ICT Seventh Framework Programme (ICT FP7)

Grant Agreement No: 288513

Policy Formulation and Validation through non Moderated Crowdsourcing

D2.3 Standards, Software Interface Modules and APIs for inter-platform communication in Web 2.0 Social Media

Deliverable Form	
Project Reference No.	ICT FP7 288513
Deliverable No.	D2.3
Relevant Workpackage:	WP2: User Requirements and Specification
Nature:	R
Dissemination Level:	PU
Document version:	Final V1
Date:	31/10/2012
Authors:	Evangelos Kanoulas (GOOGLE), Aggeliki Androutsopoulou, Leukothea Spiliotopoulou, Yannis Koulizakis, Costas Koutras, Yannis Charalabidis, Euripides Loukis (AEGEAN), Pythagoras Karampiperis, Vangelis Karkaletsis (NCSR'D')
Document description:	This deliverable provides a detailed analysis on the standards, interfaces and APIs of the Social Media platforms that form part of NOMAD sources.

Document History

Version	Date	Author (Partner)	Remarks
Draft v0.10	07/09/2012	Aggeliki Androutsopoulou (AEGEAN)	Initial ToC
Draft v0.20	17/09/2012	Aggeliki Androutsopoulou, Costas Koutras (AEGEAN) , Pythagoras Karampiperis (NSCR'D'), Evangelos Kanoulas (GOOGLE)	Updated ToC, Template for the API analysis, Initial Content
Draft v0.30	03/10/2012	Evangelos Kanoulas (GOOGLE)	Blogger and Google+ analysis
Draft v0.40	15/10/2012	Yannis Koulizakis (AEGEAN)	Facebook, Wordpress, Twitter, YouTube, LinkedIn analysis
Draft v0.50	22/10/2012	Leukothea Spiliotopoulou, Aggeliki Androutsopoulou, Yannis Koulizakis (AEGEAN)	SOTA (Section 5), Finalisation of Section 3
Draft v0.60	25/10/2012	Yannis Koulizakis (AEGEAN)	Definition of Semantic Interoperability
Draft v0.70	22/10/2012	Costas Koutras, Aggeliki Androutsopoulou, Yannis Koulizakis (AEGEAN)	1st Consolidated draft
Drat v0.80	24/10/2012	Pythagoras Karampiperis (NSCR'D')	Internal Review
Drat v0.85	25/10/2012	Aggeliki Androutsopoulou, Costas Koutras (AEGEAN)	Final Draft
Drat v0.90	28/10/2012	Antonis Koukourikos, Vangelis Karkaletsis (NSCR'D'), Anna Triantafillou (ATC)	Internal Review
Draft v0.95	29/10/2012	All partners	Final Review
Final v1.0	31/10/2012	Aggeliki Androutsopoulou, Costas Koutras , Yannis Charalabidis, Euripides Loukis (AEGEAN)	Document Finalisation

EXECUTIVE SUMMARY

The concepts of collaboration and crowdsourcing in Web 2.0 refer not only to content co-creation, but also to the implementation principles that enable mashing up different data and services. Web 2.0 applications are built on these basic technical principles, which include interoperability aspects, open standards and common formats. Following these principles, all Social Media platforms expose Application Programming Interfaces (APIs) to enable programmes develop applications that interoperate with them and authorise them to access to their content.

Interoperability is one of the basic aspects of Web 2.0 philosophy on which NOMAD capitalises, as it provides the means for leveraging policy related content. In this context, the current document delivers a study around the capabilities offered by solely Social Media NOMAD sources and can be exploited by the data acquisition modules, which will be developed in the next phase of the project. These capabilities focus on methods to retrieve content, types of available information and standards to structure and describe content.

Blogger, Wordpress, Facebook, Twitter, YouTube, Google+ and LinkedIn, are examined herein in regard to the publicly available software modules and programming interfaces. These seven Social Media Platforms comprise a representative set of places where users generate and publish policy related content. This selection is justified by the findings emerged from the "Categorisation of Web 2.0 Social Media" on their popularity and scope. The purpose of this analysis is to determine how certain aspects of these platforms can be re-used for the exploitation of social engagement in the scope of the policy making process through the NOMAD system. Chapter 3 presents the basic concepts behind each platform, the functionality and parameters of each method that each API exposes for retrieving content, as mentioned in the respective API Documentations.

What was revealed from the analysis is that the examined Social Media APIs provide methods for searching on a set of keywords and gathering related comments or posts and as well methods for extracting information about their authors. These methods will be called by NOMAD crawlers to accumulate content related to policy models. The selected methods return data that will be utilised in the NOMAD processing, with the view of establishment of a more citizen-centric and socially-rooted policy making. The inconsistency between the specific data is addressed in the next chapter 4, which prescribes the semantic interoperability essential for the unification of data fields. All the above analysis is focused on the two basic objects that NOMAD will process: comments and users who published these comments. . Hence, all capabilities required for building the NOMAD platform, are present in the APIs. However, an open issue to be decided is whether access to more data is needed through the authorization of NOMAD application in some of the Social Media platforms.

A similar to NOMAD approach on processing Web 2.0 content has already been adopted by existing tools that are used widely in business sectors. Thus, the current deliverable aims to gain insights from the comparative analysis between existing tools that refers to Social Media Monitoring and Digital Reputation Management. The review of 60 such tools available either free or commercially, revealed four groups of mainly web applications: Twitterverse Tools, Social Media Measurement Tools and Social Search Tools and Social Media Monitoring Dashboards. NOMAD combines features that appear in all these four categories. Implementation aspects resulted from this analysis will be conveyed in the policy formulation application field taking into account the different target groups engaged in the policy making procedures. The findings from this analysis enable a comparison with existing competing platforms and guide implementation decisions regarding the NOMAD capabilities.

Although, the work carried out in the current deliverable is especially focused on researching the specific Social Media related issues that contribute to the project objectives and consequently, drive the design of NOMAD tools, general conclusions for the integration of third party applications through available APIs, have emerged. Therefore, conclusions, concerning prevailing methods, standards, protocols and data formats are delivered in the final section.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF TERMS AND ABBREVIATIONS

API	Application Programming Interface
JSON	JavaScript Object Notation
REST	Representational State Transfer
URI	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
SaaS	Software as a Service
XML	Extensible Markup Language

1. INTRODUCTION

1.1 Purpose and Scope

With the advent of Web 2.0 philosophy, a new set of technologies came to light that facilitate the crowdsourcing and collaboration approach. In this era, users are treated as co-developers that are able to mash-up different existing services to create a new one. Thus, among the basic principles of Web 2.0 applications are to interoperate with other platforms and to ensure the existence of means for leveraging the volumes of data accumulated by their users on a daily basis. Social Media platforms ensure the availability of their content to external applications through their Application Programming Interfaces (APIs). NOMAD aspires to capitalise on this aspect of Web 2.0 to extract information related with policy making. The same approach is increasingly being adopted by many existing tools oriented towards digital reputation management and social media monitoring and are examined here.

In this context, the current document aims to deliver an in-depth study of the current publicly available software modules and interfaces of the Web 2.0 applications that constitute data sources of the NOMAD tools. The purpose of this study is to identify the capabilities that the Social media platforms offer to retrieve content, types of information they offer and standards to structure and describe content. The results from this analysis will determine how NOMAD components will interact with each platform to extract valuable for the project content. Methods that Social Media platform APIs exposes to allow automated interaction with third party applications are recorded herein. The analysis conducted, aspires to provide guidelines for the implementation of the NOMAD data acquisition modules regarding which of methods will be integrated and what data will be retrieved. According to this selection driven by the project needs, this deliverable finally prescribes the semantic interoperability among the targeted platforms that is needed for the unification of the data fields available from each platform.

1.2 Approach for Work Package and Relation to other Work Packages and Deliverables

As a starting point for the realisation of the WP2 objectives, a number of Web 2.0 Social media platforms with policy making related content were identified and categorised in Deliverable “D2.1 Classification of Web 2.0 Social Media and Stakeholder Characteristics”. Based on this input and the definition of usage scenarios conducted in “D2.2 Report on User Requirements”, the current deliverable consists of an attempt to map the technical characteristics of the selected NOMAD sources and result on an outcome that will be utilised in the subsequent technical WPs. Therefore, Web 2.0 Social media sources are analysed herein under a technical viewpoint, in order to produce guidelines for the development of the NOMAD components, namely the Data Acquisition module that will be implemented within WP4 “Opinion Mining and Argument Extraction” in and in parallel contribute to the architectural design principles described in deliverable “D6.1 NOMAD Architecture Design”. The implementation requirements to be obtained as a result of the in-depth analysis on available APIs and interoperability, complements “D2.2 Report on User Requirements”, in specifying the requirements that should be accomplished by the implementation of the NOMAD platform, within the WP2 scope.

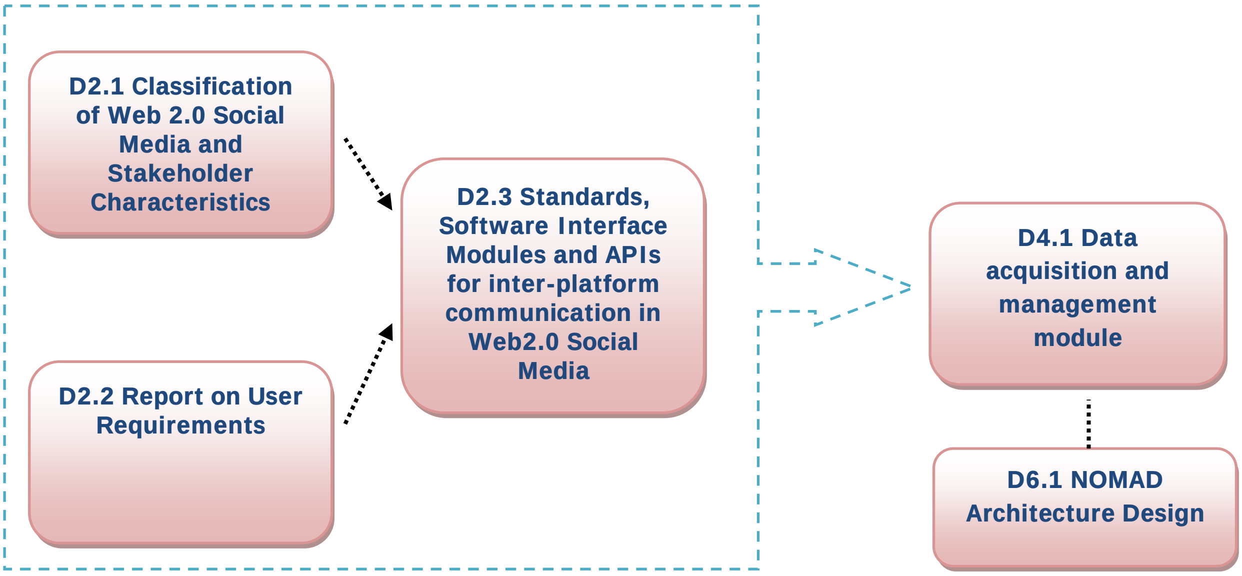
D2.1 Classification of Web 2.0 Social Media and Stakeholder Characteristics
D2.2 Report on User Requirements
D2.3 Standards, Software Interface Modules and APIs for inter-platform communication in Web2.0 Social Media
D4.1 Data acquisition and management module
D6.1 NOMAD Architecture Design

Figure 1: Interconnection of D2.3 with other project deliverables

1.3 Methodology and Structure of the Deliverable

The current document begins with an introductory section providing some definitions regarding technologies prevailing in Web 2.0. Chapter 2, also justifies the selection of seven Social Media platforms to be analysed in the subsequent sections. Blogger, Wordpress, Facebook, Twitter, YouTube, LinkedIn and Google+ are identified as representatives of the most popular platforms relevant to the policy making process according to the “Classification of Web 2.0 Social Media” delivered within D2.1.

Then, for each one of the targeted Social Media platforms, we identify specific APIs and interfaces for interacting with external applications. Among the APIs available from each Social Media Platform, we focus only on the APIs that support capabilities of performing search queries on a set of keywords. The specific APIs can be activated by the NOMAD crawler through the set of keywords generated from the policy models created by the NOMAD user. The examination of APIs unfolds in Chapter 4, which is organised around the seven social media applications. After presenting the concept behind them, each one is analysed with respect to a set of attributes that are essential for the implementation of the NOMAD data acquisition modules:

- The available Application Programming Interfaces (API) they provide and the types of communication they allow.
- Methods for retrieving data, through their APIs focusing on their main resources, for which NOMAD needs to extract content, i.e. users, posts, comments, etc.
- Description of data returned from each API call and listing of data fields that can be available for knowledge management within the project.
- Standards that platforms adapt to structure and describe this output in order to prescribe how they will be imported in the NOMAD repositories.
- Protocols to authorise request from external applications and type of permissions that will be required from NOMAD to retrieve private or publicly available data if needed.

The above information is illustrated in the deliverable as a summarisation of each individual API Documentation, limited to the elements that will be potentially used by the technical partners.

Having determined ways and methods for extracting each type of content, Chapter 4 summarises the data fields that can be acquired and utilised in the next stages of the NOMAD lifecycle, namely categorisation and visualisation. A mapping between the available data from each Social Media follows in order to define the inter-platform semantic interoperability.

Subsequently, Chapter 5 carries out a study on practices towards the exploitation of social engagement via Web 2.0 tools, used by citizens. Initially, an overview of existing tools that acquire and leverage content from Social Media Platform is provided. The 60 tools found, have been clustered in four groups, according to the approach they adopt and examined on a set of technical and general features. The findings emerged from this observation aims to provide a comparison to existing competing platforms and guide implementation decisions regarding the NOMAD capabilities.

Finally, the deliverable concludes with the remarks emerged from the conducted analysis that sets the implementation requirements for the project. Apart from the outcome essential for the completion of the project, the deliverable also draws some more conclusions associated with the social media API landscape and its exploitation, such as type of methods, availability of content, prevailing standards for exchanging content and authorisation protocols.

The structure of the document represents the methodology, which has been built in order to achieve the objectives of the current deliverable, i.e. to identify implementation aspects and requirements for the development of the NOMAD tools and is illustrated in the following Figure 2: Methodological approach of Deliverable 2.3.

