration Testing where an individual module is tested from a test harness. Once a set of individual module have been tested they are then combined into a collection of modules ,known as builds, which are then tested by a second test harness. This process can continue until the build consists of the entire application. In practice a combination of top down and bottom-up testing would be used. In a large software project being developed by a number of sub-teams, or a smaller project where different modules were built by individuals. The sub teams or individuals would conduct bottom-up testing of the modules which they were constructing before releasing them to an integration team which would assemble them together for top-down testing.

- Unit Testing:

Unit testing deals with testing a unit as a whole. This would test the interaction of many functions but confine the test within one unit. The exact scope of a unit is left to interpretation. Supporting test code, sometimes called Scaffolding, may be necessary to support an individual test. This type of testing is driven by the architecture and implementation teams. This focus is also called black-box testing because only the details of the interface are visible to the test. Limits that are global to a unit are tested here.

In the construction industry, scaffolding is a temporary, easy to assemble and disassemble, frame placed around a building to facilitate the construction of the building. The construction workers first build the scaffolding and then the building. Later the scaffolding is removed, exposing the completed building.similarly, in software testing, one particular test may need some supporting software. This software establishes can a correct evaluation of the test take place. The scaffolding software may establish state and values for data structures as well as providing dummy external functions for the test. Different scaffolding software may be needed form one test to another test. Scaffolding software rarely is considered part of the system.

Some times the scaffolding software becomes larger than the system software being tested. Usually the scaffolding software is not of the same quality as the system software and frequently is quite fragile. A small change in test may lead to much larger changes in the scaffolding. Internal and unit testing can be automated with the help of coverage tools. Analyzes the source code and generated a test that will execute every alternative thread of execution. Typically, the coverage tool is used in a slightly different way. First the coverage

tool is used to augment the source by placing information prints after each line of code. Then the testing suite is executed generating an audit trail. This audit trail is analyzed and reports the percent of the total system code executed during the test suite. If the coverage is high and the untested source lines are of low impact to the systems overall quality, then no more additional tests are required.

