Policy Cloud

Cloud for Data-Driven Policy Management

CLOUD FOR DATA-DRIVEN POLICY MANAGEMENT

Project Number: 870675 Start Date of Project: 01/01/2020 Duration: 36 months

D5.2 CROSS-SECTOR POLICY LIFECYCLE MANAGEMENT: DESIGN AND OPEN SPECIFICATION 1

Dissemination Level	PU
Due Date of Deliverable	31/08/2020, Month 8
Actual Submission Date	03/09/2020
Work Package	WP5 Cross-sector Policy Lifecycle Management
Task	T5.2, T5.3 & T5.5
Type	Report
Approval Status	
Version	V1.1
Number of Pages	p.1 – p.32

Abstract: This deliverable will include the specification of the tools used to model and design the policies as well as the design of the policy development toolkit, and the mechanisms required for the compilation of policies collections to perform cross-sector analysis and policy making. It will also architect the tools to identify groups for data-driven policies experimentation and adaptation.

PolicyCloud has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 870675.

Versioning and Contribution History

Version	Date	Reason	Author
0.1	08/06/2020	ToC and assignments	Armend Duzha
0.2	25/06/2020	Contribution in section 4	Kostas Moutselos, Ilias Maglogiannis
0.3	05/07/2020	Contribution in section 4.1.2	Ana Luiza Pontual
0.4	07/07/2020	Contribution in section 2	Armend Duzha
0.5	15/07/2020	Contribution in section 2	Javier Sancho
0.6	27/07/2020	Contribution in section 4	Luis Miguel Garcia Jesús Manuel Gallego
0.7	20/08/2020	Contribution in section 2	Iskra Yovkova
0.8	25/08&2020	Contribution in Section 3	Kostas Nasias
0.9	26/08/2020	Consolidated draft ready for internal review	Armend Duzha
0.10	28/08/2020	Internal Review	Konstantino Oikonomou, Giannis Ledakis (UBI) / Enol Fernández (EGI)
1.0	31/08/2020	Final version ready for submission	Armend Duzha
1.1	01/09/2020	Quality Check performed and changes addressed	Argyro Mavrogiorgou, Armend Duzha

Author List

Organisation	Name
MAG	Armend Duzha
ICCS	Kostas Moutselos, Ilias Maglogiannis
LXS	Luis Miguel Garcia Jesús Manuel Gallego
OKS	Kostas Nasias
ATOS	Ana Luiza Pontual
SARGA	Javier Sancho
ITA	Rafael del Hoyo
SOF	Iskra Yovkova

Abbreviations and Acronyms

Abbreviation/Acronym	Definition
API	Application Programming Interface
BDA	Big Data Analytics
DSS	Decision Support System
EC	European Commission
KPI	Key Performance Indicator
PDT	Policy Development Toolkit
PM	Policy Model
PP	Public Policy
SOA	Service Oriented Architecture
UI	User Interface
UX	User eXperience
UML	Unified Modelling Language

Contents

List of Tables

List of Figures

Executive Summary

This document is the first of the series of deliverables that will detail the specification of the tools used to model and design public policies as well as the design of the policy development toolkit (including the visualization module), and the mechanisms required for the compilation of policies collections to perform cross-sector analysis and policy making. The purpose of these series is to track the specifications throughout the project and update them during the progress of the project.

This deliverable has been released on M08 of the project, and its main aim is to define the concept of Public Policy (PP) and to propose a first approach to the modelling and evaluation of PPs that will be used in the Policy Development Toolkit (PDT) to support policy makers in the creation, modelling, and evaluation of PPs. The PDT is a web application that consists of the frontend of the PolicyCLOUD platform part, which allows policy makers to create and evaluate Policy Models (PMs), and the backend that hides the complexity of storing the data models into a persistent store and implements the services that the front end makes use in order to display the content to the user and provide the necessary user experience (UX). PDT intentionally will cover the complexity of the system dataflow to provide to the user a Decision Support System (DSS) towards evidence-based Public Policies (PPs).

1 Introduction

1.1 Purpose and Scope

This document is the first iteration of "Cross-Sector Policy Lifecycle Management: Design and Specifications", and is the first deliverable of WP5, covering tasks T5.2, T5.3, and T5.5. Its main purpose is to provide an initial approach to the modelling and evaluation of Public Policies (PPs) that will be used in the Policy Development Toolkit (PDT) to support policy makers in the creation, modelling, and evaluation of PPs.

This initial report will be further refined during the project duration. There will be two additional versions of this document (at month M20 and M32) that will refine the deliverable and are expected to include further advancements and contributions from other tasks.

1.2 Structure of the Document

The rest of the document is structured as follows:

- Section 2 introduces the definition of PPs, including state-of-the-art concepts on PP development that cover why, what, and how they are developed and who are actively or passively affected by the PPs.
- Section 3 presents the conceptual methodologies proposed for policy modelling, based mainly on the concepts of ontology and semantic reasoning technologies. It also proposes an initial approach on the policy modelling and their evaluation based on the concept of KPIs.
- Section 4 introduces the architecture of the PDT and describes the various User Interfaces (UIs) components, including the Data Visualization Framework.
- Finally, Section 5 provides the conclusion of the document.

2 Public Policies

A *public policy* (PP) is a plan, course of action, or set of regulations adopted by the policy makers to influence and determine decisions or procedures that affect a group of public and private actors in order to achieve a desired outcome. The Policy Maker gathers information through public consultation and scientific research to extract the necessary knowledge base and creates a policy or a set of policies, which serve to define and promote what is the preferred course of action or inaction, which in turn will establish targets and points of reference for the short and medium term. In PolicyCLOUD we define policy makers as government bureaucrats and technocrats from various sectors (e.g., healthcare, education, security, environment, etc.) and public sector staff who implement and evaluate programs.