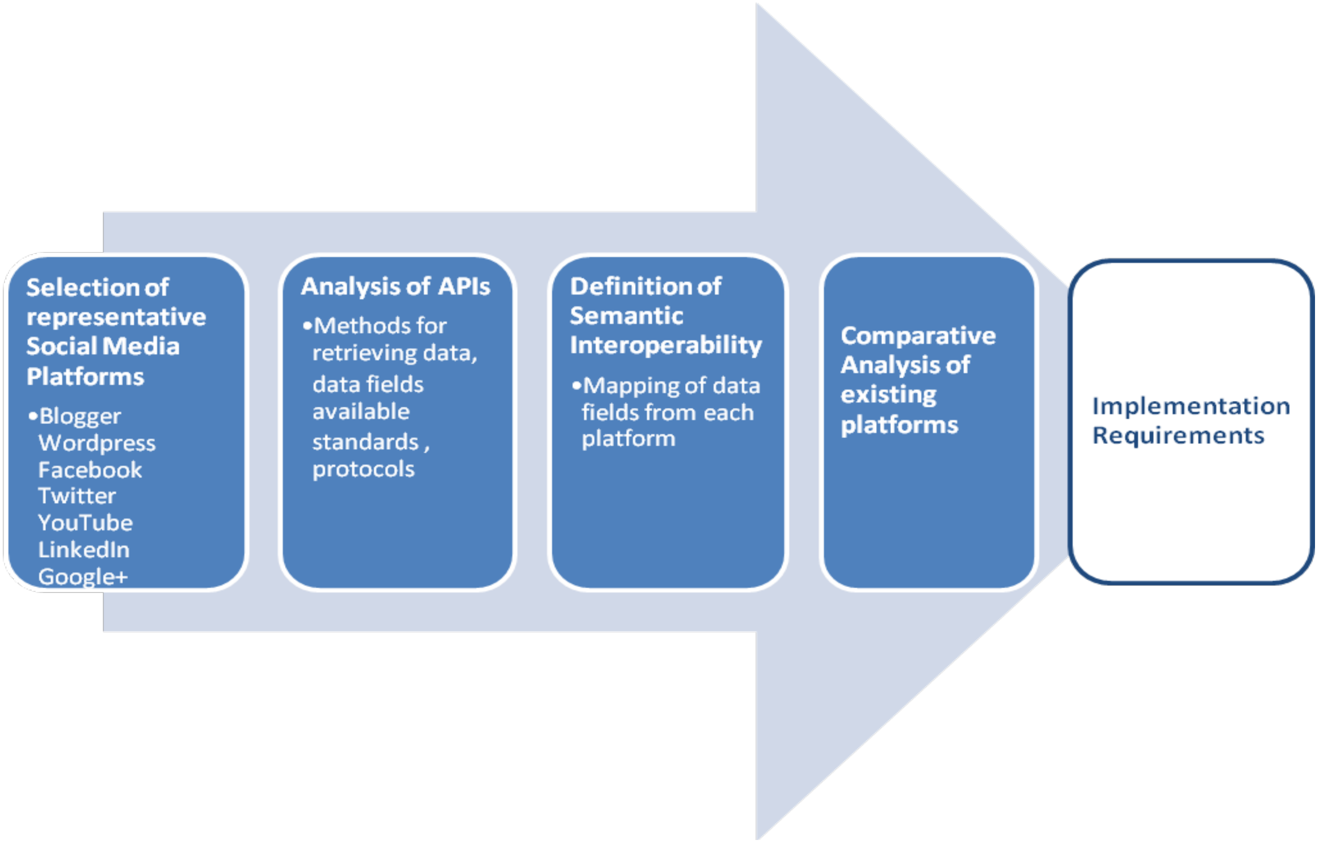
Selection of representative Social Media Platforms
•Blogger
Wordpress
Facebook
Twitter
YouTube
LinkedIn
Google+
Analysis of APIs
•Methods for retrieving data, data fields available standards , protocols
Definition of Semantic Interoperability
•Mapping of data fields from each platform
Comparative Analysis of existing platforms
Implementation Requirements

Figure 2: Methodological approach of Deliverable 2.3

2. TARGETED WEB 2.0 PLATFORMS

2.1 Rationale behind the selection of Web 2.0 platforms

In the Deliverable 2.1 we provided an extensive description and classification of Web 2.0 applications and platforms. Web 2.0 platforms were categorized into communication, collaboration, multimedia and entertainment, news, and policy making - public participation oriented ones. Further we ranked Web 2.0 platforms with respect to the number of users each platform has, while characteristics such as the Alexa ranking and whether the popularity is Worldwide or in some specific region was listed along with the platforms. Based on the results of the classification we ended up with a selection of seven Social Media platforms to be included in the current analysis, which meet two criteria; to be among the top twenty Alexa ranking positions and to belong (among other categories), to the “Policy making and public participation” category. In addition, we tried to narrow our choice taking into account their popularity in the project’s pilot countries rather than globally.

Therefore, out of all the available platforms that users generate and publish content to, we selected some representative ones to further analyse and obtain data from. Wordpress and Blogger are two of the most popular platforms for Blogging, while Twitter is by far the most popular platform for Micro-blogging. Facebook and Google+ are also some of the most popular social networking platforms with a lot of traffic worldwide. LinkedIn is not characterised as policy making related platform, as is focused more on the professional social networking. Due the fact that policy related content is not yet apparent there, for the time being LinkedIn will not constitute a NOMAD source. However, it is also included for reasons of completeness in the analysis, since it is considered as one of the most popular social media platforms (100.000.000 unique users) according to the stakeholders’ analysis of D2.1. All the above are classified as communication platforms as well.

Hence a very strong representation of communication platforms will be used to obtain data to be analyzed in the project. YouTube is a representative of the multimedia and entertaining platforms. It is one of the most popular video sharing platforms with rich content not only in terms of videos but also comments and video metadata and tags. Apart from being representative and rather popular among users all the aforementioned platforms offer easy to use APIs and return structured data in JSON format. This makes the specific platforms particularly attractive since not only content can be retrieved but connections between users and demographic information about users can be exposed through the corresponding APIs.

It should be mentioned that the selection of the aforementioned platforms deviates from what is defined as NOMAD sources in the Appendix I of Deliverable D2.1. This is due to the scope of the deliverable, to analyse platforms that can be embedded in the sources and NOMAD will access them through their APIs. Data from news sites and wiki sites that are not covered will also be obtained through the general crawling of data that will take place along with the acquisition of data from social media and hence those sites are not covered in this deliverable. A list of the Social Media platforms which have potential for NOMAD to obtain data and whose APIs will be analyzed in the remaining of the document can be viewed in Table 1. This list may be extended according to the conditions and needs that will emerge in the progress of the project.

Table 1: List of targeted Social Media platforms

Web 2.0 platform	Focus Description	Category
Blogger	Blogging platform	Communication / Policy Making & Public Participation
Wordpress	Blogging platform	Communication / Policy Making & Public Participation
Facebook	Social Networking Service	Communication / Policy Making & Public Participation
Twitter	Micro-blogging platform	Communication / Policy Making & Public Participation
YouTube	Video-sharing platform	Multimedia and Entertainment / Policy Making & Public Participation
Google+	Business Social Networking Service	Communication / Policy Making & Public Participation
LinkedIn	Professional Social Networking Service	Communication

2.2 Definitions and Terms

The following list provides definitions of the main technical terms that will be used in the subsequent sections of the deliverable.

Gadgets / Widgets: simple HTML and JavaScript applications that can be embedded in web pages and other apps. They are portable software applications for the web that can be installed and executed within a web page by an end user to offer simple functionality from third party widget publishers. A developer can build a gadget or widgets and make it available to other users, who can add it in their websites and applications.

JSON (JavaScript Object Notation): a common, language-independent data format that provides simple text representation of arbitrary data structures[1].

API: The acronym "API" stands for "Application Programming Interface" and is a specification intended to be used as an interface by software components to communicate with each other. An is API is a defined way for a program to accomplish a task, usually by retrieving or modifying data and may includes specifications for routines, data structures, object classes, and variables. Programmers use APIs to make applications, websites, widgets, and other projects that interact with other applications. For example, a program talks to the Twitter API over HTTP, the same protocol that browsers use to visit and interact with web pages. APIs establish the proper way for a developer to request services from a program. They are defined by the receiving programs, make working with other applications easier, and allow programs to communicate across different computing platforms.

REST (Representational State Transfer): is a style of software architecture for distributed hypermedia systems such as the World Wide Web. REST-style architectures consist of clients and servers[2]. Clients initiate requests to servers; servers process requests and return appropriate responses. Conforming to the REST constraints is referred to as being ‘RESTful’. RESTful applications maximize the use of the pre-existing, well-defined interface and other built-in capabilities provided by the chosen network protocol, and minimize the addition of new application-specific features on top of it.

Interoperability: the ability of two or more systems or components to exchange information and to use the information that has been exchanged. [1]

[1] json.org.

[2] http://en.wikipedia.org/wiki/Representational_state_transfer

3. API ANALYSIS OF WEB 2.0 PLATFORMS

This chapter presents a brief description of each platform, as also the functionality and parameters of each method that its API exposes, with emphasis on means for retrieving content and on what kind of data, useful for the policy formulation process, can be extracted. Due to the technical nature of this analysis section, it contains abstracts directly taken from the referenced online API Documentation of each platform.

3.1 Blogger

3.1.1 Description and Basic Concepts

Blogger is a platform for creating websites that allow people to publish their thoughts on an ongoing basis and share it on the Web.

Blogger is built on five basic concepts:

- **Blogs:** A blog has posts and pages. This is the container for blog meta-information like blog name and Description.
- **Posts:** A blog post is the publishable item that the blog author has created. This information is meant to be timely, reflecting what the authors want to publish to the world now. It is understood that as time passes, blog posts content ages and becomes less relevant. The content of the posts are of primary interest to the NOMAD project, since users through their posts can express their opinion and provide a series of arguments regarding any topic in their interest, which can be later analyzed.
- **Comments:** A comment is the place where people other than the blog post author react to what the author has written. Comments, similar to Posts, are of primary interest to the NOMAD project, since Blogger users can express their thought and opinions about the content of the Post they comment on, which can be used for extracting arguments towards some policy that might be discussed in the original Post.
- **Pages:** A page is a place for static content, such as biographical information, or the ways to contact the user. This is generally static information that doesn't change very often. Pieces of information contained in Pages can provide interesting demographic data with respect to any analysis that will be done within the NOMAD project on users' opinion and arguments on public policies. For instance, given a particular policy, analyzing Pages' data one can categorize different user arguments on the given policy by gender, age, educational background, location, etc.
- **Users:** A user is someone who interacts with Blogger, be they acting as an Author, an Administrator, or just a Reader.

3.1.2 Overview Blogger API v3

The Blogger API[3] v3 is a programming interface to Blogger. It allows client applications to view and update Blogger content. A client application can use Blogger API v3 to create new blog posts, edit or delete existing posts, and query for posts that match particular criteria. It is the latter that is of interest to the NOMAD project for the purpose of data acquisition. The API is organized around the three types of resources mentioned earlier: Blogs, which represent a blog; Posts, which represent a post on a blog; each posts resource is a child of a blogs resource; Comments, which represents a comment on a specific post; each comments resource is a child of a posts resource; Pages which represent a static page; each pages resource is a child of a blogs resource; and Users which represent a non-anonymous user; this resource is used to identify the Author of a page, post, or comment. Each resource has a JSON data representation with a number of fields and one or more methods to obtain this data. Most of the Blogger API follows a RESTful API design which means

[3] https://developers.google.com/blogger/

that standard HTTP methods can be used to retrieve and manipulate resources. For example, to get the post of a particular blog, one might send an HTTP request like:

- `GET https://www.googleapis.com/blogger/v3/blogs/blogId/posts`

The following table presents an overview of the Blogger API characteristics.

Table 2: Overview of Blogger API

	Blogger API
Version	Blogger API v3.0
Data Format	JSON
Calling styles	REST, REST from Javascript, client libraries
Authorization protocols	OAuth 2.0 / API key
Base URI	https://www.googleapis.com/blogger/v3

In what follows we are describing in more details the data and the methods to obtain this data for each one of the three resource types.
A detailed description of the data representation for Blogs is available at https://developers.google.com/blogger/docs/3.0/reference/blogs#resource.
There are three methods to obtain data for Blogs,

(a) *get*: given a blogID the method retrieves a blog.
(b) *getByUlr*: given a URL of a blog the method retrieves a blog.
(c) *listByUser*: given a userID the method retrieves a list of blogs that belong to the user.

The most relevant to the NOMAD project method here is the getByUrl method. NOMAD sources enclose a list of blogs (their URLs) that discuss topics of interest. These blogs can be obtained by supplying the getByUrl method the list of the URLs of interest. Through this method one can obtain the blogIDs of interest to be searched later on for terms specific to certain policies.

A detailed description of the data representation for Posts is available at https://developers.google.com/blogger/docs/3.0/reference/posts#resource.
There are a number of methods to obtain and manipulate data in posts. What we are interested in is gathering data rather than manipulating Posts (inserting, updating or deleting posts); hence there are three main methods for obtaining data for Posts,

(a) *list*: given a blogId the method retrieves a list of posts from the specified blog.
(b) *get*: given a blogId and a postId the method retrieves the specific blog post.
(c) *search*: given a blogId and a query string the method searches all the posts in the specified blog and returns a list of posts that contain the query string.

The most relevant to the NOMAD project method here is the search method. If for instance one searches for the word “farmers” in a blog by Nikos Dimou (a prominent Greek writer and intellectual) one of the top results returned is a post discussing the topic of EU support to Greek agriculture.

A detailed description of the data representation for Comments is available at https://developers.google.com/blogger/docs/3.0/reference/comments#resource.
There are two methods to obtain user comments,

(a) *list*: given a blogId and a postId the method retrieves a list of all comments for the particular post.
(b) *get*: given a blogId and a postId and a commentId the method retrieves the particular comment.

The most relevant to the NOMAD project method here is the list method. Having retrieved a number of posts through the search method above one can get all user comments about these interesting posts by this method.

A detailed description of the representation for Pages is available at https://developers.google.com/blogger/docs/3.0/reference/pages#resource.

There are two methods to obtain information about pages,

(a) *list*: given a blogId the method retrieves the list of pages for a blog.

(b) *get*: given a blogId and a pageId the method retrieves the specified page.

Finally the reader can get detailed description of the representation for Pages at, https://developers.google.com/blogger/docs/3.0/reference/users#resource.

There is a single method to obtain information about the user of Blogger,

(a) *get*: given a userId the method retrieves information about the user; this may include the user's country, location and language. Retrieving information about a user may require the request to be authorized by the user; hence not always it is possible to obtain this information.

3.1.3 Methods for retrieving content

The available methods to retrieve content from Blogger were described in the previous section. In this section we summarise the methods that are useful for the purposes of the project. Given a set of keywords extracted from policies or defined by policy makers and a list of blog URLs one can first search the Posts in Blogger, through the search method. After gathering the related content and obtaining the ID of each returned Post, the method list can be used to retrieve the comments on this post. Searching the posts and listing the available comments also returns the id of the authors both of the post and the comments. Given these userIds information about the users can also be obtained through the get method for the user resource. An overview of methods that will be used to obtain appropriate data is presented in Table 3: Blogger API methods for retrieving content

Table 3: Blogger API methods for retrieving content

Method	Resource Type	Parameters				Description	Returned Data
		Parameter	Type	Optional	Description		
BLOGS							
getByUrl	Blog	url	string	NO	The url of the blog to retrieve.	The method retrieves the blog that corresponds to the given url.	Id, name, description, published, updated,url, posts, locale
POSTS							
search	Post	blogId	string	NO	The ID of the blog whose posts will be searched.	The method searches the Blogger posts of a particular Blog for a certain query string and returns up to a maximum number of results. If the number of results is more than that then a pageToken is also returned that can be used to go through the next page of results in the next search request.	id, published, datetime, title, content, author, location
		q	string	NO	Full-text search query string.		
		pageToken	string	YES	The continuation token, used to page through large result sets.		
COMMENTS							
list	Comments	blogId	string	NO	The ID of the blog.	Given a blog and a post ID the method lists all the comments up to a number of results specified by maxResults, that have been posted regarding this blog post. If the number of comments is more than 20	Id, published, updated, content, author
		postId	string	NO	The ID of the blog post to get comments for.		

		maxResults	unsigned integer	YES	The maximum number of activities to include in the response, used for paging. Acceptable values are 1 to 20, inclusive. (Default: 10)	(the maximum value for maxResults) then a pageToken is also returned that can be used to go through the next page of results in the next list request. The erliest and latest dates of the comments to be fetch can also be specified.	
		pageToken	string	YES	The continuation token, used to page through large result sets.		
		endDate	datetime	YES	The latest date of comment to fetch.		
		startDate	datetime	YES	The earliest date of the comment to fetch.		
USER							
get	User	userId	string	NO	The ID of the user.	Given a user ID the method returns information about the user. The method requires however authorization.	id, about, locale

3.1.4 Authorizing requests

Blogger API uses the OAuth2 protocol for authentication and authorization. In a nutshell, one needs to (a) register their application with Google, through the Google API console, (b) redirect a browser to a URL, (c) parse a token from the response, and (d) send the token to the Google API they wish to access.

Registering an application generates a set of values that are known to both Google and the application. Before the application can access a Google API, it must obtain an access token that grants access to that API. Obtaining access token requires user consent. After an application has obtained an access token, it may send the access token in a request to a Google API. Access tokens are valid only for the set of operations and resources described in the token request. Access tokens are sent to a Google API in the HTTP Authorization header, or as a query string parameter (if HTTP header operations are not available).Access tokens have a limited lifetime and, in some cases, an application needs access to a Google API beyond the lifetime of a single access token. When this is the case, your application can obtain what is called a refresh token. A refresh token allows your application to obtain new access tokens.

