Technical Design Document

V2D

Video to Data
Product Development

Provided by

iData Sciences, LLC
April 15, 2014

TABLE OF CONTENTS

0 PREFACE

0.1 PURPOSE OF THIS DOCUMENT

#1 *This document is a Technical Design Document on project V2D for use by IDS-Vū. It provides guidance and material which is intended to assist the developers, relevant management or technical staff, in producing a project-specific Technical Design Document. It is also useful background reading for anyone involved in developing or monitoring the IDS-Vū V2D Project Management System.*

0.2 USE OF THIS DOCUMENT

#1 *This Preface is addressed to the users of this generic document and is not meant to be retained in any project-specific Technical Design Document documents based on it.*

#2 *The material contained in this document is not necessarily exhaustive; if there is a subject that is relevant to the V2D Project, but is not included in this document, it shall still be included.*

#3 *The document is intended for developers and all the stake holders of V2D project*

#4 *This document has been prepared using Microsoft Word 2013.*

a. *“Summary” Properties*

Title	*Technical Design Document*
Author	*Venkatanathan Dwarakanathan*
Keywords	*Document reference (i.e. IDS-VU-TD)*

b. *“Custom” Properties*

Proj Id	*V2D*
Project	*Video to Data*
Contr Id	*IDS*
Contract	*iData Sciences*
Version	*Version 1.0(Draft)*
Date	*April 15, 2014*

0.3 OVERVIEW

#1 *This preface is for information only.*

#2 *Vū Digital ("Vū") and iData Sciences ("IDS") would like to design and develop a video to data product which automatically converts video unstructured data to structured data. The design and development includes custom development infrastructure and resources. The solution would include both a front-end deployment and supporting back-end operations with IT infrastructure needed for Vū integrated deployment.*

#3 *Presently, it appears, there are no automated solution that can read a video, extract all of its images, objects, text, and audio, and return a list of meaningful words or data that is indexable and searchable. The purpose of the digital video to data product is to enable content producers to make their video more available for search. Content distributors can deliver personalized video recommendations; brands can search and find their logos, while advertisers can target their ads for specific circumstances and appropriate targets.*

#4 *This engagement is for designing and developing a video to data product which, unlike any current products in the marketplace, automatically converts video unstructured data to structured data. It takes all of the audio, images, and text from the video - all words, names, terms, expressions, faces, etc. - and with technology refines them down to machine readable words indexed according to time; and all faster than real time. The product converts video and audio into the machine-readable words for search engines, databases, NLP, and distributors.*

#5 *Most video search services rely on manually created 'metadata' – titles, tags, filenames, descriptions, etc. – associated with videos to determine the contents of a video and compute its relevance. But, in most instances the manually attributed metadata doesn't accurately convey what content is in the video, what is the context of the video and does not even consider when this content or context appears in time. Technology for creating searchable data from video is manual, slow, ineffective and inaccurate. Video titles and manual created tags cannot remotely capture the content and context of a video. Manually titling and tagging videos may be good enough today, but this will certainly become a challenge in the coming months and years as the volume of video increases in orders of magnitude.*

0.4 BASIS OF THIS DOCUMENT

#1 *This following sections set out an approach to designing systems that may be developed under V2D. It attempts to set standards and create a consistent approach to the design and development of systems across the V2D Programme. It will enable the Programme to benefit from 'economies of scale' and a consistency in the approach to building and deploying systems. Important issues that need to be considered include the architecture of systems, links to Database systems and components from libraries aims for code re-use and the need to develop systems that will work on an operational basis over many years and the associated desire to make such systems easily supportable and affordable.*

#2 *A key point will be to tweak and reorient the components already built by IDS in MapOptis framework to achieve the objectives of V2D and build the work on collaborative mode with Vū Digital.*

#3 *The concept of a* ***Reference Architecture*** *is also introduced as part of the process of creating an interoperable environment which facilitates the exchange of information between components through distributed computing framework deployed by Vū Digital and also setting out a number of standard building blocks around which solutions can be assembled. These building blocks or 'components reflect the emerging technologies that shall form the technical basis of the developments. These include the Component Object Request Broker Architecture, Remote Method Invocation (RMI) and Enterprise JavaBeans (EJB) technologies that highlight code re-use, scalability and the creation of interoperable architectures around legacy environments.*

0.5 A REFERENCE ARCHITECTURE FOR THE V2D PROGRAMME

#1 *In practice the V2DProgramme shall seek to create what can be referred to as a* ***Reference Architecture*** *for systems – in effect a series of established building blocks on which a variety of systems shall be assembled. The Reference Architecture shall then become part of the* V2D Architecture Guidelines.

#2 *The Reference Architecture would take an n-tier, Cloud model as its basis, and state that the persistence layer would be implemented through a number of standard Component calls to a particular database system. This use of a Reference Architecture would provide some guidelines tasked with developing individual elements of the V2D Programme.*

#3 *Recommendations could be made on the reporting tool that shall be used in systems if one were required, the Web Browser could be standardised, as could any security features that needed to be built into the system. The Reference Architecture would facilitate re-use throughout the V2D Programme as individual module could be arranged to support the development of libraries of components that can be re-used. This will be one of the major potential benefits that could arise from the creation of a Reference Architecture.*

0.6 SPECIFIC DESIGN CONSIDERATIONS

#1 *Perhaps one of the most crucial aspects of this is the level of concurrency that will be present in typical V2D projects.*

#2 *OOD will also become increasingly used in the V2D Programme as developments seek to create systems that are more maintainable over their deployment lifetime. The knock on effect of this will be an increasing emphasis on the use of Commercial Off The Shelf (COTS) packages in the software architecture in areas such as the middleware(if any), reporting tools and in the Graphical User Interface.*

#3 *V2D projects may well shift from a traditional waterfall development approach to the design and development of solutions towards AGILE methodology. Staged development, through so-called 'timeboxes', offers many attractions to designers and developers of contemporary systems. Agile software development method based on iterative and incremental development, where requirements and solutions evolve* *through collaboration between cross-functional teams It promotes adaptive planning, evolutionary development and delivery, iterative approach, and encourages rapid and flexible response to change.*

#4 V2D Project Leaders shall give serious consideration to the adoption of this approach as a way of ensuring systems are delivered into operation that meet up-to-date user requirements.

#5 Specific attention will also have to be placed in the choice of language for the implementation of the system. Today various options exist, and it is necessary as part of the design process to give very careful consideration to a range of trade-offs that need to be factored into development of a consistent design for the architecture. Languages such as Java, C, C++ and emerging languages, such as PYTHON, each offer different benefits to the developer. Careful consideration has been given to the choice of which language to implement the system to gain the appropriate long-term benefits through the full lifecycle of the system.V2D will be created using primarily C/C++

#6 It is therefore recommended that the V2D Programme also standardise on a number of tools that support development and deployment of software in the projects associated with the programme.

1 INTRODUCTION

Although there are some products on the market that claim they analyze video, primarily in the entertainment industry or pertaining to copyright infringement, there are no solutions that use distributed processing to ensure that the breakdown and return of data is faster than the length of the original video. Vū is an automated, distributed video distillation and metataging engine that processes all of the kinds of information most video search services do, but then goes a few steps further by applying a proprietary processes, using "distributed computation" and cloud based computer processing power to analyze videos, translate audio, identify objects, identify people, and extract all kinds of additional information directly from the video itself. All of this information is converted into invaluable, machine-readable metadata for the client. With distributed processing, a one hour video should take a fraction of the play-time to interpret.

1.1 PURPOSE

The purpose of V2D is to develop a product with the following main capabilities:

1. *Image recognition and classification*
2. *Facial recognition*
3. *Text Extraction for images*
4. *Speech to text transcription*
5. *Orchestrate the entire work in a Distributed Environment*

Additionally, Vū would like to develop a reference database, so all images can be cross-referenced for recognition and accuracy.

1.2 OBJECTIVE

The core objective of V2D includes:

- *Splitting a video into two components, audio and the video frames.*
- *Both components are then processed using Vū Digital's speech-to-text transcription and MapOptis proprietary platform for text extraction from images, facial recognition and image recognition*
- *The output is not only the transcript of the video and image frames, but metadata that is time-stamped with frame references*
- *Using MapOptis platform to build a reference database*

For example, if a celebrity is found an hour and 17 minutes into a video, the metadata would include that time reference a 01:17. With this technological approach video classification/clustering, search engine indexing, and personalization for content, including targeted advertisements are possible.

1.3 DEFINITIONS, ACRONYMS AND ABBREVIATIONS

#1 *This section shall define all terms, acronyms and abbreviations used in this document. Particular care should be taken to define terms that are specific to the application.*

#2 *The following is a list of definitions for this document based on the approach to system design:*

Class Diagram	*Describes the structure of a system*
Object Diagram	*Expresses possible object combinations of a specific Class Diagram*
Statechart Diagram	*Expresses possible states of a class (or a system)*
Activity Diagram	*Describes activities and actions taking place in a system*
Sequence Diagram	*Shows one or several sequences of messages sent among a set of objects*
Collaboration Diagram	*Describes a complete collaboration among a set of objects*
Use-case Diagrams	*Illustrates the relationships between use cases*
Component Diagram	*A special case of a Class Diagram used to describe components within a software system*
Deployment Diagram	*A special case of a Class Diagram used to describe hardware within the overall system architecture*
System Block diagram	*A diagram showing the major components of the system with its interconnections and external interfaces*

1.4 REFERENCES

#1 *This section shall list all the applicable and reference documents, identified by title, author and date. Each document shall be marked as applicable or reference. If appropriate, report number, journal name and publishing organisation shall be included.*

#2 *The Vu Digital's* Architecture Guidelines *which shall provide the starting point for the system architecture, shall be referenced. The elements of the recommended architecture which are used, shall be described.*

#3 *There shall also be a reference to the document, which states how changes to this document are controlled. It is recommended that one of the references highlight the system development model that is to be followed in developing the system, such as AGILE methodology.*

Num.	Title (Applicability & Reference)	Author	Date	Issue

1.5 OVERVIEW

Section 1 is the introduction and includes a description of the project, applicable and reference documents.

Section 2 provides a system overview.

Section 3 contains the system context.

Section 4 describes the system design method, standards and conventions.