### 8.3 RESULTS

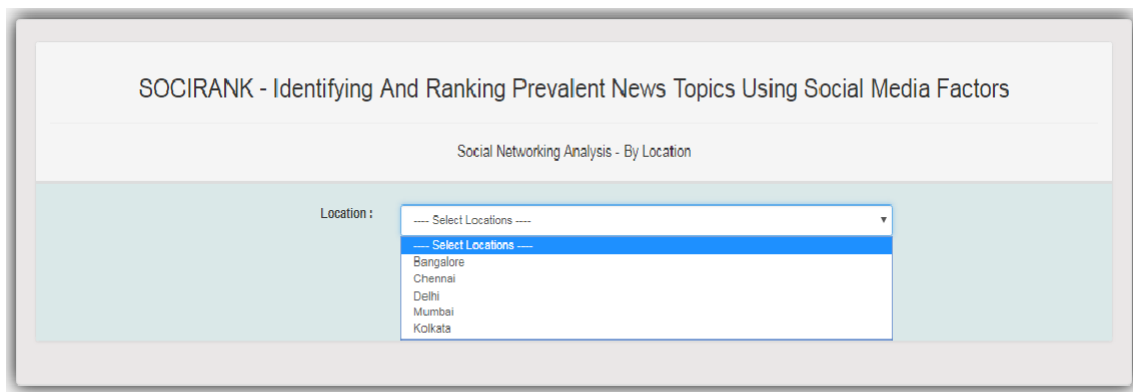


Figure 8.1: search by location

News Feed - Current Trends By Location			
News Track Id	Author	Title	Description
351	PTI	India hockey coach complains about food quality, IOA draws SAI's attention	Harendra Singh has complained to Hockey India about the sub-standard quality of food and hygiene level at the SAI centre in Bengaluru, where the national team is currently training ahead of the Champions Trophy.
352	Rahul Sadhu	Afghanistan have "superman bowlers," should set alarm bells ringing in Indian dressing room: Shapoor Zadran	Shapoor Zadran has a warning for the Indian contingent - beware of the threat that comes with facing the Afghan spin battery.
353	Sreeradha D Basu and Brinda Sarkar	From Philips to Infy, how India Inc is creating future-ready workers	From Philips to Infy, how India Inc is creating future-ready workersSeveral companies in India are helping people cope better in a fast-changing scenario by developing market-relevant skills.
354	ET Now	We won't burden people with rail fare, freight hike: Piyush Goyal	We are still paying for the huge subsidies and oil bonds of the UPA government, says the interim FM.
355	Matias S. Zavia	Nueve personas inocentes han muerto linchadas en la	Nilotpal Das y Abijeet Nath detuvieron su coche para pedir indicaciones y acabaron muriendo a manos de una muchedumbre histrica. En