Blogger doesn’t support only public blogs accessible from everyone, but also private blogs, which require authentication.

Every request an application sends to the Blogger API must identify the application to Google. An application can be identified by using an OAuth 2.0 token, which also authorizes the request, or by using the application's API key. Here's how to determine which of those options to use:

- If the request requires authorization, such as a request for an individual's private data, the application must provide an OAuth 2.0 token with the request. (include Authorization HTTP header)
- If the request doesn't require authorization, such as a request for public data, then the application must provide either the API key or an OAuth 2.0 token.

The benefit of the user authorisation is that it allows access on data that users keep private. However, what is questioned is whether the above process alters the non-moderated character of the project and introduces bias with certain users only allowing access to their private data. Due to the open nature of blogs, the only method from the ones described above that requires user’s consent is the *get* method on the *user* resource type. Hence, it should be decided if this method is going to be used by the data acquisition framework and will be depend on whether the amount of data that can be obtained without authorisation is sufficient for processing.

3.2 Wordpress

3.2.1 Description and Basic Concepts

Wordpress is a free web-based software that allows users to publish and maintain blogs. Wordpress users are able to publish content, to indicate interest and as well comment to other users' content. It also provides a user friendly interface for managing blogs' appearance, tools to integrate with other platforms, such us Yahoo! 360 and capabilities for publishing polls. Content of posts, likes and comments that refer to a post are of primary interest of NOMAD project. However, the blogging platform Wordpress.com should not be confused with wordpress.org, which is a Content Management System.

3.2.2 Overview of Wordpress API

Wordpress provides four different APIs with diverse types of functionality for handling content, such as images, posts and attachments. For the NOMAD project's purposes, we focus only on the Wordpress.com REST API[4], which enables third party applications to retrieve posts and comments from Wordpress blogs. This API has a RESTful design, meaning that the available resources can be accessed via http requests, through a unique ID.

For example, in order to get posts from a specific blog of interest, an application should make a call using an http request like the following:

- `GET https://public-api.wordpress.com/rest/v1/sites/$site/posts/`

The `$site` parameter should be replaced with the blog url, in the form of *sitename.wordpress.com*. The result of the above call is a JSON object.

Table 4: Overview of Worpress API

	Wordpress API
Version	REST API
Data Format	JSON
Calling styles	REST
Authorization protocols	OAuth 2.0 / API key
Base URI	https://public-api.wordpress.com/rest/v1/

3.2.3 Methods for retrieving content

Table 5: Wordpress API methods for retrieving content) presents methods for retrieving content concerning three types of interesting Wordpress resources, Posts, Likes on posts, Comments. Specifications for two basic methods are provided: the *get blog's posts* and *get comments for a post.* The first enables NOMAD to retrieve posts from a blog that is listed as source of information on a domain of interest, e.g. a blog on Renewable Energy Sources. Then, through the second NOMAD can obtain citizen's comments attached on each posts. The method *Get likes for a post* may provide additional information on people's interest on the specific posts.

Regarding the representation of a Wordpress user, Wordpress uses the "Gravatar"[5] web application for managing profiles. Gravatar is a service for providing globally unique avatars. An avatar is the graphical representation of the user or the user's alter ego or character. On Gravatar, users can register an account based on their email address, and upload an avatar to be associated with the account. Gravatar is provided natively in Wordpress, where when a user comments on a post or publishes a post, the Gravatar is displayed along with the content. Therefore, in order to get some information

[4] http://developer.wordpress.com/docs/api/

[5] https://en.gravatar.com/

about the user, an application has to utilise the Gravatar API[6] capabilities. Gravatar APIs requires no authentication and is based in HTTP requests, following a RESTful design approach. The Base URI of a Gravatar profile API call is *http://www.gravatar.com/HASH,* where HASH is the MD5 digest of the user's e-mail. Profile pages are fully marked up using hCard, a microformat for programmatically embedding information about people, companies, organizations, and places in HTML and other markup languages. The available user fields on an hCard are: Email address, IM accounts, Phone numbers, Verified accounts, Name, Personal Links, Image. Among these fields, only "Verifed accounts" is useful for our case, since it associates the Gravatar profile with other user's Social Media accounts (Twitter, Facebook, et.al.), and through which NOMAD can obtain data for the user utilising the methods described in the rest sub-sections.

[6] https://en.gravatar.com/site/implement/

Table 5: Wordpress API methods for retrieving content

Method	Resource Type	Parameters				Requires	Description	Returned Data
		Parameter	Type	Optional	Description			
POSTS - SEARCHING								
Get blog's posts	Blog posts	$site	(int|string)	NO	Return matching Posts	Access token	Return matching Posts http://developer.wordpress.com/docs/api/1/get/sites/%24site/posts/	id, author, title, like_count, geo, content, comment_count, date
		search	String	YES	Search query	Access token		
LIKES								
Get likes for a post	Post's likes	$site	Integer or String	YES	The site ID, The site domain	Access token	List the Likes for a Post http://developer.wordpress.com/docs/api/1/get/sites/%24site/posts/%24post_ID/likes/	Id, name
		$post_ID	Integer	YES	The post ID	Access token		
COMMENTS								
Get comments for a post	Post's comments	$site	Integer or String	YES	The site ID, The site domain	Access token	Return recent Comments for a post http://developer.wordpress.com/docs/api/1/get/sites/%24site/posts/%24post_ID/replies/	Id, author, date, comment

3.2.4 Authorizing requests

Authorized requests

Wordpress API uses the OAuth2 protocol for authentication. To act on a user's behalf and make calls from Wordpress API an access token is needed. To get an access token NOMAD developing team has to go through the access token flow and prompt the user to authorize NOMAD to act on his or her behalf. At start, NOMAD should send the user to the authorization endpoint.

`https://public-api.wordpress.com/oauth2/authorize?client_id=your_client_id&redirect_uri=your_url&response_type=code`

- *client_id* should be set to the application's client id
- response_type should always be set to "code"
- *redirect_uri* should be set to the URL that the user will be redirected back to after the request is authorized. The redirect_uri should be set in the applications manager.

The redirect to the application will include a code which will be needed need in the next step. In case the user has denied access to the app, the redirect will include `?error=access_denied`

Optionally a blog parameter (&blog=) may be passed with the URL to a WordPress.com blog. If it is not passed along a URL, or if the user does not have administrative access to manage the blog that has been passed along, then the user will be prompted to select the blog they are granting the app to access to.

Once the user has authorized the request, he or she will be redirected to the redirect_url. The request will look like the following:

- http://developer.wordpress.com/?code=cw9hk1xG9k

This is a time-limited code that the application can exchange for a full authorization token. To do this the code must be passed to the token endpoint by making a POST request.

Unauthorized requests

Wordpress, also supports unauthenticated requests in order to gain access to the available resources. It gives the ability to perform search for posts and comments.

For example, the request

- `https://public-api.wordpress.com/rest/v1/sites/en.blog.wordpress.com/posts`

will return the posts of the blog under the domain name en.blog.wordpress.com.

And similarly the request

- `https://public-api.wordpress.com/rest/v1/sites/en.blog.wordpress.com/posts/7/replies`

will return all comments matching to post with Id equal to 7.

Since access to posts and comments that is essential for our project is evitable without performing authorizing requests, NOMAD authentication is not going to be implemented at least for Wordpress. However, the steps of this process are described in this section in case it is required to harvest additional data.

3.3 Facebook

3.3.1 Description and Basic Concepts

Facebook is a social networking platform, with over one billion active users as of September 2012. Facebook provides multiple ways of sharing information and communication between the users. The ones, which are related to the project will be analysed below.

Each Facebook user has a profile where he is able to add lot of information about her or himself. For example one is able to add his age, places where he has lived, religious views, political views, contact info, info about his work and education, his relationship status, etc. We mainly focus on users' information that can be interesting for the project, for example in the demographic distribution of users' arguments, which depends on data available from each user (gender, age, location, etc.).

Each user belongs to Facebook's *Social Graph*. This Graph consists of objects and connections. Objects refer to users, photos, videos, status updates and other elements and connections refer to friend relationships, photo tags, likes etc.

One of the objects of interest of Facebook's Graph is *Status messages*. A user can express his opinion and arguments about a topic and then, this argument or opinion can be liked or commented by other users. Through Facebook's API the content of comments is available, the number of likes, the profile of the users that have commented and liked the Status message and other objects. Another object of interest is *Photos*. Like Status messages, Photos have connections such as *likes* and *comments* and so, demographic characteristics about the users that have liked a photo or have commented on it can be retrieved via the API. The same applies for objects like Video and Note. Finally, another capability that the Facebook's API offers is "Insights". Insights are offered for Pages (Fan Pages), Domains and applications. Insights provide information about a page such as number of posts within a page, the number of likes of a page, demographics about users that have visited the page, location information for users etc. Therefore, it will be valuable for the project, to get Insights about a Page that refers to a particular policy domain or belongs to a politician or an associated community.

3.3.2 Overview of Facebook API

Facebook's Graph API has a RESTful API design which means that every resource can be obtained via HTTP requests. The API is designed around Facebook's Social Graph which is described above, but more detailed reference about Graph API can be found at Social Graph API documentation[7].

An example is given below about a photo and comments of the photo:

- `GET https://graph.facebook.com/20531316728`
- `GET https://graph.facebook.com/20531316728/comments`

In Facebook's Social Graph, every object has a unique ID. The properties of any object that belongs to the graph can be accessed by requesting *https://graph.facebook.com/ID.*In the previous example we requested the properties of a photo in the Graph. The number 20531316728 is the ID of the photo.

The above request returns a JSON object:

```
{
   "name": "Facebook Platform",
   "website": "http://developers.facebook.com",
   "username": "platform",
   "founded": "May 2007",
   "company_overview": "Facebook Platform enables anyone to build...",
   "mission": "To make the web more open and social.",
   "products": "Facebook Application Programming Interface (API)...",
   "likes": 449921,
   "id": 19292868552,
   "category": "Technology"
}
```

[7] http://developers.facebook.com/docs/reference/api/

All responses of the requests to the Graph API are JSON objects.

Table 6: Overview of Facebook API

Facebook API	
Version	Graph API
Data Format	JSON
Calling styles	REST, REST from PHP, REST from JavaScript, REST from Android, REST from iOS SDK, FQL
Authorization protocols	OAuth 2.0 / API key
Base URI	https://graph.facebook.com/

3.3.3 Methods for retrieving content

In this section we summarise the methods for retrieving content from Facebook that are useful for the purposes of the project. Thus, comments on the various Facebook resources (Status messages, Photos, Videos, etc.) can be extracted with the *get comments* methods analysed below. As shown in the Table 7: Facebook API Methods for retrieving content, these methods can as well return the id of user created this comment (*form* parameter). Given this id, information about the people who have created the relevant content can be accessed through the get methods on user resource.
However, specific Facebook Pages were also included in NOMAD sources in the deliverable D2.1. So, to get insights from these or other pages devoted to a policy topic the homonymous method is going to be used. Finally, through the search method and given a set of keywords, one can search over the social graph on a set of keywords from policy models or domain models authored by policy makers.

Table 7: Facebook API Methods for retrieving content

Method	Resource Type	Parameters				Requires	Description	Returned Data
		Parameter	Type	Optional	Description			
STATUS MESSAGE								
Get comments	comments	id	String	YES	The Facebook ID of the comment	access_token	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/status/	Id, from, message, created_time, likes, user_likes, type
		from	String	YES	The user that created the comment	access_token		
		message	String	YES	The comment text	access_token		
		created_time	String	YES	The datetime the comment was created	access_token		
		likes	String	YES	The number of times this comment was liked	access_token		
		user_likes	String	YES	This field is returned only if the authenticated user likes this comment	access_token		
		type	String	YES	The type of this object; always returns comment	access_token		
Get likes	likes	id	String	YES	The Facebook ID of the person who made the like	access_token	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/status/	Id, name
		name	String	YES	The Facebook name of the person who made the like	access_token		
PHOTO								

Get comments	comments	id	String	YES	The Facebook ID of the comment	any valid access_token or user_photos or friends_photos	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/photo/	Id, from, message, created_time, likes, User_likes, type
		from	String	YES	The user that created the comment	any valid access_token or user_photos or friends_photos		
		message	String	YES	The comment text	any valid access_token or user_photos or friends_photos		
		created_time	String	YES	The datetime the comment was created	any valid access_token or user_photos or friends_photos		
		likes	String	YES	The number of times this comment was liked	any valid access_token or user_photos or friends_photos		
		user_likes	String	YES	This field is returned only if the authenticated user likes this comment	any valid access_token or user_photos or friends_photos		

		type	String	YES	The type of this object; always returns comment	any valid access_token or user_photos or friends_photos		
Get likes	likes	id	String	YES	The Facebook ID of the person who made the like	any valid access_token or user_photos or friends_photos	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/photo/	Id, name
		name	String	YES	The Facebook name of the person who made the like	any valid access_token or user_photos or friends_photos		
				VIDEO				
Get comments	comments	id	String	YES	The Facebook ID of the comment	user_videos	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/video	Id, from, message, created_time, likes, user_likes, type
		from	String	YES	The user that created the comment	user_videos		
		message	String	YES	The comment text	user_videos		
		created_time	String	YES	The datetime the comment was created	user_videos		
		likes	String	YES	The number of times this comment was liked	user_videos		

		user_likes	String	YES	This field is returned only if the authenticated user likes this comment	user_videos		
		type	String	YES	The type of this object; always returns comment	user_videos		
Get likes	likes	id	String	YES	The Facebook ID of the person who made the like	user_videos	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/video/	Id, name
		name	String	YES	The Facebook name of the person who made the like	user_videos		
					NOTE			
Get comments	comments	id	String	YES	The Facebook ID of the comment	Any valid access_token or user_notes or friends_notes.	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/note	Id, from, message, created_time, likes, user_likes, type
		from	String	YES	The user that created the comment	Any valid access_token or user_notes or friends_notes.		
		message	String	YES	The comment text	Any valid access_token or user_notes or friends_notes.		

		created_time	String	YES	The datetime the comment was created	Any valid access_token or user_notes or friends_notes.		
		likes	String	YES	The number of times this comment was liked	Any valid access_token or user_notes or friends_notes.		
		user_likes	String	YES	This field is returned only if the authenticated user likes this comment	Any valid access_token or user_notes or friends_notes.		
		type	String	YES	The type of this object; always returns comment	Any valid access_token or user_notes or friends_notes.		
Get likes	likes	id	String	YES	The Facebook ID of the person who made the like	Any valid access_token or user_notes or friends_notes.	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/reference/api/note/	Id, name
		name	String	YES	The Facebook name of the person who made the like	Any valid access_token or user_notes or friends_notes.		
PAGES, DOMAINS AND APPLICATIONS INSIGHTS								

Get Insights	Insights	since	That start and end dates can be applied using since and until, unix timestamps based on midnight in PST described in UTC	YES	start dates	generic access_token or read_insights	http://developers.facebook.com/docs/reference/api/insights/	statistics

		until	That start and end dates can be applied using since and until, unix timestamps based on midnight in PST described in UTC	YES	end dates	generic access_token or read_insights		
				USER				
Get id	User info	id	String	YES	The user's Facebook ID	No access_token required	All these fields are returned by default from an API call. If someone wants only some of them, he can use the parameters to restrict the result. http://developers.facebook.com/docs/ref	Id, gender, locale, languages, bio, birthday, education, work, religion, location,
Get gender		gender	String	YES	The user's gender: female or male	No access_token required		

Get locale		locale	String	YES	The user's locale	No access_token required	erence/api/user	home_town, relationship_status, checkins, friends
Get languages		languages	String	YES	The user's languages	user_likes		
Get bio		bio	String	YES	The user's biography	user_about_me or friends_about_me		
Get birthday		birthday	String	YES	The user's birthday	user_birthday or friends_birthday		
Get education		education	String	YES	A list of the user's education history	user_education_history or friends_education_history		
Get work		work	String	YES	A list of the user's work history	user_work_history or friends_work_history		
Get religion		religion	String	YES	The user's religion	user_religion_politics or friends_religion_politics		

Get location		location	String	YES	The user's current city	user_location or friends_location		
Get home_town		Home_town	String	YES	The user's hometown	user_hometown or friends_hometown		
Get relationship_status		relationship_status (connection)	String	YES	The user's relationship status: Single, In a relationship, Engaged, Married, It's complicated, In an open relationship, Widowed, Separated, Divorced, In a civil union, In a domestic partnership	user_relationships or friends_relationships		
Get checkins		checkins (connection)	String	YES	The places that the user has checked-into.	user_checkins or friends_checkins.		
Get friends		friends (connection)	String	YES	The user's friends.	Any valid access_token of the current session user.		
					SEARCHING			
Search	Searching over all public objects in the	QUERY	String	NO	The query string we are looking for	-	Searching over all public objects in the social graph	JSON objects referring to

	social graph	OBJECT_TYPE	String	NO	The object of the type that the above query refers to	-	developers.facebook.com/docs/reference/api	QUERY and OBJECT_TYPE parameters

3.3.4 Authorizing requests

Unauthorized Requests

Facebook API, gives the ability to retrieve basic profile information about a user by making unauthorized http requests. Access to basic profile info is obtained by making a simple http request like the following:

- `https://graph.facebook.com/btaylor`

where a JSON object is returned with the following fields: id, name, first_name, last_name, link, username, gender and locale.