Section 5 contains the component descriptions.

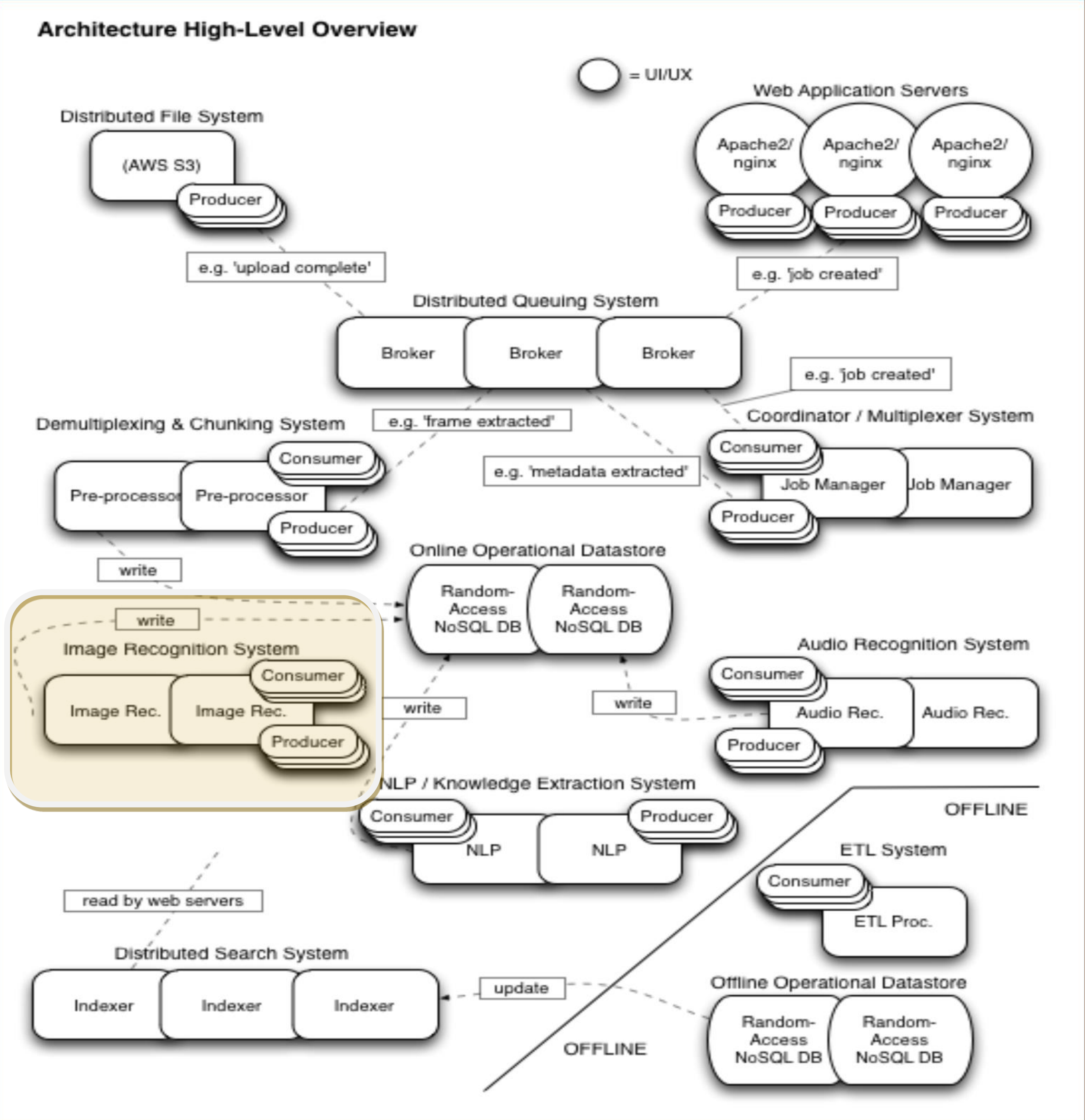
Section 6 includes the Requirements Traceability Matrix.

2 SYSTEM OVERVIEW

1. *Build an enterprise platform that will enable a user to indicate a 'source' and 'target' for video input and post-processed video output.*
2. *Transcribe the audio from the source video into text.*
    - *(see Automated Speech Recognition in "Business Requirements and Project Documentation" provided by Vū)*
3. *Break the video into still frames and detect the objects on each frame of the video, and using a network of image references, determine what that object is.*
    - *(see Image Processing in in "Business Requirements and Project Documentation" provided by Vū)*
4. *Distribute all of the processing in the audio and video processing so that the transcription and object recognition can occur in chunks and run simultaneously so that the overall processing time is a fraction of the video play time.*
    - *(see Distributed Processing in "Business Requirements and Project Documentation" provided by Vū)*
5. *Create a set of organized metadata for each video that can be analyzed by the video owner*
    a. *Video output specified as the 'target' would be*
        i. *audio transcripts with times tamped references to the original video,*
        ii. *NLP keywords over the entire video*
        iii. *frame-by-frame keywords with times-tamped references to the original video*
6. *Create a user search experience where the user can upload video and search through their video library for metadata reference*
    - *(see User Experience "Business Requirements and Project Documentation" provided by Vū)*

2.1 SYSTEM DESIGN AND ARCHITECTURE

The following depicts preliminary product architecture as defined in the "Architecture for Vū Digital Video to Data Service" document created by Vū. Please refer to the document for a detailed description of its recommended components.


Architecture High-Level Overview
= UI/UX
Distributed File System
(AWS S3)
Producer
Web Application Servers
Apache2/ nginx
Apache2/ nginx
Apache2/ nginx
Producer
Producer
Producer
e.g. 'upload complete'
e.g. 'job created'
Distributed Queuing System
Broker
Broker
Broker
e.g. 'job created'
Coordinator / Multiplexer System
Demultiplexing & Chunking System
e.g. 'frame extracted'
Consumer
Pre-processor
Pre-processor
Producer
e.g. 'metadata extracted'
Consumer
Job Manager
Job Manager
Producer
write
Online Operational Datastore
Random-Access NoSQL DB
Random-Access NoSQL DB
write
Image Recognition System
Consumer
Image Rec.
Image Rec.
Producer
write
write
Audio Recognition System
Consumer
Audio Rec.
Audio Rec.
Producer
NLP / Knowledge Extraction System
Consumer
NLP
NLP
Producer
OFFLINE
ETL System
Consumer
ETL Proc.
read by web servers
Distributed Search System
Indexer
Indexer
Indexer
update
Offline Operational Datastore
Random-Access NoSQL DB
Random-Access NoSQL DB
OFFLINE

Expert Sub System: (Provided by Vu Digital)

The Video to Data Service (V2D) takes video files and converts them into metadata that includes keywords and context knowledge. To achieve generate the metadata several expert sub-systems operate on the video:

- *A Face recognition / object identification Sub system that takes frames from the video, isolates and identifies the objects present in the frame.*
- *An audio recognition Sub system that includes speech recognition, sound identification and music identification.*
- *A natural language processing (NLP) Sub system that annotates and assigns meaning to the keywords generated by the image and audio sub-systems.*

Operational Sub-Systems

- *To support the metadata generation process, the following systems are also recommended:*
- *A web application that provides clients the ability to use the service*
  - *e.g. upload, monitor progress and receive outcomes.*
- *A client and administrative database to store user information and preferences*
- *A data storage Sub system to store client videos.*
- *A video preprocessing subsystem that transforms the video file into data on which the metadata generating sub systems operate.*
- *An auditing / coordination Sub system to monitor overall system performance and generate operational and business analytic data.*
- *An operational data storage Sub system into which the generated metadata as well as operational and business analytic data will be stored for use in active, online processes.*
- *An offline data storage system that holds the history of all operations performed in the system including business and operational data.*
- *An ETL process that regularly writes to the offline data storage system.*
- *A search Sub system that indexes client results and makes them searchable via the web application.*

Support Sub-Systems

- *To support the operation of the overall system the following are additionally recommended:*

- *An administrative / operational web application that includes the capability to perform admin tasks, monitor system (or sub-system) performance, monitor customer activity, create and report on system and business metrics.*

- *Deployment and Sub system operational tools including: new deployments, system management and zero—downtime (i.e. Polling) deployments.*

Performance & Scalability

- *Beyond the functional and operational requirements, the system has the following performance requirements:*
- *Latency with regard to metadata generation must be minimal. Specific metrics are TBD.*
- *The system must remain available in the face of hardware failures (including drives, machines, network equipment).*
- *The system (and its sub-systems) must be able to scale horizontally with near zero operational impact and commodity hardware.*

Recommendations

- *To facilitate these system objectives, an architecture based on distributed message queuing and distributed (NoSQL) data storage is recommended. Distributed message queuing systems are based on a producer--- consumer model where one or more components 'produce' data that gets delivered to a particular queue at which time it gets 'consumed' by a component that watches that queue.*
- *In these systems data can be thought of as being 'pulled' through the system as it completed each phase of processing. This is in contrast to 'push' systems that process events in a synchronous, linear execution.*
- *The advantages to using distributed message queuing systems for building scalable systems includes:*
- *Decoupling of processes, allowing independent extension and modification.*
  - *Scalability by process*
  - *Resources can be optimally allocated for performance.*
  - *Failure resiliency*
  - *When a process fails, another process can pick it up.*
  - *Overload resiliency*
- *In a spike of requests, the queues will spool allowing the system to continue operating at capacity.*
- *Delivery and order guarantees – many messaging systems include features that guarantee a message gets processed off a queue and / or that messages are processes in a particular order.*
- *Greater visibility on data flow and processing performance*
- *The decoupled nature of message queues simplifies performance monitoring and optimization. The disadvantages are primarily increased overhead and increased system complexity. In practice, at scale the increased overhead is more than offset by the ability to handle greater numbers of requests. Similarly, the complexity of a distributed message queue tends to be simpler at scale than attempting to scale traditional linear-synchronous systems. Given the volume, scalability requirements and types of data (in particular, binary data and widely varying, non---linearly associative mappings of videos to metadata) the use of traditional relational*

databases is not recommended for the core operational data store as they tend to degrade under high volumes, are difficult and/or expensive to scale and restricted to relational data modeling (which isn't appropriate here). In stead, a random--access distributed NoSQL database is recommended. A high--level depiction of the architecture can be seen in the figure below.