Figure 8.2: news feeds

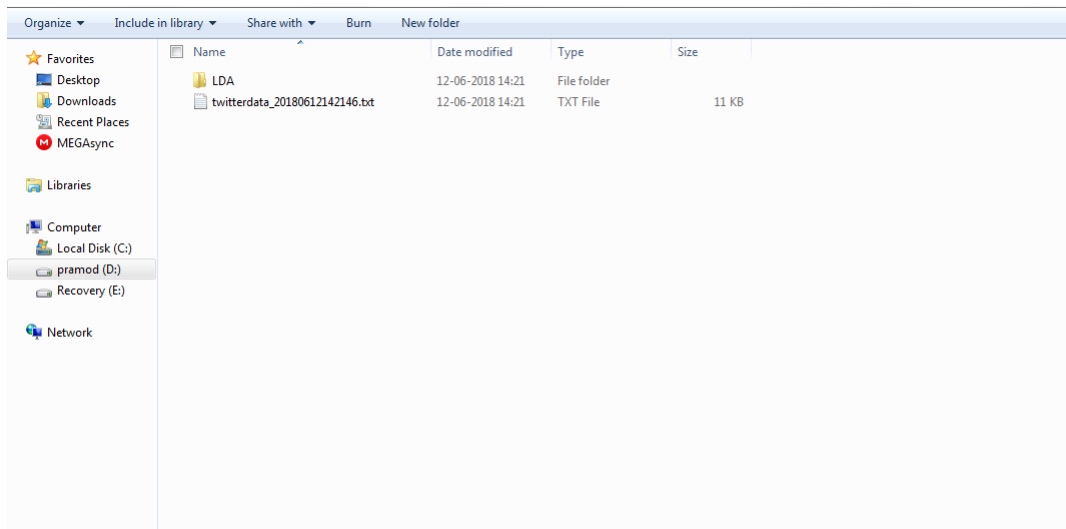


Figure 8.3: twitter data

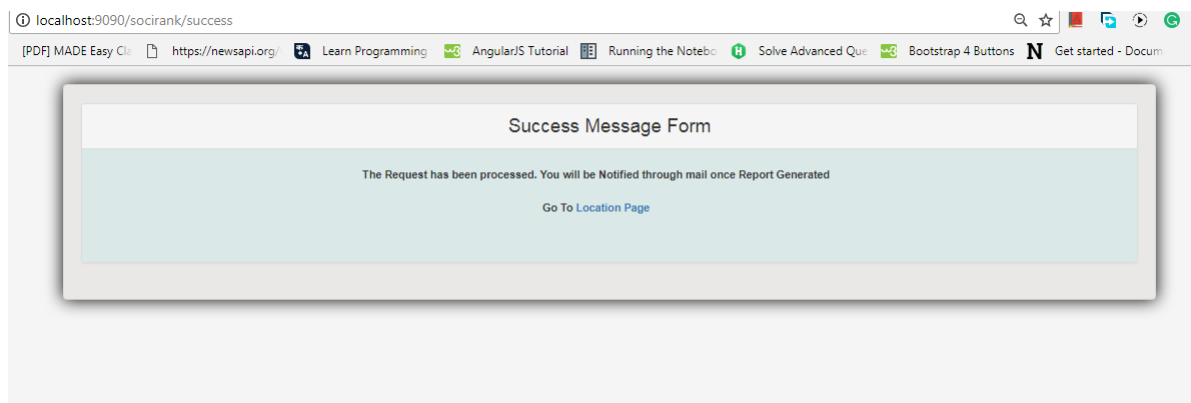


Figure 8.4: success screen

**News Feed - Current Trends By Location**

Soci Rank	News Rank	Author	Title	Description
1	3	Rahul Sadhu	Afghanistan have "superman bowlers," should set alarm bells ringing in Indian dressing room: Shapoor Zadran	Shapoor Zadran has a warning for the Indian contingent - beware of the threat that comes with facing the Afghan spin battery.
2	5	ET Now	We won't burden people with rail fare, freight hike: Piyush Goyal	We are still paying for the huge subsidies and oil bonds of the UPA government, says the interim FM.
3	10	Subrat Patnaik	With assembly team and samosas, IKEA lays ground for India debut	MUMBAI/STOCKHOLM (Reuters) - IKEA is set to open its first store in India next month, and the world's biggest furniture retailer is localizing its offering - from the food menu to its "do-it-yourself" (DIY) assembly model - to ensure success in the country.
4	2	PTI	India hockey coach complains about food quality, IOA draws SAI's attention	Harendra Singh has complained to Hockey India about the sub-standard quality of food and hygiene level at the SAI centre in Bengaluru, where the national team is currently training ahead of the Champions Trophy.
5	7		Ikea to open in India after	Hyderabad superstore will sell chicken biryani as well as furniture

Figure 8.5: socirank

## Chapter 9

# CONCLUSION & FUTURE SCOPE

## 9.1 CONCLUSION

In this paper, we proposed an unsupervised method SociRank which identifies news topics prevalent in both social media and the news media, and then ranks them by taking into account their MF, UA, and UI as relevance factors. The temporal prevalence of a particular topic in the news media is considered the MF of a topic, which gives us insight into its mass media popularity. The temporal prevalence of the topic in social media, specifically Twitter, indicates user interest, and is considered its UA. Finally, the interaction between the social media users who mention the topic indicates the strength of the community discussing it, and is considered the UI. To the best of our knowledge, no other work has attempted to employ the use of either the interests of social media users or their social relationships to aid in the ranking of topics.

## 9.2 FUTURE SCOPE

As future work, we intend to perform experiments and expand SociRank on different areas and datasets. Furthermore, we plan to include other forms of UA, such as search engine click-through rates, which can also be integrated into our method to provide even more insight into the true interest of users. Additional experiments will also be performed in different stages of the methodology. For example, a fuzzy clustering approach could be employed in order to obtain overlapping TCs. Lastly, we intend to develop a personalized version of SociRank, where topics are presented differently to each individual user.

# References

- [1] D. M. Blei, A. Y. Ng, and M. I. Jordan, Latent Dirichlet allocation, *J. Mach. Learn. Res.*, vol. 3, pp. 9931022, Jan. 2003.
- [2] T. Hofmann, Probabilistic latent semantic analysis, in *Proc. 15th Conf. Uncertainty Artif. Intell.*, 1999, pp. 289296.
- [3] D. Bollegala, Y. Matsuo, and M. Ishizuka, Measuring semantic similarity between words using Web search engines, in *Proc. WWW, Banff, AB, Canada, 2007*, pp. 757766.
- [4] About coreNLP <https://stanfordnlp.github.io/CoreNLP/>
- [5] About LDA on Wikipedia <https://github.com/myleott/JGibbLabeledLDA/tree/master/src/jgibbllda>
- [6] About Spring <https://spring.io/>
- [7] About Angular on <https://angular.io/>
- [8] About JDBC <https://docs.oracle.com/javase/tutorial/jdbc/basics/connecting.html>
- [9] about REST [https://en.wikipedia.org/wiki/Representational\\_state\\_transfer](https://en.wikipedia.org/wiki/Representational_state_transfer)