Also, comments and likes about a post are available, via http requests like the followings:

- `https://graph.facebook.com/POST_ID/comments`

and

- `https://graph.facebook.com/POST_ID/likes`

The above methods are easy enough and provide a little information, but if someone has applied restricted access to his personal data or the comments and likes are not set as visible to everyone, Oauth authentication and extra permissions will have to be granted in order to have the above and more even more detailed results and information about a User or a Post.

Authorized Requests

In order to get additional information about a user, his permission is required. An Access Token must be obtained for the user. After an Access Token has been obtained, authorized requests on behalf of that user can be made by including the access token in the Graph API requests.

- Eg. `https://graph.facebook.com/220439?access_token=...`

The Graph API uses OAuth 2.0 for authorization.

Getting an access token for a user with no extended permissions allows accessing the information that the user has made available to everyone on Facebook. If specific information is needed about a user, like their email address or work history, specific extended permissions must be obtained, which are described in detail, in Table 7: Facebook API Methods for retrieving content.

Authorisation can expand the availability of data for NOMAD. These means that unless user' permission is asked, NOMAD crawlers will perform request to access public information about users and other Facebook objects of interest. Focusing only on information with unrestricted access ensures that the non moderated character is maintained since user is not informed. However, this approach prerequisites that enough data are publicly available.

3.4 Twitter

3.4.1 Description and Basic Concepts

Twitter is an online social networking and micro-blogging service that enables users to send and read text-based messages known as "tweets". Each tweet is maximum 140 characters long. A user can view videos and conversations directly in Tweets to get the whole story at a glance, and all in one place.

A user can retweet another user’s tweet or add it to his favourites. He can also post pictures or photos. Another thing he can do is to follow other users on order to receive their tweets. For the project the topics of interest are tweets referring to a particular keyword, the retweets of a specific tweet and various information about the user that published the tweet or retweeted it. Twitter text corpus can be vary valuable for the project , as it has been adopted by civilians as a means of succinct expression in hot policy topics.

3.4.2 Overview Twitter API

In Twitter REST API[8] , which has a RESTful design, every resource has a unique id and can be accessed with an http request.

For example, if someone wants to get the most recent tweets of the logged in user, makes an http request. For example:

- `GET https://api.twitter.com/1.1/statuses/user_timeline.json`

And an array of objects is returned. Twitter API returns JSON.

The main elements of interest for the project are tweets, number of retweets for a particular tweet, profile data about users like demographic information. Twitter API provides that data with many methods which are presented below.

Table 8: Overview of Twitter API

	Twitter API
Version	REST API v1.1
Data Format	JSON
Calling styles	REST, REST from ActionScript/Flash, REST from C++, REST from Clojure, REST from ColdFusion, REST from Erlang, REST from Java, REST from Javascript, REST from .NET, REST from Objective C / Cocoa, REST from Perl, REST from PHP, REST from Python, REST from Ruby, REST from Scala
Authorization protocols	OAuth 2.0 / API key
Base URI	https://api.twitter.com/1.1/

3.4.3 Methods for retrieving content

The most useful Twitter method for our project is ***Search tweets***, since it enables the NOMAD application to search and return all Twitter messages, where a keyword is mentioned. It provides also the capability to narrow search results coming from users from a specific location or written in a specific language. This can be vary valuable for the policy maker who wants to find out what people say in Twitter regarding a policy in a specific country or region during a given time interval. Furthermore, *Get user* timeline can be utilised to retrieve tweets from a particular Twitter account that is of NOMAD’s interest.

[8] https://dev.twitter.com/docs

Table 9: Twitter API Methods for retrieving content

Method	Resource Type	Parameters				Requires	Description	Returned Data
		Parameter	Type	Optional	Description			
				TWEETS				
Get user timeline	Tweets	user_id	String	YES	The ID of the user for whom to return results for. Helpful for disambiguating when a valid user ID is also a valid screen name. Always specify either a user_id or screen_name when requesting a user timeline.	Access token, Read permission	Returns a collection of the most recent Tweets posted by the user indicated by the screen_name or user_id parameters. This method can only return up to 3,200 of a user's most recent Tweets. Native retweets of other statuses by the user is included in this total,	id_str, retweet_count, text, user
		screen_name	String	YES	The screen name of the user for whom to return results for. Helpful for disambiguating when a valid screen name is also a user ID. Always specify either a user_id or screen_name when requesting a user timeline.	Access token, Read permission		
		since_id	Integer	YES	Returns results with an ID greater than (that is,	Access token, Read		

					more recent than) the specified ID. There are limits to the number of Tweets which can be accessed through the API. If the limit of Tweets has occurred since the since_id, the since_id will be forced to the oldest ID available.	permission	regardless of whether include_rts is set to false when requesting this resource. https://dev.twitter.com/docs/api/1.1/get/statuses/user_timeline	
		count	Integer	YES	Specifies the number of tweets to try and retrieve, up to a maximum of 200 per distinct request. The value of count is best thought of as a limit to the number of tweets to return because suspended or deleted content is removed after the count has been applied. We include retweets in the count, even if include_rts is not supplied. It is recommended you always send include_rts=1 when using this API method.	Access token, Read permission		
		max_id	Integer	YES	Returns results with an ID less than (that is, older than) or equal to the specified ID.	Access token, Read permission		

		trim_user	Boolean	YES	When set to either true, or 1, each tweet returned in a timeline will include a user object including only the status authors numerical ID. Omit this parameter to receive the complete user object.	Access token, Read permission		
		exclude_replies	Boolean	YES	This parameter will prevent replies from appearing in the returned timeline. Using exclude_replies with the count parameter will mean you will receive up-to count tweets — this is because the count parameter retrieves that many tweets before filtering out retweets and replies. This parameter is only supported for JSON and XML responses.	Access token, Read permission		
		contributor_details	Boolean	YES	This parameter enhances the contributors element of the status response to include the screen_name of the contributor. By default only the user_id of the contributor is included.	Access token, Read permission		

		include_rts	Boolean	YES	When set to false, the timeline will strip any native retweets (though they will still count toward both the maximal length of the timeline and the slice selected by the count parameter). Note: If you're using the trim_user parameter in conjunction with include_rts, the retweets will still contain a full user object.	Access token, Read permission		
				RETWEETS				
Get retweets of a tweet	retweets	id	Integer	NO	The numerical ID of the desired status.	Access token, Read permission	Returns up to 100 of the first retweets of a given tweet. https://dev.twitter.com/docs/api/1.1/get/statuses/retweets/%3Aid	id_str, retweet_count, text, user
		count	Integer	YES	Specifies the number of records to retrieve. Must be less than or equal to 100.	Access token, Read permission		
		trim_user	Boolean	YES	When set to either true, or 1, each tweet returned in a timeline will include a user object including only the status authors numerical ID. Omit this parameter to receive the complete user object.	Access token, Read permission		
				USER				

Get info about the user	Information about the user	user_id	Integer	NO	The ID of the user for whom to return results for. Either an id or screen_name is required for this method.	Access token, Read permission	Returns a variety of information about the user specified by the required user_id or screen_name parameter. The author's most recent Tweet will be returned inline when possible. https://dev.twitter.com/docs/api/1.1/get/users/show	follow_request_sent, followers_count, geo_enabled, id, lang, location, statuses_count, time_zone
		screen_name	String	NO	The screen name of the user for whom to return results for. Either a id or screen_name is required for this method.	Access token, Read permission		
		include_entities	Boolean	YES	The entities node will be disincluded when set to false.	Access token, Read permission		
				SEARCHING				
Search tweets	tweets	q	@String	NO	A UTF-8, URL-encoded search query of 1,000 characters maximum, including operators. Queries may additionally be limited by complexity.	Access token, Read permission	Returns a collection of relevant Tweets matching a specified query.	id_str, retweet_count, text, user
		geocode	float	YES	Returns tweets by users located within a given radius of the given latitude/longitude. The location is preferentially taking from the Geotagging API, but will fall back to	Access token, Read permission		

					their Twitter profile. The parameter value is specified by "latitude,longitude,radius", where radius units must be specified as either "mi" (miles) or "km" (kilometers). Note that you cannot use the near operator via the API to geocode arbitrary locations;however you can use this geocode parameter to search near geocodes directly. A maximum of 1,000 distinct "sub-regions" will be considered when using the radius modifier.			
		lang	String	YES	Restricts tweets to the given language, given by an ISO 639-1 code. Language detection is best-effort.	Access token, Read permission		
		locale	String	YES	Specify the language of the query you are sending (only ja is currently effective). This is intended for language-specific consumers and the default should work in the majority of cases.	Access token, Read permission		
		result_ty	String	YES	Specifies what type of	Access		

		pe			search results you would prefer to receive. The current default is "mixed." Valid values include: * mixed: Include both popular and real time results in the response. * recent: return only the most recent results in the response * popular: return only the most popular results in the response.	token, Read permission		
		count	Integer	YES	The number of tweets to return per page, up to a maximum of 100. Defaults to 15. This was formerly the "rpp" parameter in the old Search API.	Access token, Read permission		
		until	String	YES	Returns tweets generated before the given date. Date should be formatted as YYYY-MM-DD. Keep in mind that the search index may not go back as far as the date you specify here.	Access token, Read permission		
		since_id	Integer	YES	Returns results with an ID greater than (that is, more recent than) the specified ID. There are limits to the number of Tweets which can be	Access token, Read permission		

					accessed through the API. If the limit of Tweets has occured since the since_id, the since_id will be forced to the oldest ID available.			
		max_id	Integer	YES	Returns results with an ID less than (that is, older than) or equal to the specified ID.	Access token, Read permission		
		include_entities	Boolean	YES	The entities node will be disincluded when set to false.	Access token, Read permission		
		callback	String	YES	If supplied, the response will use the JSONP format with a callback of the given name. The usefulness of this parameter is somewhat diminished by the requirement of authentication for requests to this endpoint.	Access token, Read permission		

In the following table, the user profile fields that will be extracted to be used for the project:

Table 10: Twitter User's data to be obtained

Field	Description
follow_request_sent	*Nullable*. *Perspectival*. When true, indicates that the authenticating user has issued a follow request to this protected user account. Example: "follow_request_sent":false
followers_count	The number of followers this account currently has. Under certain conditions of duress, this field will temporarily indicate "0." Example: "followers_count": 21
geo_enabled	When true, indicates that the user has enabled the possibility of geotagging their Tweets. This field must be true for the current user to attach geographic data when using POST statuses/update. Example: "geo_enabled":true
id	The integer representation of the unique identifier for this User. This number is greater than 53 bits and some programming languages may have difficulty/silent defects in interpreting it. Using a signed 64 bit integer for storing this identifier is safe. Use id_str for fetching the identifier to stay on the safe side. See Twitter IDs, JSON and Snowflake. Example: "id":6253282
lang	The BCP 47 code for the user's self-declared user interface language. May or may not have anything to do with the content of their Tweets. Examples: "lang":"en" "lang":"msa" "lang":"zh-cn"
location	*Nullable*. The user-defined location for this account's profile. Not necessarily a location nor parseable. This field will occasionally be fuzzily interpreted by the Search service. Example: "location":"San Francisco, CA"

statuses_count	The number of tweets (including retweets) issued by the user. Example: "statuses_count": 42
time_zone	*Nullable*. A string describing the Time Zone this user declares themselves within. Example: "time_zone":"Pacific Time (US & Canada)"

3.4.4 Authorizing requests

Authorized requests

Twitter API uses Oauth protocol for authentication. Once an application is registered, an access Token is generated in order to make API calls. Extra permissions are set in the application's control panel like read only for reading only, read-write for reading and posting and read-write and direct messages in order to have access to users' direct messages.

Unauthorized Requests

Twitter API gives us as well the option for unauthenticated search requests. This is required for the NOMAD to perform searches for tweets, based on a keyword. This can be done directly and without obligation of going through the authentication flow.

For example:

The HTTP request

- `http://search.twitter.com/search.json?q=blue`

returns tweets referring to blue.

In any case, Twitter encapsulates open character. Therefore, as shown above, the second option meets the needs of NOMAD, since unauthorized requests can ensure access to all public tweets through the search method. Thus, data acquisition modules may skip the authorization process concerning Twitter.

3.5 YouTube

3.5.1 Description and Basic Concepts

YouTube is a video-sharing website, created by three former PayPal employees in February 2005, on which users can upload, view and share videos. The YouTube service uses Adobe Flash Video and HTML5 technology to display a wide variety of user-generated video content, including movie clips, TV clips, and music videos, as well as amateur content such as video blogging and short original videos. The most interesting part of YouTube functionality is the comments that are attached on a video and are uploaded by its viewers, among the over 800 million YouTube unique users per month.

3.5.2 Overview of YouTube API

Youtube API, has a RESTful design. That means that the available resources can be accessed via http requests. Also, every resource in Youtube API has a unique ID. Data API[9] lets a user perform most of the operations a normal YouTube user can on the YouTube website. Also a number of different platforms libraries are offered. YouTube Data API may returns data either in JSON or XML format.

For example, in order to make a call to get comments on a video the http request is like below:

- `https://gdata.youtube.com/feeds/api/videos/VIDEO_ID/comments`

Table 11: Overview of YouTube API

YouTube API	
Version	2(version 3 exists as well, but is not used herein due to its experimental state)
Data Format	XML, JSON
Calling style	REST, REST from Java, REST from .NET, REST from PHP, REST from Python, REST from Objective –C, REST from Javascript
Authorization protocols	OAuth 2.0 / API key
Base URI	https://gdata.youtube.com/feeds/api/

3.5.3 Methods for retrieving content

As already mentioned, among the available YouTube resources, only comments can be processed by NOMAD. However what is needed to trigger this process is to find out all videos that are in a particular category or that are associated with a particular keyword related with the policy under examination. This is evitable throuth the method *Browsing with Categories and Keywords.* After listing these videos, the associated comments are accessible through the method *Retrieve comments.* Then NOMAD can use *Retrieve user profiles* to gather data about users who have contributed to content.