- *Note that not all sub--systems are represented, nor are all system / Sub system operations.*

Image Recognition System :

Image Recognition System has two principal components

- *Detection*
- *Recognition*

Further the images are classified as Faces and Objects. Typically an image can contain Faces and Objects as well.

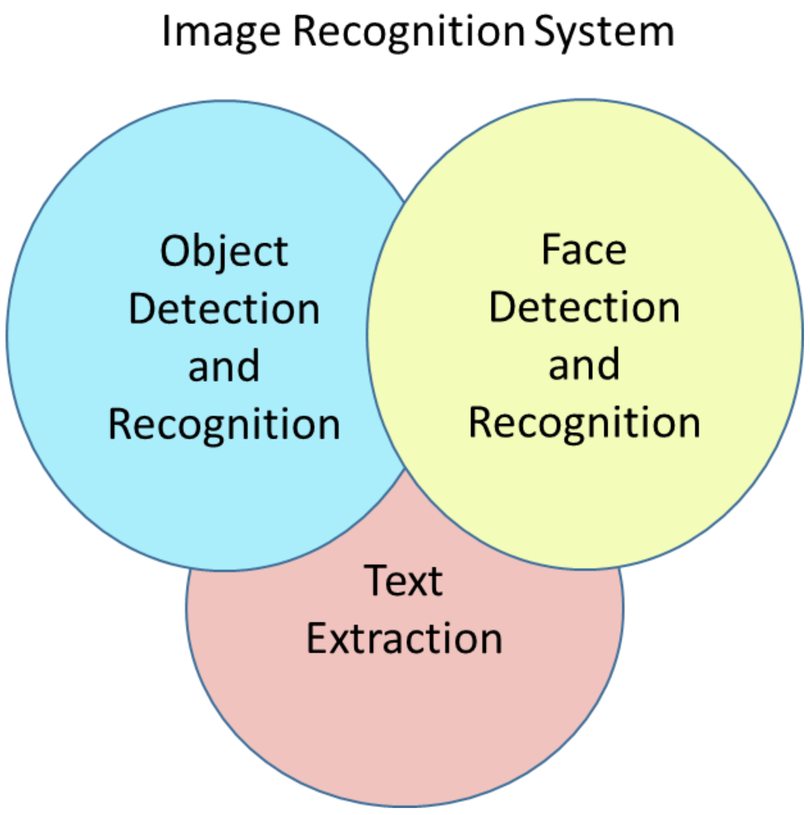
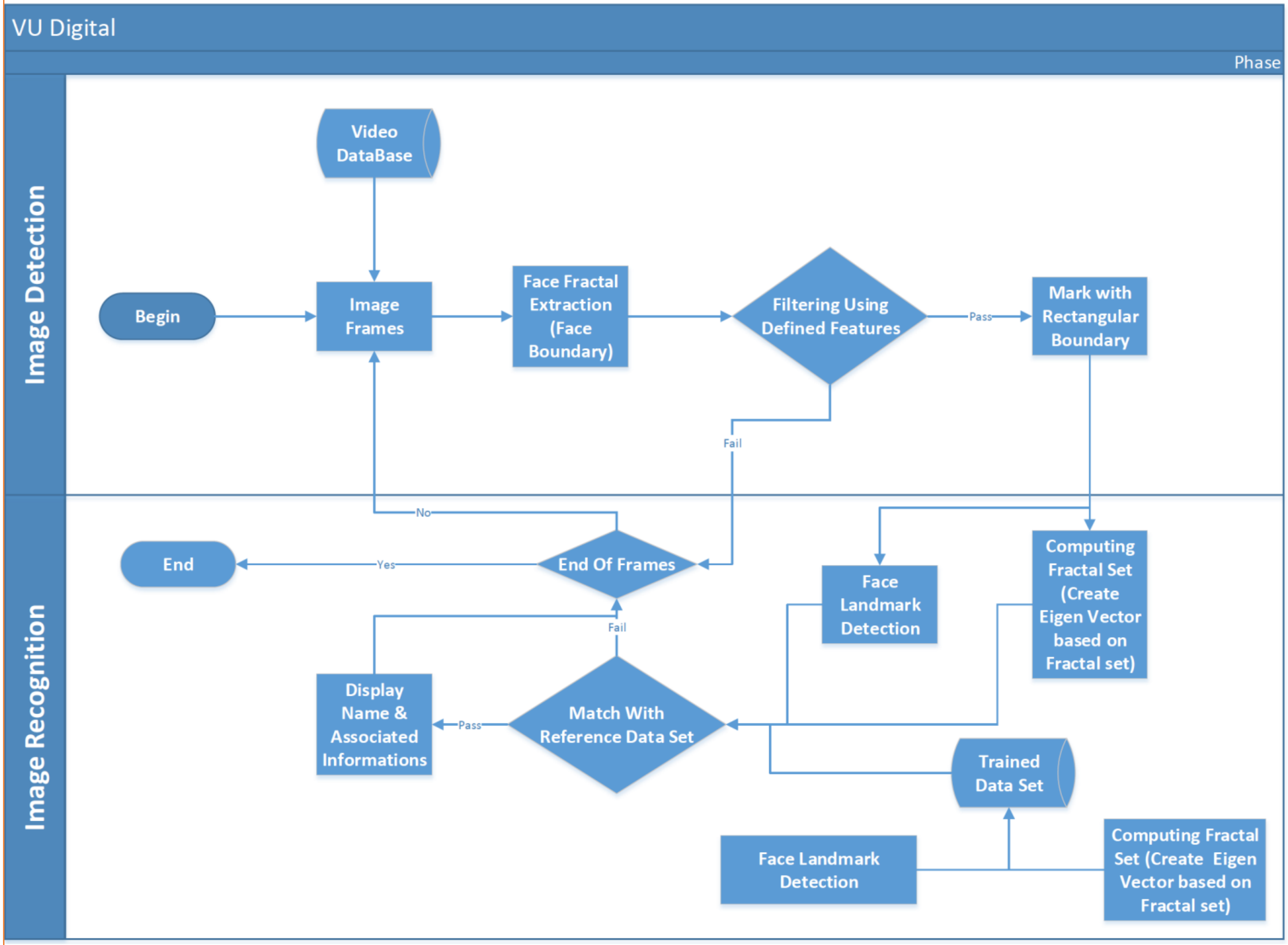


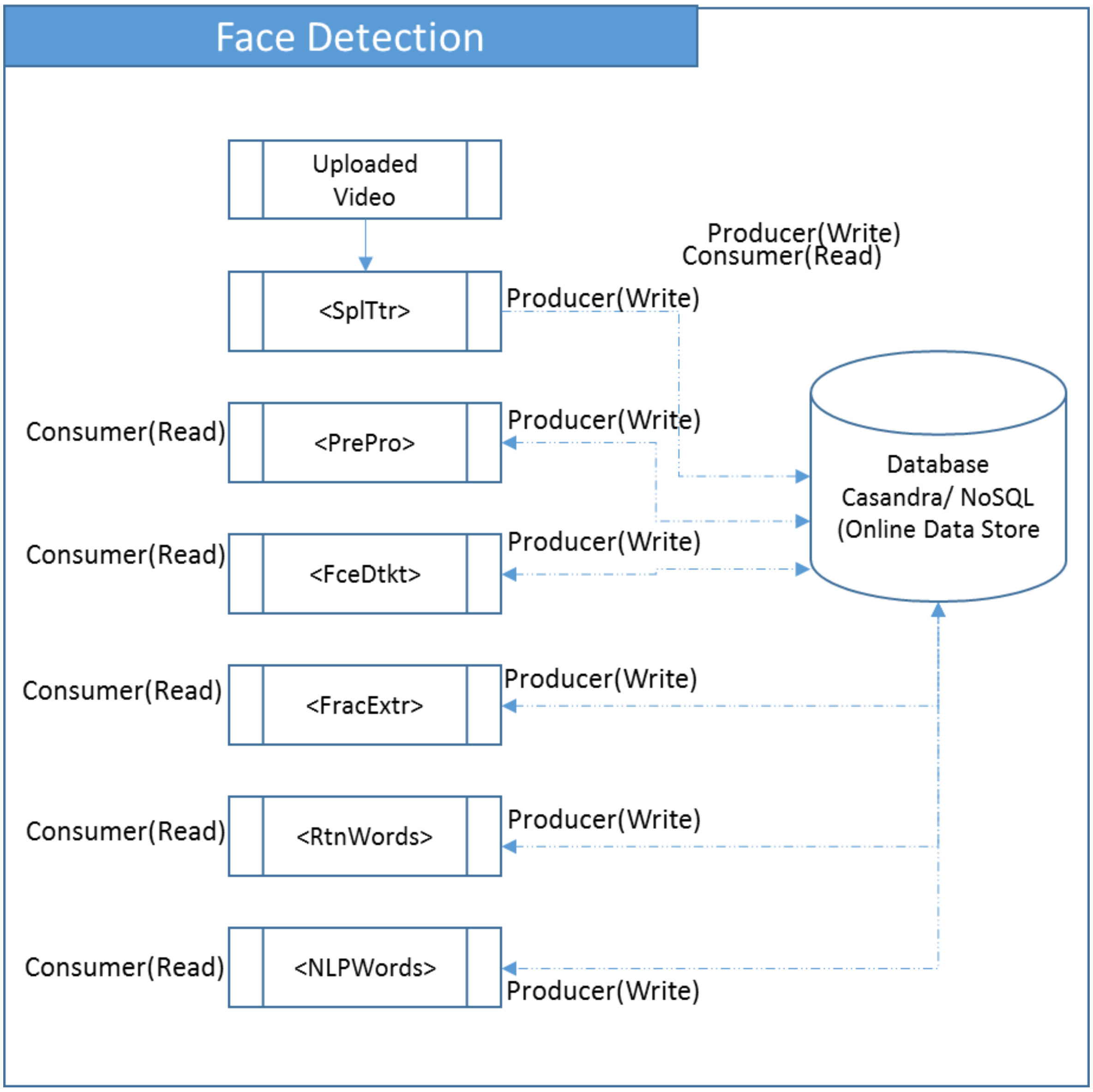
Image Recognition System
Object Detection and Recognition
Face Detection and Recognition
Text Extraction

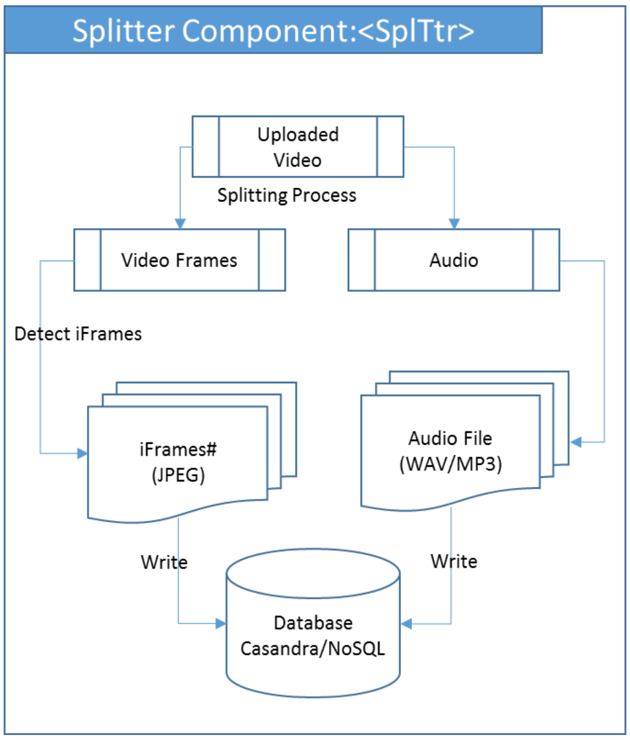
Architecture: Block Diagram

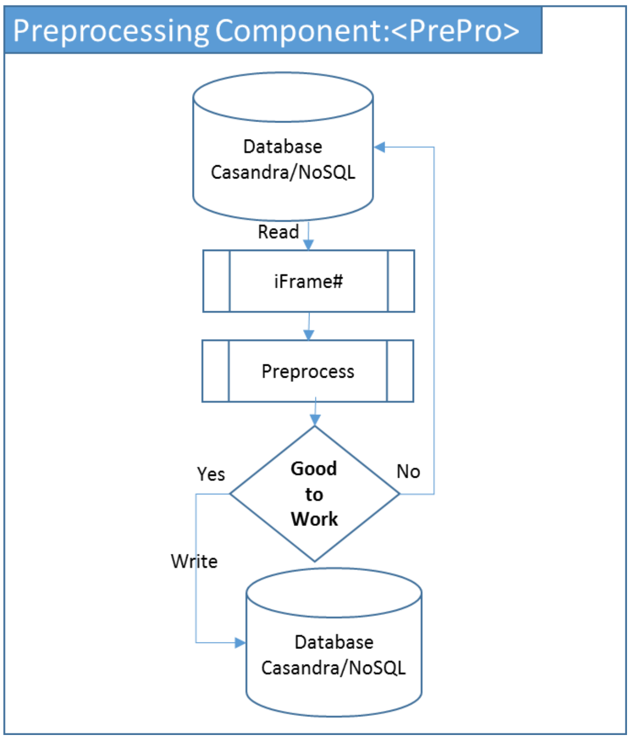

VU Digital
Phase
Image Detection
Image Recognition
Video DataBase
Begin
Image Frames
Face Fractal Extraction (Face Boundary)
Filtering Using Defined Features
Pass
Mark with Rectangular Boundary
Fail
No
End
Yes
End Of Frames
Face Landmark Detection
Computing Fractal Set (Create Eigen Vector based on Fractal set)
Fail
Display Name & Associated Informations
Pass
Match With Reference Data Set
Trained Data Set
Face Landmark Detection
Computing Fractal Set (Create Eigen Vector based on Fractal set)

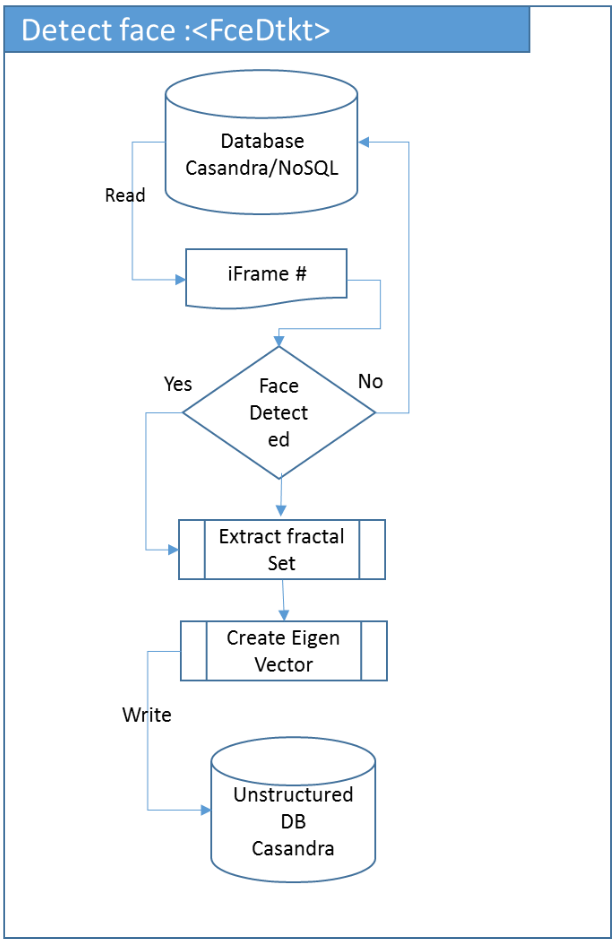
Architecture: Component Level view (Sub system)

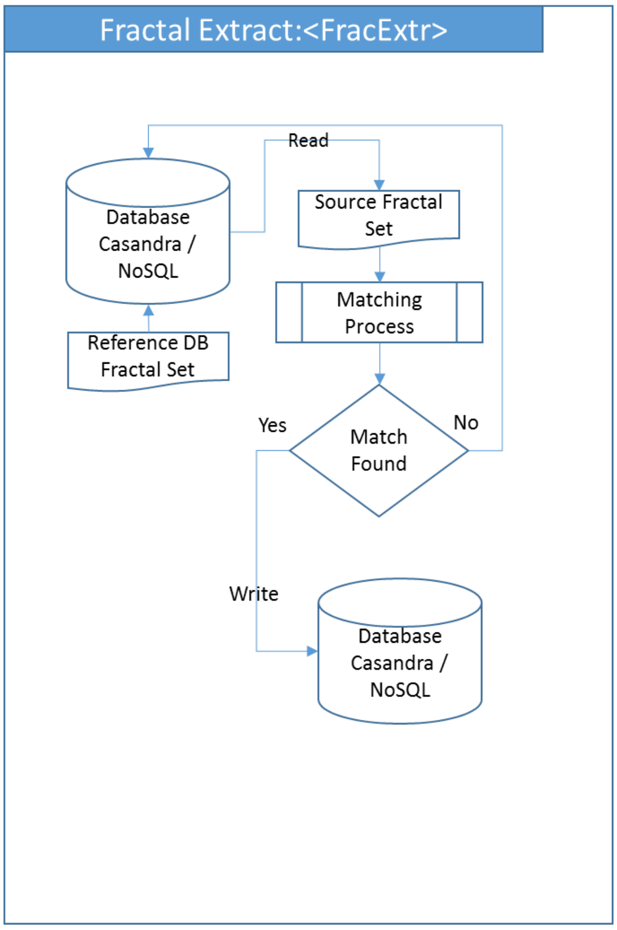
In order to segment various tasks of Face recognition framework and prioritize to work in distributed computing platform following method is chosen. Dotted line indicate simultaneous read/write scenarios.

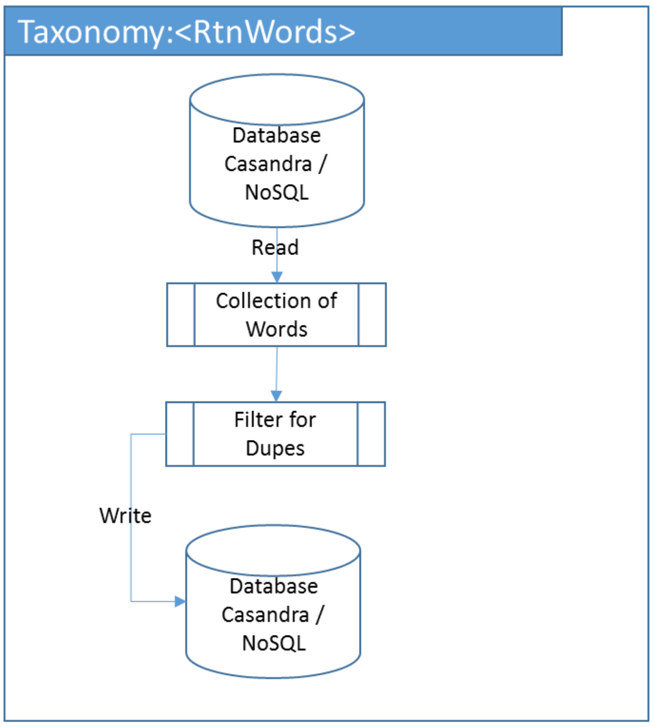


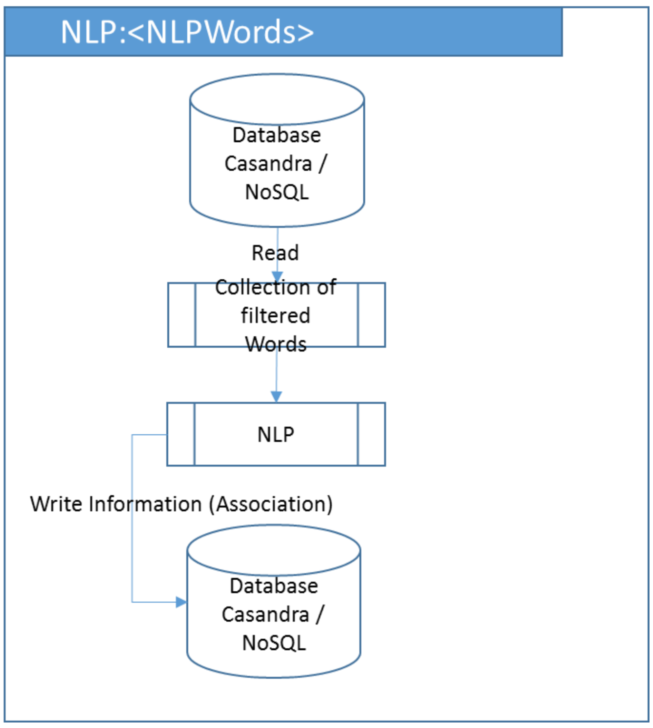

Splitter Component:<SplTtr>
Uploaded Video
Splitting Process
Video Frames
Audio
Detect iFrames
iFrames# (JPEG)
Audio File (WAV/MP3)
Write
Write
Database Casandra/NoSQL