[9] https://developers.google.com/youtube/2.0/developers_guide_protocol_audience

Table 12: YouTube API Methods for retrieving content

Method	Resource Type	Parameters				Requires	Description	Returned Data
		Parameter	Type	Optional	Description			
VIDEOS								
Browsing with Categories and Keywords	Videos	Category	String	YES	The category of videos the user is interested in. Either category or tag must be specified.	Developer key, access token	retrieve a feed of all of the videos that are in a particular category or that are associated with a particular keyword. (YouTube uses the term "tag" to identify a keyword relevant to a video.) https://developers.google.com/youtube/2.0/developers_guide_protocol_category_keyword_browsing	videos
		Tag	String	YES	The tag of videos. Either category or tag must be specified.	Developer key, access token		
DATA FOR SINGLE VIDEO								
Retrieving data for single video	Videos	videoid	String	NO	The video ID for a video is identified in feed entries by the <yt:videoid> tag. This tag appears in video feed entries – including standard feeds, search results, user-uploaded video feeds, etc. – as well as in favorite video feed entries, playlist feed entries, inbox feed entries and several types of activity feed entries.	Developer key, access token	retrieve information about a single video https://developers.google.com/youtube/2.0/developers_guide_protocol_video_entries	Id, Uploader_Id, viewCount, favourite_count, numDislikes, numLikes
USER								

Retrieve user profiles	User	userId	String	NO	User's unique id	Developer key, access token	allows you to retrieve user profiles https://developers.google.com/youtube/2.0/developers_guide_protocol_profiles	About me, age, gender, hometown, location, hobbies, occupation, relationship
COMMENTS								
Retrieve comments	Comments	**Video id**	String	NO	Video's unique id	Developer key, access token	https://developers.google.com/youtube/2.0/developers_guide_protocol_comments?hl=en#Retrieve_comments	Author, Content, published

3.5.4 Authorizing requests

Authorized requests

The YouTube Data API supports the OAuth 2.0 protocol for authorizing access to private user data. The list below explains some core OAuth 2.0 concepts:

- When a user first attempts to use functionality in the external application that requires the user to be logged in to a Google Account or YouTube account, the application initiates the OAuth2 authorization process.
- The application directs the user to Google's authorization server. The link to that page specifies the scope of access that application is requesting for the user's account. The scope specifies the resources that your application can retrieve, insert, update, or delete when acting as the authenticated user.
- If the user consents to authorize your application to access those resources, Google will return a token to your application. Depending on your application's type, it will either validate the token or exchange it for a different type of token.

For example, a server-side web application would exchange the returned token for an access token and a refresh token. The access token would let the application authorize requests on the user's behalf, and the refresh token would let the application retrieve a new access token when the original access token expires.

Unauthorized requests

However, YouTube provides the option for the project implementation to make unauthenticated http requests in order to search for videos, comments and user profile information.

For example:

The request

- `https://gdata.youtube.com/feeds/api/videos/-/category_or_tag`

will return videos relative to the category or tag given as a parameter.

The request

- `https://gdata.youtube.com/feeds/api/videos/VIDEO_ID/comments`

will return comments for the video with the specified video Id.

And finally the request

- `https://gdata.youtube.com/feeds/api/users/userId`

will return first_name, last_name, location and statistics info about the user.

The same situation with Twitter applies here. Thus there is no demand yet for NOMAD’s authorisation to use YouTube API.

3.6 Google+

3.6.1 Description and Basic Concepts

Google+ is a social networking platform for sharing on the Web. Google+ provides numerous ways of sharing and communication, including sharing stream of context posts, videos and pictures to video conferencing among people on the Google+ platform. Here we focus on those of interest to the project.

Each user of Google+ has a profile that consists of several pieces of information about the user. This includes the user's name, birthday, gender, a short piece of text written by the user about themselves, the user's location, relationship status, the languages spoken by the user, a list of current or past organizations with which this person is associated, e.g. organizations the user has worked for, or studied at, and other details. These pieces of information can provide interesting demographic data with respect to any analysis that will be done within the NOMAD project on users' opinion and arguments on public policies. For instance, given a particular policy, analyzing Google+ data one can categorize different user arguments on the given policy by gender, age, educational background, location, etc.

Each user is associated with a stream of activities. Activities are typically pieces of text (content) posted to be shared with others user, but may include for instance checking-in at a restaurant or re-sharing other users' activities. The content of the posts is of primary interest to the NOMAD project, since users through their posts can express their opinion and provide a series of arguments regarding any topic in their interest, which can be later analyzed. Apart from the content, activities may also come with attached articles, videos and images. The attached articles can also be of interest to the project since posts may express a user's opinion about an article, and hence the article can help to place a user's opinion in the right context. However, given that attached articles may or may not include publication dates and that time plays a significant role in the analysis to follow, it is unclear whether attached articles would help or introduce extra noise in the process. Activities have a large number of other descriptors, e.g. the time the activity was originally published and modified, the person that performed the activity, people who +1'ed the activity, people who re-shared the activity, some times even the latitude and longitude of the placethe activity occurred at, etc.

Finally, each activity is associated with comments. Comments are replies to the activity. The content of the comments are also of primary interest for the analysis of arguments that will take place later in the project. Information about the users that post these comments is also available and it could be used for the purpose of gathering demographics and categorizing the results of the analysis that will follow.

3.6.2 Overview Google+ API

The Google+ API is a programming interface to Google+. It enables people to obtain data published through the Google+ platform. The API is organized around the three types of resources mentioned earlier, People, Activities and Comments. Each resource has a JSON data representation with a number of fields and one or more methods to obtain this data. Most of the Google+ API follows a RESTful API design that is standard HTTP methods can be used to retrieve and manipulate resources. For example, to get the profile of a user, you might send an HTTP request like:

- `GET https://www.googleapis.com/plus/v1/people/userId`

Table 13: Overview of Google+ API presents an overview of the Google+ API characteristics.

Table 13: Overview of Google+ API

Google Plus API	
Version	3.0 (Previous 1.0, 2.0)
Data Format	JSON
Calling styles	REST, REST from Javascript, REST from python, client libraries
Authorization protocols	OAuth 2.0 / API key
Base URI	https://developers.google.com/+/api/

In what follows we are describing in more details the data and the methods to obtain this data for each one of the three resource types.

The reader can get a detailed description of the data representation for People at,

- https://developers.google.com/+/api/latest/activities#resource.

There are three methods to obtain data for People,

a. *get*: given a userID the method gets a user's profile.
b. *search*: given a query string the method searches all text in public profiles and returns a list of users (along with their userIDs)
c. *listByActivity*: given the ID of an activity and a collection (plusoners or reshares) returns a list of People in the specified collection for a particular activity.

The most relevant to the NOMAD project method here is the search method. If for instance one searches for the word "taxes" one of the top results returned is the public interface of Mitt Romney.

The reader can get a detailed description of the data representation for Activities at,

- https://developers.google.com/+/api/latest/people#resource,

There are again three methods to obtain data for Activities,

a. *list*: given a userID the method returns a list of activities associated with that user. Combined with the *search* method on People one can for instance search for "taxes", obtain the id of Mitt Romney and the get all posts posted by Mitt Romney that can be possibly of interest for the purpose of public policy and arguments analysis.
b. *get*: given the ID of an activity returns the data of the activity.
c. *search*: given a query string the method searches all text in public activities and returns a list of them.

The latter method is the most useful for the purpose of collecting data for NOMAD. Keywords extracted from policies can be used to query Google+ and collect activities that best match these keywords. For instance when querying Google+ by the keyword *φοροδιαφυγή* (tax evasion), the top two posts are:

- *"Με έκπληξη διάβασα ότι το 98% των Ελλήνων πολιτών (Πηγή: Ευρωβαρόμετρο, Σεπτέμβριος 2011) θεωρούν τη διαφθορά «μείζον πρόβλημα της χώρας» και το Πανεπιστήμιο του Σικάγο υπολογίζει τη φοροδιαφυγή μόνο από τους ελεύθερους επαγγελματίες στην Ελλάδα, σε 28 δις ευρώ! Πηγή: NikolaosArtavanis / Adair Morse / Margarita Tsoutsoura: Tax Evasion Across Industries: Soft Credit Evidence From Greece, June 2012)."*
- *"Ενας εκ των 13 βουλευτών της ΝΔ που κατέθεσαν ερώτηση στη Βουλή ζητώντας να γίνουν γνωστά τα ονόματα όσων έβγαλαν τα χρήματά τους στο εξωτερικό και αν τα χρήματα αυτά προέρχονται από φοροδιαφυγή εμπλέ..."*

The reader can get a detailed description of the data representation for Comments at,

- https://developers.google.com/+/api/latest/comments#resource.

There are two methods to obtain data for Comments,

a. *list*: given an activity ID the method returns a list of comments on that activity. Combined with the search method on activities one can obtain the comments for all the activities that match the search terms "tax evasion".
b. *get:* given the ID of a comment the method returns the data for that comment.

Using the *list* method with the activity ID of the first post above regarding tax evasion we can obtain the following two comments on the post along with interesting information about the users that posted these comments (as mentioned in the previous section):

- "*Η πλήρης αποτυχία (βλ. απροθυμία) της Ελλάδας να συμμορφωθεί σε 27 σαφείς και συγκεκριμένες προτάσεις της Επιτροπής κρατών κατά της διαφθοράς του Συμβουλίου της Ευρώπης (Greco), οι οποίες αγνοήθηκαν σε μεγάλο βαθμό, οδήγησαν τη χώρα μας σε καθεστώς επιτήρησης μέχρι το τέλος του 2012. Οι προτάσεις αυτές, οι οποίες συγκροτήθηκαν σε μία έκθεση, η οποία πρόκειται να μεταφραστεί στα ελληνικά και να δοθεί σύντομα* *στη δημοσιότητα, αφορούν αλλαγές στον Ποινικό Κώδικα, στη χρηματοδότηση των κομμάτων καθώς και στο εποπτικό πλαίσιο, το οποίο κρίνεται ως άκρως αναποτελεσματικό, αφού κριτές και ελεγχόμενοι ουσιαστικά ταυτίζονται. (Πηγή: ΝομικόςΣύμβουλοςΣεπτέμβριος - Οκτώβριος 2012, τεύχος 69)."*

- *"Βέβαια, η Greco δεν έχει τη δυνατότητα να επιβάλει ουσιαστικές κυρώσεις στη χώρα μας, απλά την εκθέτει για μια ακόμη φορά, καθώς όπως αναφέρεται στην εν λόγω έκθεση «ακόμα και αν τα πενιχρά αποτελέσματα δικαιολογούνται εν μέρει από τις δύσκολες συνθήκες που επικρατούν στην Ελλάδα, προκαλεί εντύπωση το γεγονός ότι δεν σημειώθηκε πρόοδος ακόμη και σε συστάσεις που δεν απαιτούσαν νομοθετικές πρωτοβουλίες για να εφαρμοστούν». (Πηγή: ΝομικόςΣύμβουλος, Σημείωμα τηςσύνταξης, Σεπτέμβριος - Οκτώβριος 2012, τεύχος 69)."*

3.6.3 Methods for retrieving content

The available methods to retrieve content from Google+ were described in the previous section. In this section we are repeating the methods that are most useful for the purposes of the NOMAD project. Given a set of keywords extracted from policies or defined by policy makers one can first search the Activities (and in particular the posts) in Google+, through the *search* method. The language of the search term typically specifies the language of the obtained search results, however further constraints on the language of the results can be imposed. After gathering the related posts and obtaining the ID of each returned Activity, the method *list* can be used to retrieve the comments on this activity. Demographic information can be gathered about the users that posted the original post, the comments or even those that reshared or +1'd the post.

An overview of methods that will be used to obtain appropriate data is presented in Table 14: Google+ API methods

Table 14. Google+ API methods to obtain data

Method	Resource Type	Parameters				Description	Returned Data
		Parameter	Type	Optional	Description		
				ACTIVITIES			
search	Activities	query	string	NO	Full-text search query string.	The method searches the Google+ Activities data for a certain query string and possible within a certain language and returns up to a number of results specified by maxResults, in a best-ranked or recent-ranked order. If the number of results is more than 20 (the maximum value for maxResults) then a pageToken is also returned that can be used to go through the next page of results in the next search request.	Title, published, updated, actor, content, original_content, geocode
		language	string	YES	Specify the preferred language to search with.		
		maxResults	unsigned integer	YES	The maximum number of activities to include in the response, used for paging. Acceptable values are 1 to 20, inclusive. (Default: 10)		

		orderBy	string	YES	Specifies how to order search results. Acceptable values are: "best" and "recent" (default)		
		pageToken	string	YES	The continuation token, used to page through large result sets.		
COMMENTS							
list	Comments	activityID	string	NO	The ID of the activity to get comments for.	Given an activity ID the method lists all the comments up to a number of results specified by maxResults, that have been posted regarding this activity in an ascending or descending order regarding best-ranked or recent-ranked order. If the number of comments is more than 20 (the maximum value for maxResults) then a	Published, actor, content
		maxResults	unsigned integer	YES	The maximum number of activities to include in the response, used for paging. Acceptable values are 1 to 20, inclusive. (Default: 10)		

		pageToken	string	YES	The continuation token, used to page through large result sets.	pageToken is also returned that can be used to go through the next page of results in the next list request.	
		sortOrder	string	YES	The order to sort the list of comments. Acceptable values are: "ascending" i.e., oldest first (default) and "descending"		
PEOPLE							
listByActivity	People	activityID	string	NO	The ID of the activity to get people for.	Given an activity ID and a collection, which is either plusoners or resharers, the method returns a list of people in those collections.	Name,birthday, string, gender, aboutMe, currentLocation, relationshipStatus
		collection	string	NO	“plusoners” or “resharers”		
		maxResults	unsigned integer	YES	The maximum number of people to include in the response, used for paging. Acceptable values are 1 to 20, inclusive. (Default: 10)		
		pageToken	string	YES	The continuation token, used to page through large result sets.		

get	Person	userId	string	NO	The ID of the person to get the profile for.	The method is used to get People information by giving a user's id. This can be used for the People that post an Activity or a Comment.	

3.6.4 Authorizing requests

This sections describes the authorization process in Google+ . The benefit of this process that is present in all Social Media is that it can provide access to a larger amount of data. Thus, the capabilities of authorized and unauthorised requests are examined separately in each selected platform to drive the decision of whether this process should be foreseen in the NOMAD development.

Google+ API uses the OAuth2 protocol for authentication and authorization. In a nutshell, one needs to (a) register their application with Google, through the Google API console, (b) redirect a browser to a URL, (c) parse a token from the response, and (d) send the token to the Google API they wish to access.

Registering an application generates a set of values that are known to both Google and the application. Before the application can access a Google API, it must obtain an access token that grants access to that API. Obtaining access token requires user consent. After an application has obtained an access token, it may send the access token in a request to a Google API. Access tokens are valid only for the set of operations and resources described in the token request. Access tokens are sent to a Google API in the HTTP Authorization header, or as a query string parameter (if HTTP header operations are not available). Access tokens have a limited lifetime and, in some cases, an application needs access to a Google API beyond the lifetime of a single access token. When this is the case, your application can obtain what is called a refresh token. A refresh token allows your application to obtain new access tokens.

The only method from the ones described above that requires user's consent is the *get* method on the *user* resource type. Hence it is yet uncertain if this method is going to be used by the data acquisition framework in order not to risk the non-moderated character of the project by asking access to private users' data.

3.7 LinkedIn

3.7.1 Description and Basic Concepts

LinkedIn is a social networking platform mainly used for professional networking. As of June 2012, LinkedIn reports more than 175 million registered users in more than 200 countries and territories in all occupations.

People who have joined LinkedIn create a profile and fill it with their interests, qualifications, knowledge, past jobs and more. Registered users can connect with people by searching their name or organization, on the condition that they have stated how they know the person they are connecting to. Also there is a default 'Friend' option that doesn't require to mention how do someone knows the person he is connecting to. Also LinkedIn provides the option to share a status update and also integration with Twitter. LinkedIn also gives the option to like, comment or share an Update or a Group, or share an Update. LinkedIn also provides the ability to create or join groups such as Alumni groups, Corporate groups etc. Although, from the first analysis LinkedIn seems not to offer useful, extractable content for the project, there are some interesting elements, such as Status updates, Comments on Status Updates and likes on Status updates that may provide potential to be exploited. Therefore, in order not to exclude the potential to be added at the NOMAD sources in a later stage, its capabilities is analysed herein. It is notable that various information about a user that has commented, shared or liked an update can be retrieved as well.