Preprocessing Component:<PrePro>
Database Casandra/NoSQL
Read
iFrame#
Preprocess
Yes
Good to Work
No
Write
Database Casandra/NoSQL


Detect face :<FceDtkt>
Database Casandra/NoSQL
Read
iFrame #
Yes
Face Detect ed
No
Extract fractal Set
Create Eigen Vector
Write
Unstructured DB Casandra

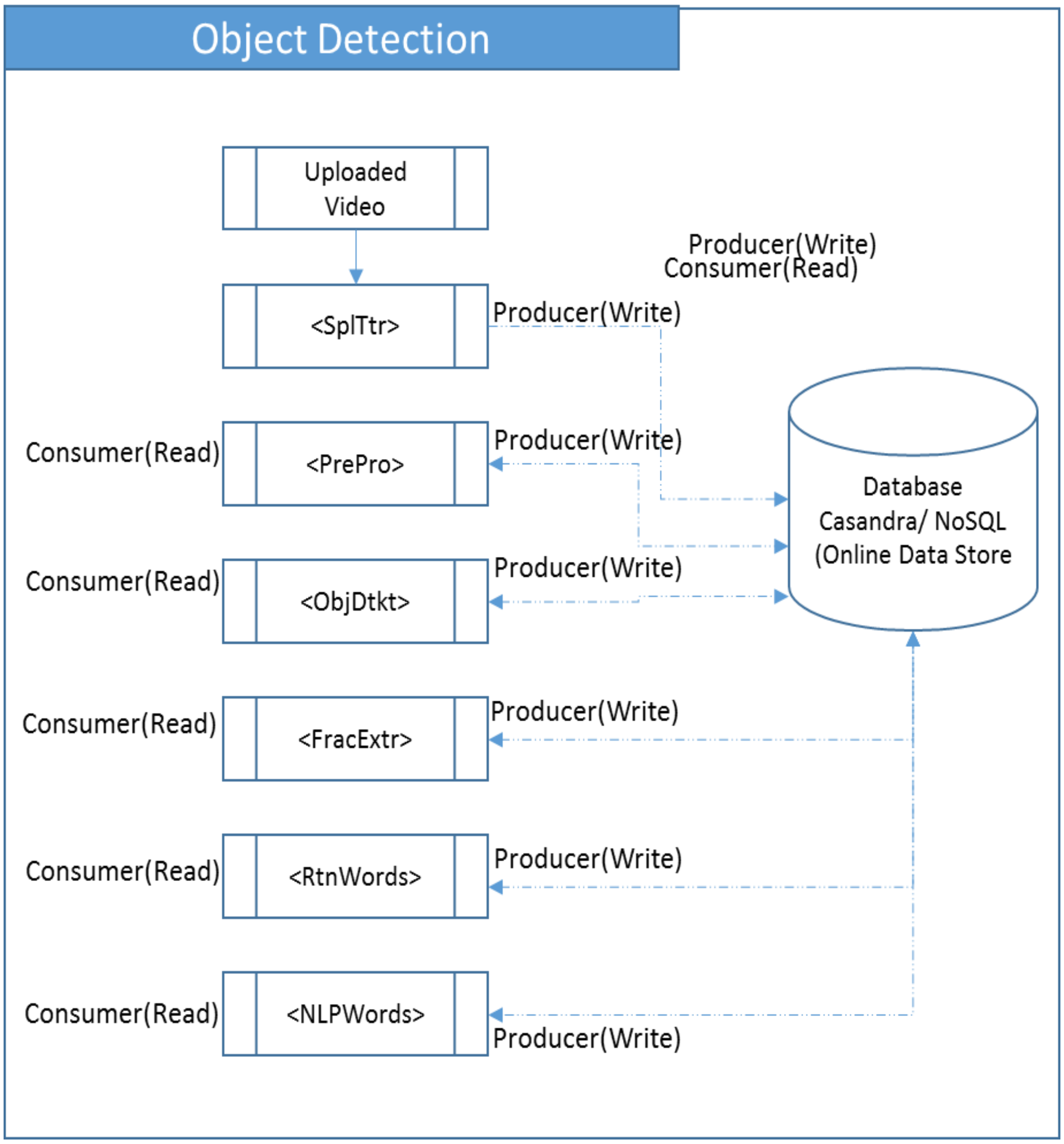

Fractal Extract:<FracExtr>
Read
Database Casandra / NoSQL
Source Fractal Set
Matching Process
Reference DB Fractal Set
Yes
Match Found
No
Write
Database Casandra / NoSQL

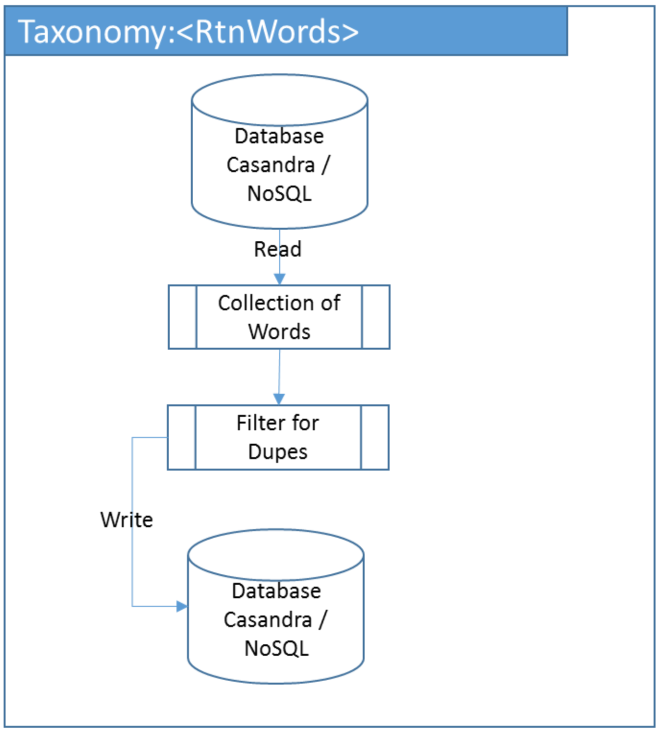

Taxonomy:<RtnWords>
Database Casandra / NoSQL
Read
Collection of Words
Filter for Dupes
Write
Database Casandra / NoSQL

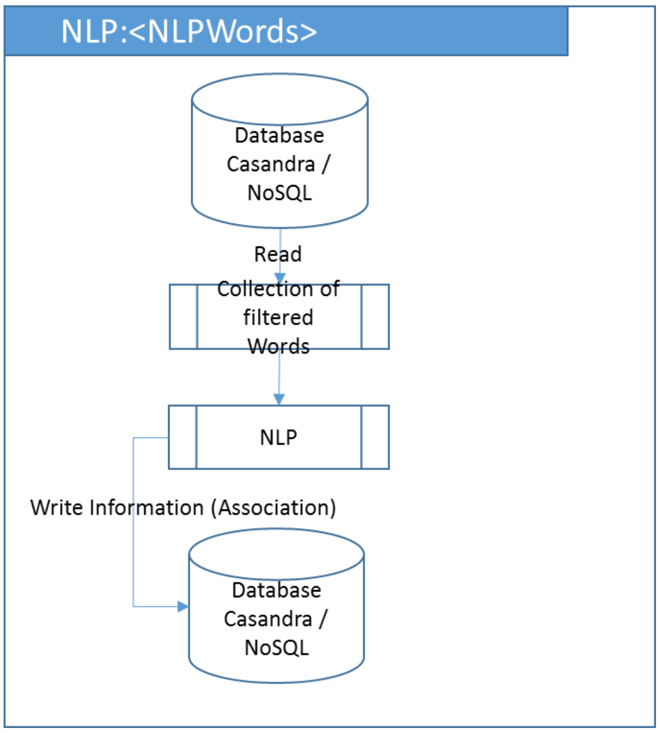

NLP:<NLPWords>
Database Casandra / NoSQL
Read
Collection of filtered Words
NLP
Write Information (Association)
Database Casandra / NoSQL

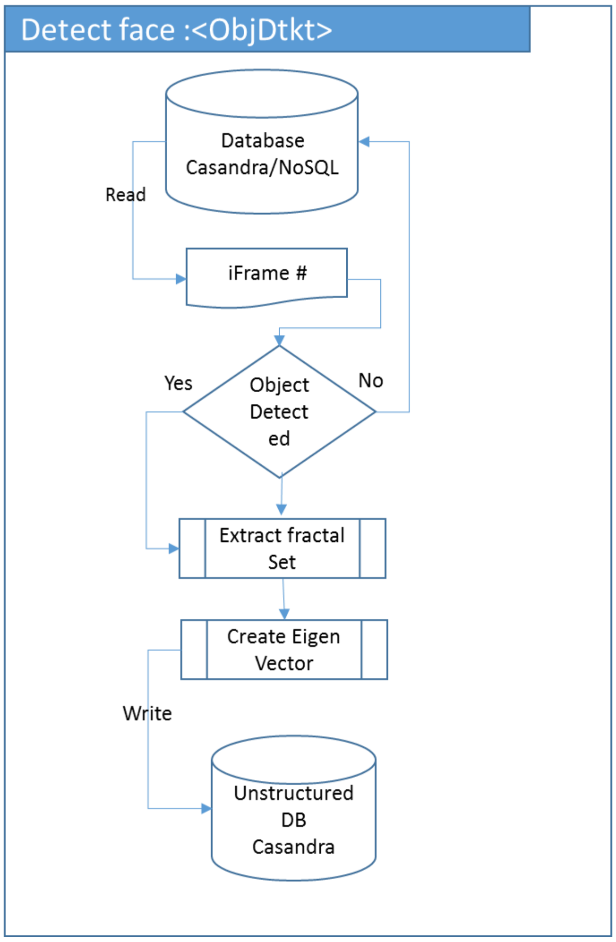
Object Detection and Recognition

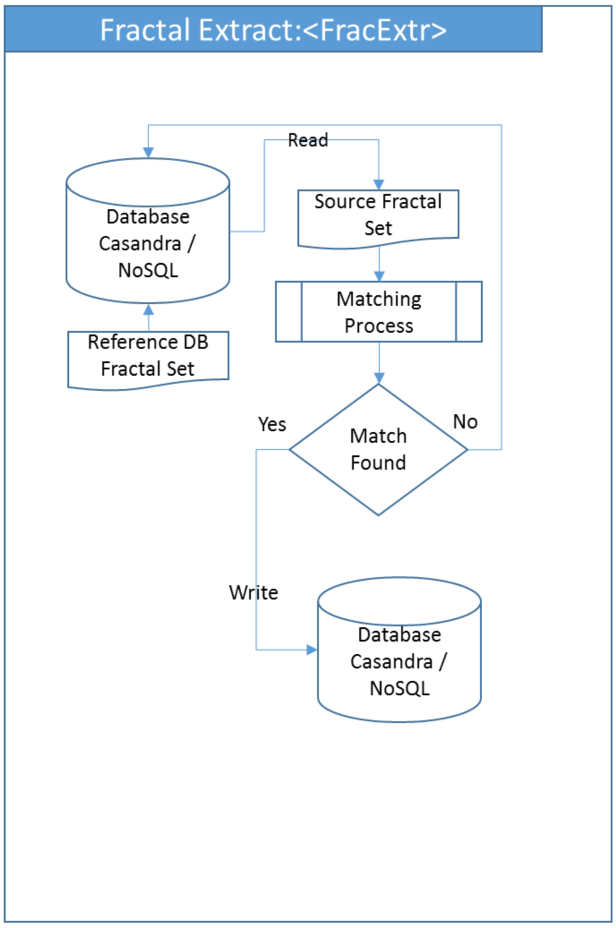
In order to segment various tasks of Object recognition framework and prioritize to work in distributed computing platform following method is chosen. Dotted line indicate simultaneous read/write scenarios

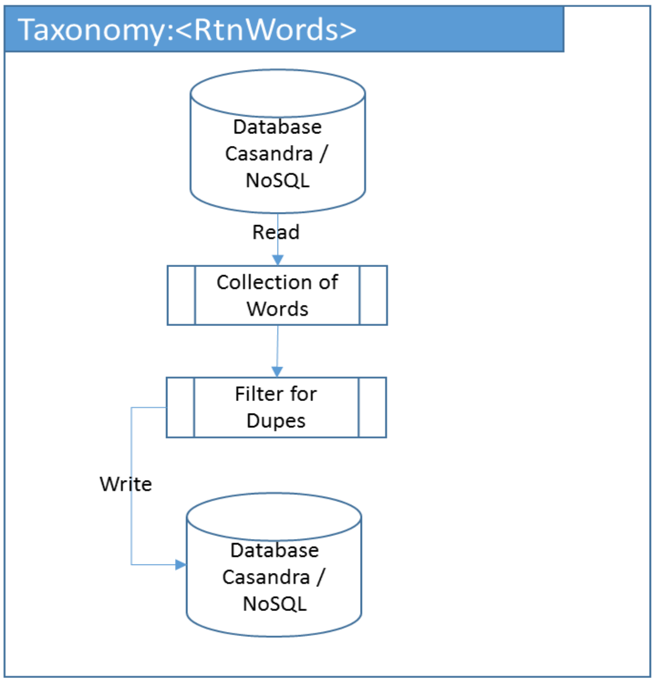


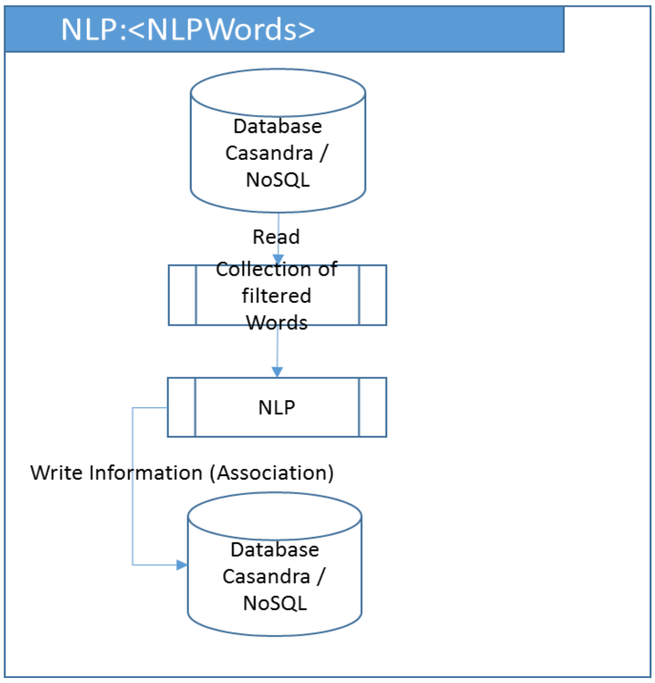

Taxonomy:<RtnWords>
Database Casandra / NoSQL
Read
Collection of Words
Filter for Dupes
Write
Database Casandra / NoSQL


NLP:<NLPWords>
Database Casandra / NoSQL
Read
Collection of filtered Words
NLP
Write Information (Association)
Database Casandra / NoSQL


Detect face :<ObjDtkt>
Database Casandra/NoSQL
Read
iFrame #
Yes
Object Detected
No
Extract fractal Set
Create Eigen Vector
Write
Unstructured DB Casandra

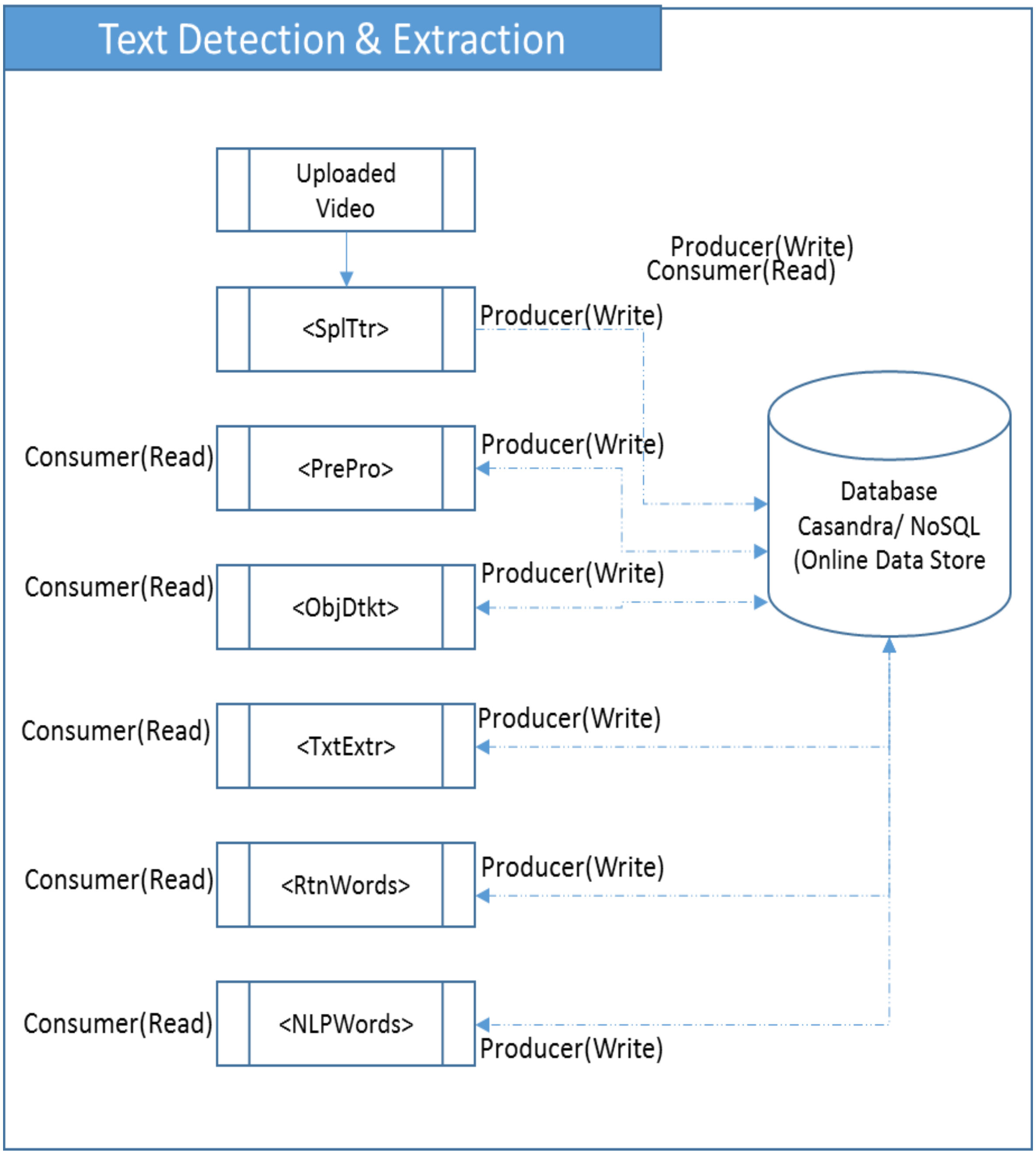

Fractal Extract:<FracExtr>
Read
Database Casandra / NoSQL
Source Fractal Set
Matching Process
Reference DB Fractal Set
Yes
Match Found
No
Write
Database Casandra / NoSQL