3.7.2 Overview of LinkedIn API

Like the above Social Media Platforms, LinkeIn's API[10] has a RESTful API design that enables every resource to be obtained via HTTP requests.

To give an example the current logged in user's profile can be obtained by requesting to

- `http://api.linkedin.com/v1/people/~.`

The response can be either in XML or JSON, an option which is specified in the http request.

So after a LinkedIn user has liked or commented to an update, his personal information can be obtained and use for further analysis, such as what is his educational background, where he has lived etc. To access his full profile, extra permissions must be granted. Also, every profile or network update in LinkedIn has a key that unique identifies it.

Table 15: Overview of LinkedIn API

LinkedIn API	
Version	REST API
Data Format	JSON, XML
Calling styles	REST from Java, REST from Javascript, REST fromPHP, REST from Python, REST from Objective C, REST from Ruby, REST from Clojure
Authorization protocols	OAuth 2.0 / API key
Base URI	http://api.linkedin.com/v1/

3.7.3 Methods for retrieving content

Although LinkedIn is not going to be used within the project, we identified methods that can be useful under the same approach followed for the rest platforms. The following table describes three different methods related to LinkedIn statuses: comment, likes and users created them (*Get Status updates comments, Get Status updates likes, Get characteristics of a user's profile).*

[10] http://developer.linkedin.com/rest

Table 16: LinkedIn API methods for retrieving content

Method	Resource Type	Parameters				Requires	Description	Returned Data
		Parameter	Type	Optional	Description			
STATUS UPDATE								
Get Status updates comments (STAT Updates)	comments	NETWORK UPDATE KEY	String	NO	*update/update-key* representing an update	Api key, Secret Key, OauthuserToken, Oauth user secret, rw_nus	{NETWORK UPDATE KEY} is a*update/update-key* representing an update. Returns all comments belonging to this network update http://developer.linkedin.com/documents/commenting-reading-comments-and-likes-network-updates	Id, timestamp, Person, Comment, Likes
		update-comments	String	NO	The parameter to receive comments			
Get Status updates likes (STAT Updates)	likes	NETWORK UPDATE KEY	String	NO	*update/update-key* representing an update	Api key, Secret Key, OauthuserToken, Oauth user secret,	{NETWORK UPDATE KEY} is a *update/update-key* representing an update. Returns all comments belonging to this network update http://developer.linkedin.com/documents	Num likes, likes, like, person

		update-comments	String	NO	The parameter to receive comments	rw_nus	/commenting-reading-comments-and-likes-network-updates	
				USER				
Get characteristics of a user’s profile	profile	id	String	NO	The user’s id	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	This field might return a value of *private* for users other than the currently logged-in user depending on the member's privacy settings https://developer.linkedin.com/documents/profile-fields	Id, location (name), location (country), num_connections, headline, industry, positions, interests, languages, aducations, date_of_birth
		location:(name)	String	YES	Generic name of the location of the LinkedIn member, (ex: "San Francisco Bay Area")	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	Generic name of the location of the LinkedIn member, (ex: "San Francisco Bay Area")	

		location:(country:(code))	String	YES	country code for the LinkedIn member	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	Lower case values as defined by ISO 3166-1 alpha-2 standard.	
		num-connections	String	YES	the # of connections the member has	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	Available in some places, such as **/people/~/connections,** when *connections* is not. More efficient than checking the *total* attribute of *connections* even when *connections* is available.	
		headline	String	YES	the member's headline (often "Job Title at Company")	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	the member's headline (often "Job Title at Company")	
		industry	String	YES	the industry the LinkedIn member has indicated their profile belongs to (Industry Codes)	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	the industry the LinkedIn member has indicated their profile belongs to (Industry Codes)	

		positions	String	YES	A collection of positions a member has had, the total indicated by a *total* attribute	Api key, Secret Key, OauthuserToken, Oauth user secret, r_basicprofile	A collection of positions a member has had, the total indicated by a *total* attribute	
		interests	String	YES	A short-form text area describing the member's interests	Api key, Secret Key, OauthuserToken, Oauth user secret, r_fullprofile	A short-form text area describing the member's interests	
		languages	String	YES	A collection of languages and the level of the member's proficiency for each	Api key, Secret Key, OauthuserToken, Oauth user secret, r_fullprofile	A collection of languages and the level of the member's proficiency for each	
		educations	String	YES	A collection of education institutions a member has attended, the total indicated by a *total* attribute	Api key, Secret Key, OauthuserToken, Oauth user secret, r_fullprofile	A collection of education institutions a member has attended, the total indicated by a *total* attribute	

		date-of-birth	String	YES	member's birth date	Api key, Secret Key, OauthuserToken, Oauth user secret, r_fullprofile	May return only month and day, but not year, or all three, depending on information provided.	

3.7.4 Authorizing requests

LinkedIn API uses OAuth 1.0 a protocol. The flow of authorizing a request is:

a. Register your application
b. Get api key and secret
c. Grant member permissions to the application
d. Refresh access tokens if they expire

LinkedIn does not support any unauthenticated requests and this is one of the reasons of reaching the decision not to be crawled by the NOMAD tools for the time. This should be reconsidered in case the structure of LinkedIn API is changed.

4. SEMANTIC INTEROPERABILITY BETWEEN WEB 2.0 SOCIAL MEDIA PLATFORMS

4.1 Interoperability Analysis

For the purposes of our project, there is not need of technical interoperability between the Social media data sources. This is because interaction between sources of content is absent from the NOMAD approach. As it will be described in technical deliverables, NOMAD crawlers will acquire content from each of the above applications individually. Data from each application are merged in a later stage, unified and imported in a common repository. So, what is actually needed is the integration of available data for the establishment of an inter-platform communication. In the current section we prescribe a semantic interoperability between them that will enable this unification of information. This actually refers to a mapping between the data fields that are retrieved and are named differently by each API.

The following table s form an attempt to correlate the outcome of each API call identified in the previous chapter, around the Social Media Platforms. The list of data fields is restricted to the information that is considered as valuable for the project via the specification of the User Requirements. Therefore, herein we focus only to the two basic resources for the project, which are users to obtain their profile data and comments as the basic element of NOMAD processing.

Table 17: Semantic Interoperability between User Data

USER DATA						
Facebook	**Twitter**	**LinkedIn**	**Wordpress / Gravatar Profile**[11]	**YouTube**	**Blogger**	**Google+**
id	id	id	id	id	id	id
Gender	-	-	-	Gender	-	gender
locale	-	-	-	-	locale	-
languages	-	languages	-	-	language	-
Bio	description	Skills, certifications, educations, courses, summary, headline	-	Occupation, about me	about	About me
birthday	-	date-of-birth	-	age	-	birthday
education	-	educations	-	school	-	-
work	description	three-current-positions, three-past-	-	occupation	-	organizations

[11] The Wordpress user's data are stored in a Gravatar profile. Matching depends on the associated Social Media verified account is stored in the Gravatar profile.

		positions				
religion	-	-	-	-	-	-
location	location	location:(name), location:(country:(code))	-	location	country	currentLocation
Hometown	-	-	-	hometown	-	-
Relationship_status	-	-	-	relationship	-	Relationship_status
	Followers_count	Num_connections	-	-	-	-
work		industry	-	-	-	organizations
-		interests	-	hobbies	about	About me
-	Follow_request_send	-	-	-	-	-
-	Geo_enabled	-	-	-	-	-
-	Statuses_count		-	-	blogs	-
timezone	timezone	-		-	-	-
website	url	-	-	-	-	urls

As shown in the above table, Wordpress doesn't retain any information about its users. Thus, to obtain data about Wordpress users we can only make calls to the Gravatar API to find an associated verified Social Media account through the process described analytically in the previous chapter. User may have associated different types of accounts in his/her Gravatar profile (Facebook, Twitter, etc.) and thus we cannot be sure what kind of data could NOMAD retrieve. That is the reason why user's data fields cannot be included in the mapping.

It is also notable that the availability of the user's data depends on the access rights defined by their owner. In the next section we explain the process of authentication that provides an option to access additional data.

Table 18: Semantic Interoperability between Comment Data

COMMENT DATA						
Facebook	**Twitter**	**LinkedIn**	**Wordpress**	**YouTube**	**Blogger**	**Google+**
id	id	id	id	id	id	id
from	user	person	author	author	author	actor
message	text	comment	content	content	content	content
created_time	created_at	timestamp	date	published	published	content
likes	retweet_count	-	-	-	-	-

4.2 Authorization protocols

Authentication and authorization of client applications to obtain access to the resources of the Web 2.0 social media platforms described in the sections above is performed through the OAuth protocol. LinkedIn uses OAuth 1.0, while the remaining of the platforms use OAuth 2.0. A detailed description of the OAuth 2.0 protocol can be found at http://tools.ietf.org/html/draft-ietf-oauth-v2-22.

In the traditional client-server authentication model, the client requests an access-restricted resource (protected resource) on the server by authenticating with the server using the resource owner's credentials. In order to provide third-party applications access to restricted resources, the resource owner shares its credentials with the third party. The OAuth protocol introduces an authorization layer separating in this way the role of a client and a resource-owner. According to the OAuth protocol a client requests access to resources is controlled by the resource owner. The client is granted an access token – a string with specific scope and lifetime – by an authorization server with the approval of the resource-owner. The client can then use the access token to access the protected resources hosted by the resource server.

OAuth 2.0 authorization is used for client applications to access private data and it requires the owner of the private data to grant permission to the application. For the purpose of the NOMAD project however we obtain data that are publically available. We use the OAuth2.0 protocol to authenticate the client application but we do not ask permission to obtain private data. Given that our data acquisition methods depend on searching in posts and comments and they do not focus on a limited number of resources owned by specific users requests to access private data would result to a large number of Web 2.0 platform users having to authorize our applications. However this is a matter to be reexamined according to what amount of data is publicly available especially with respect to demographics needed for the categorisation and visualisation.

The process of a client application authentication and obtaining access to (public) data through the OAuth protocol is the following: one needs to (a) register their application, (b) redirect a browser to a URL, (c) parse a token from the response, and (d) send the token to the API they wish to access. Before initiating the protocol, the client registers with the authorization server. This typically involves end-user interaction with an HTML registration form, e.g. in the case of Google APIs, this is done via the Google API console. A client ID and client secret is obtained through the registration. These can be used in the request to the authorization server for an access token. The application is authenticated by redirecting the browser to a url and using credentials (username and password) of the user that registered the application. For this purpose we have created a NOMAD project user in most of the aforementioned platforms. The access token obtained through this process can then be used in every request to the resource server to obtain resources.

5. STATE OF PLAY ON SOCIAL MEDIA CONTENT ANALYSIS

5.1 Introduction

Before proceeding to the analysis of the selected platforms APIs, and in order to set the technical requirements for NOMAD, it is essential to have an overview on what tools embracing the "crowdsourcing" concept already exist, what capabilities they offer and how they capitalize on the Social Media APIs for content search and acquisition, categorisation and visualisation. Since crowdsourcing has been exploited in the business world for multiple functions of firms, such as market research, customer support, sales management and crisis identification management, many tools have emerged that harness the social engagement in Web 2.0 and the wealth of user generated content [4][5]. Therefore, the current section presents a State of the Art analysis concerning software tools performing similar to the NOMAD approach functions, such as tools for Social Media Monitoring, Opinion Mining and Text Analytics, Digital Reputation Management, etc. Although they are used mainly in the private sector, they could form NOMAD competitors. The objective of this analysis is, not only to identify a number of existing competing choices, but also to provide some useful insights from this comparative analysis regarding the features that should be supported by NOMAD and have to be taken into account in the system design.

In the current study we provide a list of 60 major social media monitoring platforms and opinion mining tools available either free or commercially. The list is not exhaustive, but consists of the strongest players, in the market so-called "Social Media Monitoring and Analytics" whose functionality is pertinent to the NOMAD project objectives. This new emerging mainly marketing research field refers to the "*tracking or crawling of various social media content such as blogs, wikis, news sites, micro-blogs, social networking sites, video and photo sharing websites, forums, message boards, blogs and user-generated content in general as a way to determine the volume and sentiment of online conversation about a brand or topic*"[12]. The proliferation of such tools is due to their capabilities for extracting and providing analytics about opinions, sentiments, arguments and judgments about people, companies, brands and products found in blogs, forums and Social Media sites. According to Stavrakantonakis et al. [4], their added value lies that on the speed that they can offer these investigations in comparison with the traditional methods, at real time and in a highly scalable way.

As already mentioned, the aim of the current investigation is to undertake a review of other approaches on monitoring Social Media conversations under a technical viewpoint, and incorporate emerging conclusions in the NOMAD approach. Thus, includes an overview of the features provided and technologies employed by the existing solutions in this area and an analysis of important characteristics, such as scalability, real time processing or response time, etc. Due to the project needs, we emphasized as well on content analysis and visual representation of results.

5.2 Overview of Social Media Monitoring Tools

As a first step of our methodology, we tried to identify the whole spectrum of existing tools to be examined. To find and browse relevant online information we searched on the web based on a set of appropriate keywords or phrases, such us "*social media monitoring*", "*text mining tools*", "*social media analytics*", etc. A considerable amount of various environments, incorporating different characteristics, resulted from the desk research conducted on scientific publications, marketing reviews and online surveys[1][3].

In the initial observation, we tried to identify the common approach between these tools. It was observed that although they provide different capabilities, there are some standard functionalities present in the majority of them:

1. Extract content from a pre-defined list of Web 2.0 sources using either Social Media APIs or web scrapping mechanisms.
2. Perform some kind of data processing according to the features provided in each environment (Sentiment Analysis, Issues detection, Top influencers identification, etc.).
3. Extract additional information in terms of analytics, for example demographic distribution of results.
4. Visualise the current results often in an interactive or customisable environment.

[12] http://en.wikipedia.org/wiki/Social_media_measurement

5. Provide reporting capabilities to the users to enable them export, save and reuse this information.

The above functionalities are generally used sequentially. Thus, they can be translated in the five steps for analysing content acquired from Social Media. The following figure visualises the process following these steps and which is also similar with the NOMAD cycle. This cycle represents a feedback loop as user can return to previous steps and refine results.

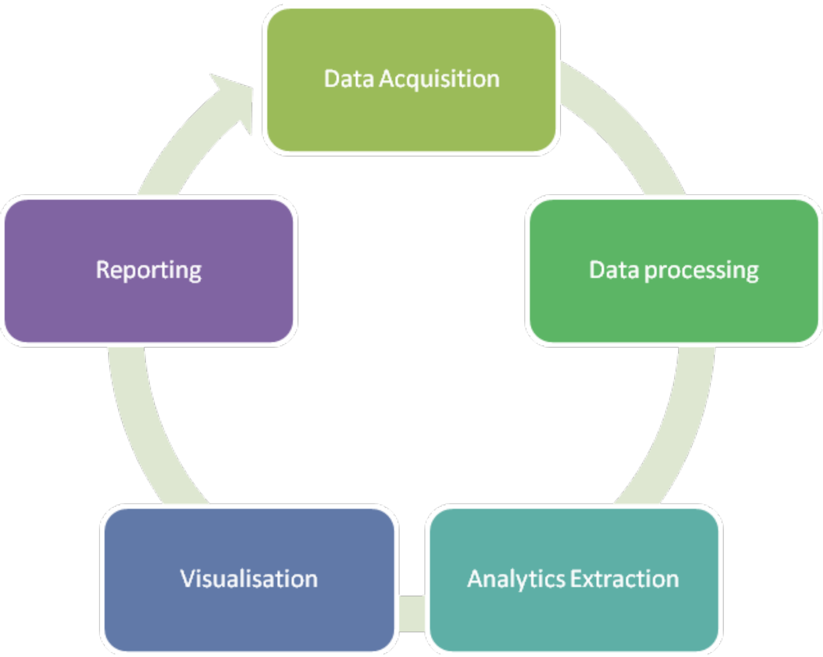

Figure 3: The five stages of Social Media Content Analysis

According to the different approaches they follow, we clustered the list of tools in four major categories:

1. **Social Search Tools**

Social Search tools embody search engines that serve search queries on keywords and provide metrics on their occurrences across various Web 2.0 platforms (number of occurrences, top users mentioning the search term, relative hashtags, sentiments, mentions per specific time intervals). In their simplest form, the user enters any search term and receives real time results on where it is mentioned. Some of them also include traditional media and opinion sites in their data sources.

Topsy *Crawls social web and provides real time insights from billions of conversations for specific time frames on the past.* *http://topsy.com*
Socialmention *Real-time social media search and analysis platform that aggregates information from 100+ sources into a single stream, displaying sentiments, keywords, statistics, top users, top hashtags and top Sources.* *http://socialmention.com*
25 trends *Provides Twitter Analytics in terms of popular tags, relevant tweets, top users, languages, links, etc.* *http://25trends.me*
Twazzup *Operates as a real-time news platform and real-time Twitter analytics service obtaining data through its API and external applications, such as Twitterfeed, TweetMeme, TweetDeck, etc..* *http://www.twazzup.com*

Whostalkin *A Social Search tool that allows users to search for conversations across 60 of the internet's most popular social media gateways.* *http://www.whostalkin.com*

Addict-o-matic *Draws results for the latest buzz on any topic from the top 200 blogs and news sites on the web.* *http://addictomatic.com*

Boardreader *Web application that searches for activity on the topic for a time period and displays threads, sites, relevant topics from multiple web sources such a online forums, message boards, blogs, news sources, and videos.* *http://boardreader.com*

Monitter *A Twitter Search tool for monitoring a set of keywords on twitter. It also allows the user to narrow the search to a particular geographic location, to find out what's discussed on a topic in a particular part of the world* *http://monitter.com*

Slidebar Monitor *Allows comprehensive monitoring of Greek Social media discussions and reporting combined with users' demographics, through advanced search capabilities.* *http://monitor.sidebar.gr*

Sync3 *A news aggregation and analysis tool with event labeling and news clustering delivered by research project*. *www.sync3.eu*

2. Twitterverse Tools

The term Twitterverse refers to the cyberspace area of twitter[13] including each users and their habits. Therefore, this cluster refers to services that acquire data exclusively from Twitter and are designed based on capabilities that Twitter offers for leveraging information. They are applications with simple graphical interface that offer individual services by extracting specific content from twitter messages, such as location – based visualisation of latest tweets, volume of tweets on specific keyword, daytime frequency of tweets, etc. However, a whole subcategory "Twitter Sentiment Analysis" has emerged here, as many of them implement sentiment analysis algorithms to garner the emotion within tweets.

Tweet Archivist *Service for monitoring Tweets. Usesr can search any term on Twitter and create archives form the day the archive is created and later (archivist cannot go back in time)* *http://www.tweetarchivist.com*

Twitter Earth *Service for visualizing location of tweets on 3D Maps, integrated with Google Earth.* *http://www.twitter-earth.com*

Xefer *Produce charts showing a Twitter user's tweet statistics per Time of Day & Day of Week. Data is gathered and generated using Yahoo Pipes and the Google Chart API.* *http://xefer.com/twitter/*

Twitrratr *Displays list of positive or negative Tweets against search terms referenced in tweets containing polar words.* *http://twitrratr.com/*

Sentiment140 *(formerly known as "Twitter Sentiment") allows user to search for the sentiment of a brand, product, or topic on Twitter in English or Spanish* *http://www.sentiment140.com/*

TweetFeel *Real Time Twitter Search monitoring positive and negative feelings in twitter conversations and displaying them in a simple way.* *http://www.tweetfeel.com/*

Twends *A web application for twitter mining conversations (sentiment and themes) that utilises Twitter Search to evaluate tweets that are dynamically updated, minute by minute (limit of 70 tweets per time).* *http://twendz.waggeneredstrom.com/*

Twitter Weather *Positive and negative feelings on twitter about a topic are mapped to a "temperature" from 0 to 100.* *http://twitterweather.media.mit.edu/*

Twitter geoweather *Location based visualization of tweets sentiment on a topic.* *http://twitterweather.media.mit.edu/geolocation/*

[13] http://www.urbandictionary.com/define.php?term=twitterverse

Twitter mood *Captures the mood of all tweets for which the author has provided a location on the Map of United States. http://www.twittermood.org/*

Monitter *This tool helps the user to monitor a set of keywords on twitter. It also allows narrowing the search to a particular geographic location, allowing the user to find out what's going on in a particular part of the world. http://monitter.com/*

Political Twitter Sentiment *Public sentiments toward the 2012 United States presidential candidates as expressed through Twitter. Provides Statistics, Tweets, Words. http://politics.twittersentiment.org*

Twelect *Mobile Application, which present poll results for Elections extracted from twitter messages. http://www.twelect.com*

3. Social Media Measurement Tools

The specific tools offer mechanisms for measurement social media influence and produce metrics based on performance across most popular social media with the ultimate goal to identify trending topics and influencers. They differentiate from the rest categories, since they focus on impact assessment rather than more general social media analytics.

Klout *The Klout Score (a number between 1 and 100) measures users' overall social media influence and shows how they impact the people connected to them across several social networks. http://klout.com*

Twitris 360° Social Media Analysis *A Semantic Social Web application with real-time monitoring and multi-faceted analysis of social signals to provide insights and a framework for situational awareness, in-depth event analysis and coordination, emergency response aid and reputation management. Use case scenario on US President Elections 2012 available. http://twitris.knoesis.org/election*

Starcount *Provides a global popularity leaderboard that creates dynamic daily charts with public profiles. Starcount score measures social media engagement and popularity through public data across 11 top social networks. http://www.starcount.com*

Trending *Service that analyzes in everyday basis Greek Tweets and records interesting statistics like trending topics, re-tweeted messages, top users, top links. It can also provides statistics on specific Greek Twitter users through its Search Engine. http://trending.gr*

WildFire Social Media Monitoring *Service that compares the performance of Facebook and Twitter accounts (number of likes, check-ins, and followers each page boasts). http://monitor.wildfireapp.com*

BuzzFeed *A platform for social news organisation detects viral content with an editorial selection process and reports what is trending on the web across various subject areas in real time. http://www.buzzfeed.com*

Pulse of the Tweeters *This tool uses a trend-specific ranking scheme for users who shape Twitter discussions and is able to identify the top trending topics in real-time and the most influential users across each topic. http://www.pulseofthetweeters.com*

PeerIndex *Connects to user's social media (Twitter, Facebook, LinkedIn) and give scores users to show their online influence and impact, identifies the opinion leader in a particular niche. http://www.peerindex.com*

4. Social Media Monitoring Dashboards

The last cluster of tools combines features from the aforementioned categories. These tools are the most complicated, since they offer a full experience of Social Media Management including capabilities of handling multiple accounts, engaging with audience, posting content in multiple accounts simultaneously, track real word reactions, monitoring social media marketing campaigns, managing online reputation and assess impact across a full range of social media platforms. All elements of this cluster are commercial, since they provide comprehensive solutions with multilingual support.

Radian6 *Fully integrated environment for Social Media Monitoring and Engagement, Provides results with Demographics, location and Trending Topics in a customizable dashboard. Examples of various case studies are available.* *http://www.radian6.com/*

Sysomos *A social media analytics suit that provides users with the tools to measure, monitor, understand and engage with the social media landscape.* *http://www.sysomos.com*

Brandwatch *Tool for monitoring and analysing social media data, that emphasizes in cleaning and matching data against users' advanced queries.* *http://www.brandwatch.com/*

Converseon *Offers a full service, combining automated and human analysis, of social media conversation to create the deep insights required to drive business objectives.* *http://converseon.com/*

Cymfony Maestro *An enterprise-class listening platform for monitoring and measuring Social Media performance that sorts the vast amount of information online, giving clients real-time access to the most comprehensive archive of traditional and social media.* *http://www.kantarmediauk.com/cymfony/maestro.aspx*

evolve24 Mirror *A market intelligence platform for supporting business decisions and strategic planning that uses predictive algorithm to forecast future behaviors and perceptions acquiring data from both traditional and social media.* *http://www.maritzresearch.com/mirror-2.0.aspx*

Media Metrics socialMeme *A browser-based SaaS solution providing a detailed overview of opinions expressed on all media channels (Social Web, TV, radio and print) in real time.* *http://www.en.high-tech-gruenderfonds.de/2012/03/mediametrics-gmbh-high-tech-grunderfonds-invests-in-intelligent-media-monitoring-solution-socialmeme*

NM Incite My BuzzMetrics *A scalable, web-based social insights platform designed for Fortune 1000 marketers and their agencies to organize, segment and analyze clean, trusted, industry-specific global insights in real time.* *http://nmincite.com/solutions/my-buzzmetrics*

Mutual Mind *Aims to extract business value from social interactions and enable user to integrate social analytics to existing business applications.* *http://www.mutualmind.com*

Synthesio *Social Media Monitoring and Engagement tool focusing in customers relationship management supporting 50 languages.* *http://synthesio.com*

Attensity360 *SaaS "listening and engagement system" monitoring customer's voices across multiple channels and providing actionable insights.* *http://www.attensity360.com*

SimpleMeasured *Social Media Analytics and Reporting Tool* *http://simplymeasured.com*

Trackur *Tool for monitoring online reputation to get the buzz from any web content that contains brand name* *http://www.trackur.com*

Visible Intelligence *Targeted towards improvement of Business Decisions through Social Media Monitoring, Analytics and Engagement.* *http://www.visibletechnologies.com*

Lithium *Offers multiple solutions for analyzing and managing activity across social media channels.* *http://www.lithium.com/products*

NetVibes *A full customizable and personalized dashboard publishing platform integrating various types of content on the Web.* *http://www.netvibes.com*

Alterian SM2 Freemium *Provides strategic decision support and listening consultancy through customised services, delivering social media insights and analytical tools.* *http://www.alterian.com/socialmedia*

Market Sentinel Live Buzz *Analyzes and measures online commentary to inform about marketing strategy and help businesses take better decisions* *http://www.sentinel-projects.com*

uberVU *End-to-end social intelligence dashboard that covers all 4 social media value pillars: Monitoring, Analytics & Reporting, Engagement, Workflow.* *http://www.ubervu.com*

Beevolve *Social Media Monitoring and Measurement Platform that integrates services for Tagging, Engagement, Analysis and Collaboration.* *http://www.beevolve.com*

Hootsuite Provides a full dashboard to handle multiple Social Media streams for free. User A Report Builder is offered to users through subscriptions to give them access to advanced analytics on their associated accounts. *http://hootsuite.com/*

Subsequent to the classification of tools, an attempt was made, to set a number of criteria to perform the comparative analysis among them. The characteristics were selected to unfold the analysis on three axes: to observe and compare the approaches behind existing mechanisms in relation with the corresponding NOMAD components, to find out which key technical features are present in the majority of tools, and finally to identify "must-have" capabilities for the NOMAD user. Apart from these, some general characteristics were included in the analysis, which overall was based on the following features:

General characteristics:

- *Commercial VS Free:* the analysis contain both free available and commercial (paid) solutions to discover what the limitations of the first and the advantages of the second are.
- *Application Field:* the application's scope in terms of what is the usual purpose of using it (marketing research, brand monitoring, political usage, etc). Here we investigate how much it does converge with the policy domain oriented NOMAD needs.

Technical Components:

- *Visual Analytics:* the variety of visualisation means offered and how these are capable of capturing insights for the policy domain and simultaneously being readable from a simple user such as a policy maker or advisor.
- *Opinion Mining and Sentiment Analysis:* the presence of integrated modules to analyse content and discern sentiments in the social media corpus and how these are implemented.
- *Data Sources:* the range and plurality of Web 2.0 platforms from which the applications extract content (Blogs, Micro – Blogs, Social Networking Sites, Video and Photo sharing sites, Forums, Wikis, etc.)
- *Linguistic analysis*: the different languages supported in cases where linguistic processing applies.

Technological features

- *Response Time:* the time interval that elapses until the system provides results on the user's input. Some of the tools show real time results, while other may require some time for processing (usually a day).
- *Interoperability***:** refers to the capability of the tool to integrate with 3rd party applications either through API exposure or through widgets embedding their services.

The analysis of the commercial tools was conducted mainly through the information extracted via their vendors' official websites, whereas in the publicly available tools experimentation on specific use cases took place in order to observe and compare the provided results. The findings of the analysis are summarized in the following table.

Table 19: Review of Social Media Monitoring and Online Reputation Tools

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
Klout	F	Area Charts, Pie Charts, Metrics	-	Social media sites, Wikis	X (Politician's Klout Scores)	All	1 day	API, Widgets
Twitris	Demo for U.S. Presidential Elections 2012	Tag Cloud, Social Network Analysis, Map, Line Charts, Spatial Markers	X	social data, web resources (news, Wikipedia pages, multimedia), SMS data,	X	English	Real Time	
Topsy	F/C	Line Charts	X	Social media sites, Websites	X	All	Real Time	API
Radian6	C	Terms Cloud, Line Chart, Pie charts, Map, Bar Charts	X	Blogs, Social Media, Photo, Video, Forums, News sites		17 (Chinese, Danish, Dutch, English, Finnish, French, German, Italian, Japanese, Korean, Norwegian, Polish, Portuguese, Russian, Spanish, Swedish and Turkish)	Real Time	API
Sysomos	C	Pie charts, Line charts, Tables, Word Clouds, Bar Charts, Graph, Speedometer	X	Blogs, Forums, Twitter, YouTube, Wikis, News sites		Translation on the fly	Real Time	API
Brandwatch	C		X	Twitter, Blogger, WordPress, Typepad, LiveJournal, YouTube, Vimeo, Metacafe, Bliptv, Facebook, MySpace, Discussion forums, News sites, Flickr, Corporate sites		Multilingual (including English, French, German, Italian, Spanish, Dutch,	Near Real Time	X

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
				and more		Swedish, Danish)		
Converseon	C	Online Dashboards	X	Blogs, Twitter, Message boards, Social networks and more		Multilingual	Real Time	API
Cymfony Maestro	C	Google Charts	X	Traditional Media, Facebook, Twitter, Youtube, Blogs, Forums, Social Networks, Microblogs, Consumer Review Sites		14	Real Time	API, Widgets
evolve24 Mirror	C	Interactive Dashboards	X (Machine Learning techniques)	Both Social and Traditional media		30	Near Real Time	-
Media Metrics socialMeme	C		X	100 million+ online sources		48	Real Time	-
Meltwater Buzz	C	Word Cloud, Histograms, Graphs and Charts to determine trends, map press activity and identify target markets.	X	Blogs, Micro-blogs, Social networks, Forums, Video and Photo websites, Product reviews and other Social media sites		Multilingual	Real Time	-
NM Incite My BuzzMetrics	C		X (NLP techniques and DATA CLEANSING process)	Social Media Platforms such as Facebook, Blogger, LinkedIn, YouTube, Twitter, Yahoo Groups, Amazon Reviews, Flickr, ...)		15	Real Time	-
Synthesio	C	Charts	X (NLP techniques & human analysis)	Social media channels (like Orkut, Seina Weibo, RenRen and more)		50	Real Time	-
Attensity360	C	Charts	X	review sites, blogs, forums (user forums, discussion forums, LinkedIn Answers, etc), Twitter, Facebook, YouTube videos, mainstream news and more				Widgets