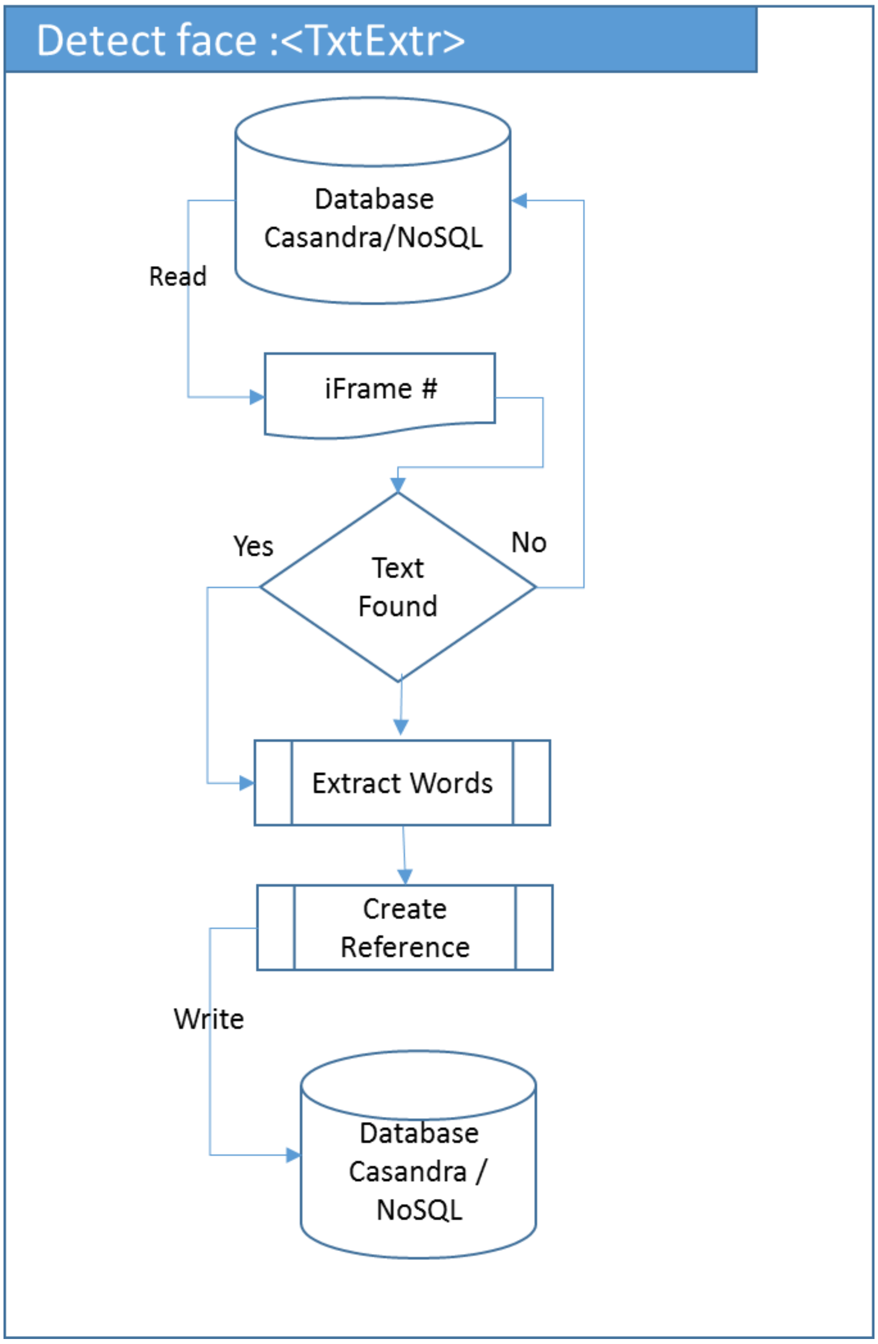

Taxonomy:<RtnWords>
Database Casandra / NoSQL
Read
Collection of Words
Filter for Dupes
Write
Database Casandra / NoSQL

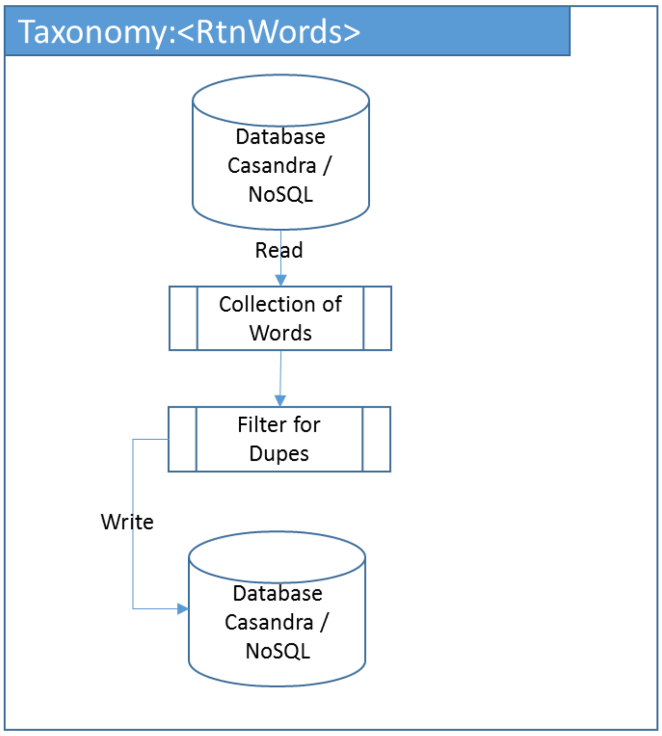

NLP:<NLPWords>
Database Casandra / NoSQL
Read
Collection of filtered Words
NLP
Write Information (Association)
Database Casandra / NoSQL

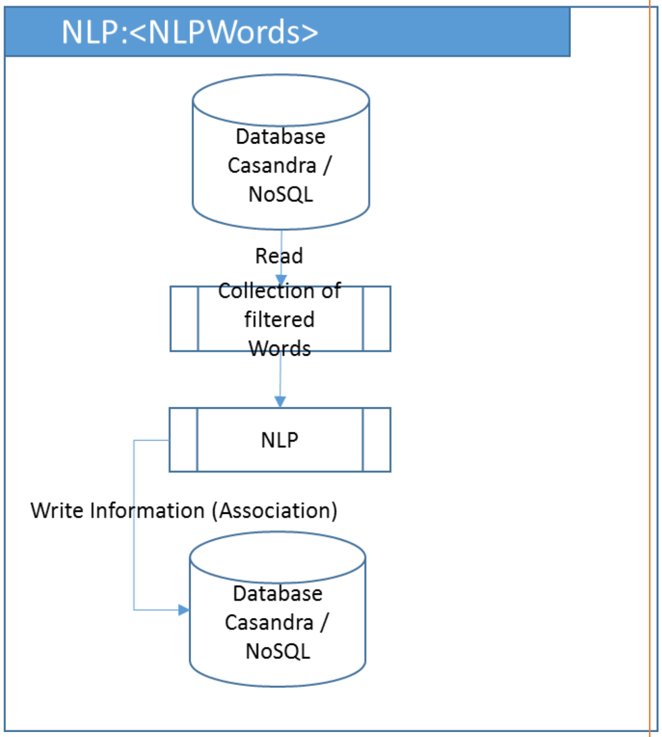
Text Detection and Extraction

In order to segment various tasks of Text extraction module and prioritize to work in distributed computing platform following method is chosen. Dotted line indicate simultaneous read/write scenarios




Detect face :<TxtExtr>
Database Casandra/NoSQL
Read
iFrame #
Yes
No
Text Found
Extract Words
Create Reference
Write
Database Casandra / NoSQL




NLP:<NLPWords>
Database Casandra / NoSQL
Read
Collection of filtered Words
NLP
Write Information (Association)
Database Casandra / NoSQL

#1 *Given this likely wide range of means of access, the current emphasis in cloud based distributed architecture of breaking down the components into several layers, will be factors that will have to be addressed in the course of development. The architecture proposed will provide the flexibility to support ever-changing business. Its key feature is that it separates out business logic, client access technology and centrally held data into discrete layers with standard, open interfaces.*

#2 *Depending upon the solution proposed for the system in question there may be a balance of functions placed into each of these layers. We believe that in the design process it is vital, however, to have a layered architecture as the basis of the solution to facilitate change and adaptation of the system in the future – in other words to protect the investment that has been made in the system as technologies develop in the future. The degree of isolation afforded to elements of the system through adopting a layered approach is significant and can help immunise the project from having to be replaced in full in years to come – some key elements will survive technological developments.*

#3 *Enlargement, both in the medium and long term, could be a crucial factor in the design of V2D systems. Scalability of the proposed architecture may not pose an issue. With a layered approach the ability to scale to meet future requirements is relatively easy as additional hardware infrastructure can be deployed into the various layers of the system.*

3 SYSTEM CONTEXT

Facial recognition algorithms identify facial fractals by extracting landmarks from an image of the subject's face. For example, the algorithm may analyze the relative position, size, and/or shape of the eyes, nose, cheekbones, and jaw. These features are then used to search for other images with matching features. Other algorithms normalize a gallery of face images and then compress the face data, only saving the fractal data in the image that is useful for face recognition. A probe image is then compared with the face data. Recognition algorithms can be divided into two main approaches, geometric, which looks at distinguishing features, or photometric, which is a statistical approach that distills an image into values and compares the values with templates to eliminate variances.

Popular recognition algorithms include Principal Component Analysis using Eigen faces, Linear Discriminate Analysis, Elastic Bunch Graph Matching using the Fisherface algorithm, the Hidden Markov model, the Multi-linear Subspace Learning using tensor representation, and the neuronal motivated dynamic link matching.

On a careful consideration, a hybrid using fractal genesis is being constructed to detect the face with elements described above.

An alternate approach is three-dimensional face recognition. This technique uses 3D sensors to capture fractal information about the shape of a face. This information is then used to identify distinctive features on the surface of a face, such as the contour of the eye sockets, nose, and chin.

One advantage of 3D facial recognition is that it is not affected by changes in lighting like other techniques. It can also identify a face from a range of viewing angles, including a profile view. Three-dimensional data points from a face vastly improve the precision of facial recognition.