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
SimpleMeasured	C	Charts	-	Google+, Facebook, Twitter, Klout, YouTube, Instagram, Vimeo				-
Socialmention	F	Bar Charts	X	100+ Social media (Twitter, Facebook, FriendFeed, YouTube, Digg, Google etc.)		Multilingual	Real Time	API, Widgets
Twitrratr	F	Metrics	X (list of polar words)	Twitter		English (United States)		-
Twitter Sentiment	F	Pie Charts, Bar charts	X	Twitter	X	English, Spanish	Real Time	API
Twends	F	Word visualization based on a series of packed circles. The larger the circle and the redder it is, the more frequently the word is being said at the moment.	X	Twitter			Real Time (updated per minute, up to 70 tweets at a time)	-
Tweet Archivist	F/C	Pie charts, Line charts, Bar Charts	-	Twitter		All	Real time	API
TweetFeel	F	Statistics	X	Twitter	X	English	Real Time	API
Twitter Earth	F	Map (Google Earth) / Need downloading Google Earth Plugin	-	Twitter		English, Spanish		
Twelect	Mobile application for Elections	Linecharts	X	Twitter	X (Polls)		Real Time	NA
Twitter Weather	F	Weather visualisations mapped by location of tweets on a topic	-	Twitter			-	-
Twitter Geoweather	Demo for USA	Map visualisations	X	Twitter			Real Time	API

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
25 trends	F	Tree Map	X (topics + sentiments detection)	Twitter	X	English, Arabian	Real Time	API
Political Twitter Sentiment	Demo for US Election canditates	Linecharts, Barcharts, TagCloud	X	Twitter	X	English	Real Time (updates every 30 seconds)	API
Twitter mood	Demo for USA	Map	X	Twitter			Real Time	
Trending	F	Histograms, Barcharts	-	Twitter	X	Greek, English	Real Time	Widgets
Starcount	F	List of charts	-	Posts, pictures, videos in Social Media Platforms such as (Facebook, Google +, Twitter, Orkut, Kontakte, Renren, Mixi, YouTube, Youku, Weibo, QQ) and from other services (blogs, wikipedia, news reports).		English, Russian, Chinese	1 day	API
Trendpedia	F			Blogs				-
Xefer	F	Google Charts	-	Twitter	X	6 (English, Dutch, Espaniol, Francais, Italiano, Japanese, Russian)	Real Time	X
Trackur	C	Google Charts	X	Blogs, Forums, Twitter, Facebook, Google, News sites, Klout		English	1 day	API
Mutual Mind	C	Analytics dashboards	X	Blogs, forums and news sites, Social networks, Facebook, Twitter, YouTube, Flickr and more			Real Time	API
Lithium	C	Graphs	X	Twitter, Facebook, Blogs, Online Communities			Real Time	API
Visible Technologies	C		X (granular categorization,	Social Media		50+	Real Time	API

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
			sentiment measurement)					
Market Sentinel Live Buzz	C	Graphs, Charts	X	Blogs, Forums, Chat rooms, Mainstream media				
Addict-o-matic	F	NA	-	Social Media, News and Blogs,	X	English (United States)	Real Time	Search Pugin
NetVibes	F/C	Trend graphs	X	Social Media, News and Blogs, Google, Yahoo, Technorati, Ask, YouTube, Truveo, Flickr, Blinkx, Ice Rocket, Digg, Topix, Newsvine and Tweetscan.		nearly 50 (including 100% native UI versions in Japanese, Spanish, English and French)	Real Time	API
Twazzup	F	Graphs, Charts	X	Twitter	X	English, Japanese, Greek, Danish, Arabic, Dutch, Finnish		-
Whostalkin	F		-	Blogs, Forums, Social networks	X	English	Real Time	iGoogle Gadget, Browser Search plugin
Evolve 24	C	Multiple actionable visualisations	X	Social Media, Blogs, message boards, online news sites, product rating sites			Real Time	-
TrendMiner	R&D Project	Line Chart	X	Weblogs, Twitter, Facebook	X	English,German, Italian, Bulgarian,	Real Time	-
WildFire Social Media Monitoring	F	Charts	-	Facebook (Likes, Check-ins), Twitter (Followers, Following, Tweets)	X		Real Time	-
boardreader	F	Linechart	-	Forums, message boards, blogs, news	X	English	Real Time	API

Tool	Commercial / Free	Visual Analytics	Opinion Mining and Sentiment Analysis	Data sources	Political Usage	Languages Supported	Response Time	Interoperability
				sources				
BuzzFeed	F		-	News, Micro-blogging	X	English	Real Time	API
Alterian SM2 Freemium	C	Charts	X	Message boards, review sites, blogs, news channels, niche networks, Facebook, Twitter				-
Monitter	F / Widgets paid		X	Twitter	X	English	Real Time	Widgets
Pulse of the Twitters	F		X	Twitter	X		Real Time	-
PeerIndex	F/C	Metrics	-	Facebook, Twitter, Google +, LinkedIn			7 days	API
uberVU	C	Analytics based on metrics and geolocation	X	Social Media		English	Real Time	API
trackur	C		X	News sites, blogs, forums, Twitter, Google+ and Facebook, Reddit Delicious		English		API
Beevolve	C		X	Twitter, Facebook, YouTube, Blogs, Traditional News Publications and Real-time Web		English	Near Real Time	API
Slidebar Monitor	C	Google charts, Line chats, Pie charts	X	Twitter, YouTube, News sites, Facebook, Blogs, Forums		Greek	Near Real Time (up to 24 hours depending on the social media channel)	-
Sync3	R&D Project	Map chart, Pie charts, Bar charts	X	News Portals, Blogs	X	English	Real Time	
Hootsuite	F/C	Google charts	X	Social Media accounts	X	All	Real Time	API
PADGETS	R&D Project	Google charts	X	Facebook, Twitter, YouTube	X	Greek, German, Slovenian	Near Real Time	

5.3 Conclusions of the analysis

In order to analyse the existing State of the Art, which includes several software options for extracting and processing content from Web 2.0 platforms, we have identified 60 platforms that utilise Web 2.0 content. A great portion of them consist of web applications that provide non sophisticated solutions of Social Search, usually combined with Sentiment Analysis services. For more advanced services integrated commercial solutions are available, which combine different modules for supporting all phases of a Social Media Strategy.

It was observed that there is a large number of tools focusing on Twitter, which is reasonable due to the open access it provides, and the amount of data it makes publicly available. Twitter is considered ideal for measuring current public opinion, since it provides a dataset instantly updated by its users. The majority of tools allow discovering the sentiment for a brand, product, or topic expressed on Twitter, because, due to the short length informal character of the tweets, people are encouraged to express directly their opinions. In the Twittverse Tools that perform Sentiment Analysis, the user submits a search query and triggers the sentiment analysis on the results returned that mention the keyword in tweets.

Data Acquisition is realized mainly via a combined approach, the same that NOMAD should follow using both crawlers and the Social Media APIs. There is also inter-operation among the tools that expose an API, such as Klout and Topsy which are reused as data sources of other ones. Moreover integration with external tools, such as Yahoo Pipes has been observed in some tools to retain data from the beginning of the time when is no longer available in the original Social Media source. For instance, in Twitter Mood, data is collected from the twitter *gardenhose*, which is a stream containing a significant sample of all public twitter statuses and supplies about 1.800.000 messages per day.

Regarding Sentiment Analysis, it consists of an available module in all commercial tools. There are various technical approaches behind these modules. For example, Sentiment 140 uses classifiers built from machine learning algorithms. Some other sites use a simpler keyword-based approach, which may have higher precision, but lower recall and they are not improved over the time. In Twitter Mood valence of each individual tweet is based on the Affective Norms for English Words (ANEW) data set [6]. There are cases that use language-independent approaches in order to cover multilingual sentiment classification, but it has been observed that are less effective than natural language processing methods that are designed for specific languages. What should also be taken into account in the implementation of the NOMAD Opinion Mining component is that Social Media corpus is characterised by peculiarities due to the informality of discussions, often containing emoticons, slang or misspellings, which consists the reason of low accuracy of existing tools. Finally, although there is quite wide range of languages supported by their linguistic processes in total, very few of them support Greek language, which consists of a competitive advantage for NOMAD.

A common tactic for serving the visualisation needs of the tools is integration with Google Chart Tools[14]. Although, the visualisation power in some cases is remarkable, existing techniques are not oriented to the policy context, meaning that they doesn't meet policy maker's needs insights regarding citizen's opinions on a policy topic in an easily – absorbable way. Therefore, it is a challenge for NOMAD to design and introduce innovative Information Visualization and Visual Analytics techniques applicable to the field of policy modelling. An example of this is the time-series visualisation of the sentiments on specific policy arguments for a given time interval, which can show to a policy maker how citizenry's opinion evolves over time in a simple way.

Another set of characteristics that were generally observed in tools refers to the User Capabilities that are present in some tools, and can be valuable for the NOMAD user as well, such as:

- *Search Engine*: the existence of a search engine to enable user to perform search queries based on a keyword, in a similar approach with the NOMAD crawler
- *Export report:* services for the user to export the results (charts, tables, statistics, etc.) in a common file format. NOMAD may copy this approach to export results in excel format or on web, or to notify policy makers via mail alerts.

[14] https://developers.google.com/chart/

- *Demographic analysis:* the means of visually representing analytics accompanied with their demographical distribution, focusing on statistical analysis valuable for policy making.
- *Customisation of the dashboard:* the capability of providing customised views on the results according to user's preferences
- *Summarization / Detailed Data:* the level of depth the presentation of analytics reaches (individual post / user),. In our project is translated as the ability of user to drill down into the results concerning a specific policy argument.
- *Filtering:* the user's ability to select specific data sources to narrow results

Finally, only a small portion, particularly six out of the 60 applications are available on a mobile version and this can be considered as a further exploitation of NOMAD tools.

As a general conclusion it should be mentioned, that this kind of tools are usually oriented to business functions. Only a few use cases (Twelect, Klout politician's score, Twitter Political Sentiment) are oriented to meet exclusively to the needs of policy formulation. NOMAD offers this comparative advantage through the policy modelling feature, providing more complete and aggregated set of results. For example, search queries are performed on a single term in the examined applications instead of a given set of keywords that derive from domain and policy models. NOMAD aims to provide these more sophisticated solutions in the policy making context.

6. CONCLUSIONS

As NOMAD aspires to create a two-way dialogue between citizens and government and empower citizens' role by increasing their participation in governmental decision making, it aims to deliver ways and tools to transform political content produced in social media into valuable information for policy makers. This deliverable, based on the foundations built in the analysis of the underlying content and knowledge of Web 2.0 social media, aims to discover the technical means for the exploitation of citizens' social engagement and guide the implementation of NOMAD tools.

At first, since the project focuses on the places where political discussions take place, a representative set of popular Web 2.0 sources that accumulate content that can be utilised in policy formulation and expose publicly available application programming interfaces, has been selected. Seven Social Media platforms have been identified from which political and public policy related content can be acquired, categorised and visualised. Therefore the deliverable reaches the decision to study Blogger, Wordpress, Facebook, Twitter, YouTube, LinkedIn and finally Google+, with the view to discover the prevailing software modules, data, formats and protocols that NOMAD should support. The rationale behind this choice is, as these platforms belong to the most prominent Social Media categories and follow common standards, their examination will reveal general insights towards embedding also others Social Media APIs functionality in a later stage.

The main point of this deliverable is to examine what capabilities Social Media APIs offer for extracting content that expresses citizens' needs, opinions and comments and could converted into useful knowledge via the NOMAD processing. The analysis conducted in Chapter 3 resulted in some useful findings regarding the programming methods for retrieving content, the kind and structure of data that can be extracted, the output formats and the protocols for acquiring permissions to access this content.

It was generally concluded that a common pattern prevails in the APIs structure, something that facilitates the development of external applications, which like NOMAD, desire horizontal integration with multiple Social Media simultaneously. This concentrates in the widely adopted RESTful architecture, the easy to manipulate and represent JSON data format and the OAuth 2.0 authorisation protocol, as captured in the overview of the examined APIs. There is also correlation between the concepts behind each API methods, at least regarding the main methods for obtaining data. For instance, methods that our project needs to get data for users, who argue in a specific policy discussion, exist in all APIs. However, there is inconsistency among the data returned by each method invocation. Thus, in the previous example only Facebook, YouTube and Google+ stores user's gender and age, that NOMAD will visualise to the policy maker. Each Social Media platform produces diverse information due to the different resources it maintains and attributes that stores for these resources. This diversity is caused by the various special characteristics that Social Media inherit from their type, indicated by the classification. To address this inconsistency, we determined the Semantic Interoperability between the fields that NOMAD will retrieve, in Chapter 4. This delivers a useful guidance for the technical partners, since it both provides a listing of all retrievable data and a mapping between them in order to store and handle them in a uniform way. Due to the reasons explained before, the matching is not complete but is rather high in the comment, which is the basic resource type for extracting policy arguments within the project.

What also differentiate the retrieval of data in each one platform are the required permissions. Although, a common protocol regarding authentication and authorisation is followed by all applications, different types of data request different permissions, as elaborated in the pertinent subsections of Chapter3. With little effort, access to public data is feasible, while in case of more private information is requested by NOMAD, the application should be registered and authenticated. Public data includes basic profile information, such as name, last name, gender of the citizen that posted comments whereas educational level or occupation are usually private. This applies for all aforementioned APIs, apart from Google+ and Blogger, where no data is available unless you have an API key for a registered external application. So authorisation of NOMAD is a decision to be taken for the implementation of the Data Acquisition Module according to the data needed for the categorisation and visualisation.

In parallel, a State of the Art analysis on relevant applications, in Chapter 5 indicates that there are plenty of tools that capitalise on Social Media capabilities and their underlying content, but very few of them are oriented towards political scope. However the practices they adopt for business decision making in regard to content search, data acquisition, and visualisation can be applied in the policy domain. Thus, the features they provide were compared in relation with the corresponding NOMAD components (Data acquisition module, Opinion Mining and Sentiment Analysis module, Argument

Summarization Module Information visualization module). Since data sources in total overlap NOMAD sources, the tested combined approach of crawlers and API calls, is considered suitable for our project. Opinion Mining modules adopt various approaches, providing different accuracy results, so NOMAD's respective language specific module should be well trained and tested with relevant datasets. What is also evident is that existing visualisation techniques are insufficient and should be adapted to the policy modelling applications of our project. In general, NOMAD aims to offer a comparative advantage through the expression of policies in policy models. The matching between policy models and Social Media content provides more complete and aggregated set of results and time-series visualisations on policy arguments, something which is absent in the existing tools.

To conclude, a very positive finding for our project is that all Social Media APIs adopt common standards and provide methods for searching on a set of keywords and gathering related comments or posts and as well methods for extracting information about their authors. Thus, the analysis conducted in Chapter 3 produces general knowledge that can be exploited for integrating additional platforms beyond the selection made in Chapter 2 and beyond the boundaries of this project. In addition, it should be pointed out that such Web 2.0 platforms update their APIs constantly and should be observed at continuous level until the completion of the project in order to offer more functionality to policy makers. Yet, a lot of progress on the evolvement on Social Media APIs has taken place and capabilities are improved in comparison with the findings that another PADGETS project relevant analysis has revealed two years ago [7].

7. REFERENCES

[1] Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY. 1990.

[2] All about Social Media Monitoring Tools, Oneforthy.2011

[3] Laine, M.O.J. and C. Frühwirth (2010). Monitoring Social Media: Tools, Characteristics and Implications Software Business, 51, p. 193-198.

[4] I. Stavrakantonakis, A.-E. Gagiu, H. Kasper, I. Toma, and A. Thalhammer (2012). An approach for evaluation of social media monitoring tools. In Proceedings of the Common Value Management Workshop CVM, co-located with the 9th Extended Semantic Web Conference ESWC2012, Heraklion, Crete, May 28, 2012, pp. 1-17

[5] Hofer-Shall, Z., Murphy, E., Grant, M., Vittal, S. (2010). The Forrester Wave: Listening Platforms

[6] Dodds, P. and C. Danforth (2010). Measuring the Happiness of Large-Scale Written Expression: Songs, Blogs, and Presidents. Journal of Happiness Studies, 11(4), p. 441-456.

[7] PADGETS Deliverable 1.2 (2010), "'Standards, Interfaces and APIs for interplatform communication in Web2.0 Social Media". www.padgets.eu