To improve the accuracy of detection, the hybrid also uses the visual details of the skin, as captured in standard digital or scanned images. This technique, called skin texture analysis, turns the unique lines, patterns, and spots apparent in a person's skin into a mathematical fractal space. Tests have shown that with the addition of skin texture analysis, performance in recognizing faces can increase 20 to 25 percent

PCA : Derived from Karhunen-Loeve's transformation. Given an s-dimensional vector representation of each face in a training set of images, Principal Component Analysis (PCA) tends to find a t-dimensional subspace whose basis vectors correspond to the maximum variance direction in the original image space. This new subspace is normally lower dimensional (t<<s). If the image elements are considered as random variables, the PCA basis vectors are defined as eigenvectors of the scatter matrix.

Linear Discriminant Analysis (LDA) finds the vectors in the underlying space that best discriminate among classes. For all samples of all classes the between-class scatter matrix SB *and the within-class scatter matrix* SW *are defined. The goal is to maximize* SB *while minimizing* SW, *in other words, maximize the ratio det*|SB|/*det*|SW| . *This ratio is maximized when the column vectors of the projection matrix are the eigenvectors of (*SW^-1 × SB*).*

Aa eigenspace-based adaptive approach that searches for the best set of projection axes in order to maximize a fitness function, measuring at the same time the classification accuracy and generalization ability of the system. Because the dimension of the *solution space of this problem is too big, it is solved using a specific kind of genetic algorithm called Evolutionary Pursuit (EP).*

Elastic Bunch Graph Matching (EBGM). All human faces share a similar topological structure. Faces are represented as graphs, with nodes positioned at fiducial points. (exes, nose...) and edges labeled with 2-D distance vectors. Each node contains a set of 40 complex Gabor wavelet coefficients at different scales and orientations (phase, amplitude). They are called "jets". Recognition is based on labeled graphs. A labeled graph is a set of nodes connected by edges, nodes are labeled with jets, edges are labeled with distances.

Kernel Methods : The face manifold in subspace need not be linear. Kernel methods are a generalization of linear methods. Direct non-linear manifold schemes are explored to learn this non-linear manifold.

Trace transform: A generalization of the Radom transform, is a new tool for image processing which can be used for recognizing objects under transformations, e.g. rotation, translation and scaling. To produce the Trace transform one computes a functional along tracing lines of an image. Different Trace transforms can be produced from an image using different trace functional.

3-D Morphable Model: Human face is a surface lying in the 3-D space intrinsically. Therefore the 3-D model should be better for representing faces, especially to handle facial variations, such as pose, illumination etc. Blantz et al. proposed a method based on a 3-D morphable face model that encodes shape and texture in terms of model parameters, and algorithm that recovers these parameters from a single image of a face.

Bayesian Framework: A probabilistic similarity measure based on Bayesian belief that the image intensity differences are characteristic of typical variations in appearance of an individual. Two classes of facial image variations are defined: intrapersonal variations and extrapersonal variations. Similarity among faces is measures using Bayesian rule.

Hidden Markov Models (HMM): These are a set of statistical models used to characterize the statistical properties of a signal. HMM consists of two interrelated processes: (1) an underlying, unobservable Markov chain with a finite number of states, a state transition probability matrix and an initial state probability distribution and (2) a set of probability density functions associated with each state

Fractal Genesis(Hybrid): The image intensity differences are characteristic of typical variations in appearance of an individual. Human face is a surface lying in the Fractal space intrinsically. Since most of the parameters are self-similar, the Fractal model should be better for representing faces, especially to handle facial variations, such as pose, illumination etc.

A Hybrid, wherein many proponents of other algorithms are integrated to form a fractal genesis.

There are several components which are being used to detect the face and recognize it with the reference database built for the purpose.

3.1 PROGRAMMING STANDARDS

#1 *The core algorithms are being written in C and are tweaked and modularized to handle the proposed distributed computing architecture.*

#2 *All the interfaces shall also be written in C/Java.*

#3 *We are evaluating the adoption of unstructured database Casandra / NoSQL*

#4 *Standard naming conventions shall be followed while writing the code.*

#5 *A structured library shall be maintained to host the components for various tasks mentioned in the document. Where there are external interfaces, the programming standards for the interfaces required shall be referenced.*

#6 *In general, the programming standard shall be followed with an uniform programming style. Specific points to mention are:*

a. modularity and structuring;

b. headers and commenting;

c. indenting and layout;

d. library routines to be used;

e. language constructs to use;

3.2 SOFTWARE DEVELOPMENT TOOLS

#1 *The list may include:*

a. an Mathematical laboratory to test the algorithms;

b. a configuration manager / builder;

c. HTML/UML/XML authoring tools;

d. a word processor for documentation;

e. a tool for drawing diagrams;

f. Automated testing tools.

#2 *External interfaces may require some of the modules to be pre-compiled.*

3.3 OUTSTANDING ISSUES

#1 *This document is a draft and will undergo changes during the journey of development. Provide details of any design issues that remain unresolved at the date of issue of this document. Explain options, pros and cons, and give an estimate of which option is most likely. Outline impact of each option on the rest of the design.*

3.4 DECOMPOSITION LIST

Following are the major decomposition elements (Not exhaustive):

- *Video Splitter – VSplTtr*
- *Audio Splitter – AsplTtr*
- *iFrame Grabber – FrmGrab*
- *Pre Processing Routine – PrePro*
- *Face Detection Routine – FceDtkt*
- *Object Detection Routine – ObjDtkt*
- *Text Extraction Routine – TxtExtr*
- *Grab Words – RtnWords*
- *NLP – NLPWords*
- *Sizing the video – Sizer*
- *Database Orchestrator – DBOrchestrate*
- *Error handler – ErrorHander*
- *Knowledgebase Builder – KnowBuild*
- *Refernce DB builder – RefDBuilder*
- *Repeat Checker – RepCheck*

4 COMPONENT DESCRIPTION

#2 *The descriptions of the components shall be laid out hierarchically. There shall be subsections dealing with the following aspects of each component:*

- *4.n Component identifier*
- *4.n.1 Type*
- *4.n.2 Purpose*
- *4.n.3 Function*
- *4.n.4 Subordinates*
- *4.n.5 Dependencies*
- *4.n.6 Interfaces*
- *4.n.7 Resources*
- *4.n.8 References*
- *4.n.9 Processing*
- *4.n.10 Data*

#3 *The number 'n' shall relate to the place of the component in the hierarchy.*

4.1 COMPONENT IDENTIFIER

#1 *Each component shall have a unique identifier. The identifiers to be used for components shall be defined by the project and described elsewhere.*

4.1.1 Type

#1 *This section shall describe the type of component, e.g. task, subroutine, subprogram, package, file.*

#2 *The contents of some component description sections depend on the component type. For the purpose of this document the categories: executable, i.e. contains computer instructions, or non-executable, i.e. contains only data, are used.*

4.1.2 Purpose

#1 *The purpose of a component shall be defined by tracing it to the software requirements that it implements.*

#2 *Backwards traceability depends upon each component description explicitly referencing the requirements that justify its existence.*

4.1.3 Function

#1 *The function of a component shall be defined in this document. This shall be a short description of what the component does and will depend upon the component type e.g. it may be a description of the process or of the data to be stored or transmitted.*

#2 *More details will be provided in Processing*

4.1.4 Subordinates

#1 *This section shall list the modules that are 'called by' this component. The subordinates of a database could be the files that 'compose' it. The subordinates of an object are the objects that are 'used by' it.*

4.1.5 Dependencies

#1 *The dependencies of a component shall be defined by listing the constraints placed upon its use by other components. For example:*

- *what operations have to have taken place before this component is called?*
- *what operations are excluded when this operation is taking place?*
- *what components have to be executed after this one?*

4.1.6 Interfaces

#1 *Both control flow and data flow aspects of an interface shall be specified. Data aspects of 'non-executable' components shall be defined in Subsection.*

#2 *The control flow to and from a component shall be defined in terms of how this component is executed, e.g. subroutine call, and how it is to be terminated, e.g. return.*

#3 *The data flow input to and output from the component shall be detailed. Data structures shall be identified that:*

a. are associated with the control flow, e.g. call argument list;

b. interface components through common data areas and files.

#4 *If a component interfaces to components in the same system then the interface description shall be defined. If a component interfaces to components in other systems, the interface description shall be defined in an interface document.*

4.1.7 Resources

#1 *The resources a component requires shall be defined by itemising what the component needs from its environment to perform its call. Items that are part of the component interface are excluded.*

4.1.8 Assumptions and Constraints:

#1 *The Reference image databases is the key to achieve the objectives and the scope mentioned in the proposal. Though iData Sciences cannot guarantee the quality of such images(being external), iData sciences will put in best efforts to recommend, purchase, classify and build the reference database.*

#2 *The input source video should be of good quality in order to recognize the objects, text and faces. Compressed videos do suffer from quality and the noises in such video will impact the output.*

#3 *We assume all the necessary hardware infrastructure to execute the said assignment already exists and available and if not Vū Digital will make it available for successfully executing the project.*

#4 *We assume all the software infrastructure like software monitoring tool, load balancing tool available to successfully execute the project.*

#5 *We also assume a staging site exists in order to demonstrate the tool in near real time before production and Vū Digital assumes the responsibility to make it available.*

#6 *We assume the distributed computing model exists and iData Sciences need only to follow instructions form Vū digital and implement.*

#7 *We assume the Audio to Text model exists and iData Sciences need only to implement such model.*

5 SOFTWARE REQUIREMENTS TRACEABILITY MATRIX

#1 This section shall contain a table that summarises how each software requirement has been met in this document. The tabular format permits one-to-one and one-to-many relationships to be shown.

System Req. Number	System Ref. Item	Component Identifier	Component Item

DOCUMENT CONTROL

Title:	*Technical Design Document*
Issue:	*Version 1.0*
Date:	*April 15, 2014*
Author:	*Venkatanathan Dwarakanathan*
Distribution:	*Vū Digital Project Team & iData Sciences*
Reference:	*IDS-VŪ-TD*
Filename:	*IDS-VU-TD-V1*
Control:	*Reissue as complete document only*

DOCUMENT SIGNOFF

Nature of Signoff	**Person**	**Signature**	**Date**	**Role**
Authors	Venkatanathan Dwarakanathan			Project Owner
Reviewers				

DOCUMENT CHANGE RECORD

Date	**Version**	**Author**	**Change Details**