The PP is carried out following a defined process, taking into account the context and characteristics of the geographic area (e.g. region) where it has to be implemented, with the purpose of driving the PP content. The PP will affect some actors that have to be considered during its design. When a PP is developed, a proper evaluation process shall be defined to assess whether the actions or inactions considered are fulfilling the defined goals. The evaluation process includes the definition and measurement of Key Performance Indicators (KPIs) to assess if the proposed goals are reached (and to what extent), and to evaluate if the correct parameters were selected to this end. KPIs are generally presented to a group of key actors that are interested in the results of the PPs, in order to decide whether to continue with the same policies, apply corrective measures, or to partially or completely redefine the content of the PPs. This schema is illustrated in Figure 1, as proposed by Gagnon and Labonté [1].

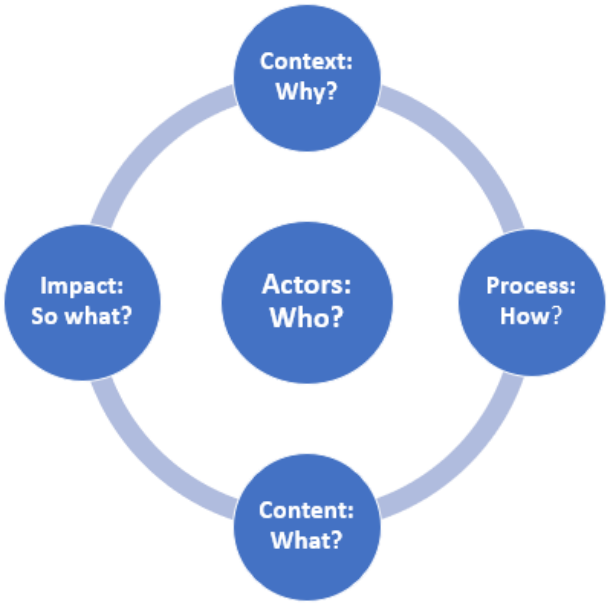

FIGURE 1 – POLICY ANALYSIS FRAMEWORK

2.1 Actors

Actors are defined as individuals, groups, or organizations that are relevant decision-makers and/or interested parties in monitoring the effect and evolution of the PP that have been created or modified.

In PolicyCLOUD we group them in two types depending on their ability to take decisions when formulating the content of a PP:

- **formal actors** are defined as groups who belong to the public authorities at any level and are responsible for the definition of the PPs and, thus, have formal political power or are direct policy makers;
- **informal actors** are defined as entities or groups who do not have any political power and, thus, are outside of the decision-making process, but they can have a scientific or political influence on the final decisions, such as the civil society, NGOs, etc.

The PolicyCLOUD project will be mainly focused on the formal actors, since the goal of the project is to serve as a Decision Support System (DSS) for policy making. This does not mean, however, that policy making using PDT will be performed without taking into account the opinion of relevant stakeholders. Indeed, informal actors could be considered as key entities or groups for the purpose of evaluating the KPIs. Hence, the PDT will have to consider which actors should be informed on the results of the KPIs related to the PP developed.

2.2 Context

The context of a PP defines why the policy is needed and refers to those cultural, structural, situational and/or external factors that may have a relevant effect on the society.

- **Cultural factors** refer to the existence of formal/informal hierarchies, different languages or ethnic minorities that may affect the equity of a PP, gender role, and the importance of religious factors among others.
- **Structural factors** are stable elements of the society, such as the political system, the level of participation of the civil society in the discussion and decision process, or the economic system, etc.
- **Situational factors** are temporary conditions, such as epidemic, natural disaster, austerity measures in times of economic crisis, that may also have an influence on PPs and/or affect the monitoring in temporary situations.
- **Other factors** refer to conditions of inter-dependence between states, cooperation among organizations/states or international laws and norms that may affect policies.

The context is not directly integrated in the policy modelling task of PolicyCLOUD since it cannot be modelled directly. However, the PDT should encourage policy makers to consider all the relevant contextual factors before developing any PP.

2.3 Content

The content refers to what the policy is mainly about, and which questions are taken into account in the design of the PP. The content of a PP defines specific issues, problems and/or challenges and is linked directly to the domain it applies (e.g., security – uc#1, agri-food – uc#2, mobility and transport – uc#3, and employment – uc#4). Some examples for each domain are provided below:

Security:

- How to improve access to public spaces
- How to increase efficient use of resources (e.g. number of police officers in the territory)
- How to promote a better inclusion of different groups in the society (e.g. dedicated programmes and activities targeting specific vulnerable groups, such as immigrants or minorities.

Agri-food:

- What trends in the wine sector do exist

- How to increase sales in an increasingly competitive market.
- How to increment the market of D.O Aragon product show
- How to increase efficient use of resources of marketing

Mobility and transport:

- How to efficiently allocate resources
- How to proactively target areas for preventive measures
- How to improve transport and parking utilisation and customer experience
- Identify trends and problem concentration areas

The content of a PP cannot be separated from the dimension of the process (how the PP is carried out), the actors (who define the PP and need to be informed on the evolution), the context (where these policies will be implemented and what specificities they should have), the stakeholders (who are affected by the PP), and the evaluation (what impact is the PP having, as measured by KPIs that should have been defined previously).

The purpose of the PolicyCLOUD project is to support policy makers in developing the content of the PHP as an evidence-based outcome of the PDT.

2.4 Process

The process defines how the PP was brought forward and implemented. The process of policy making can be seen as a methodology or approach that is defined by seven phases, which extends the policy analysis cycles provided by Patton and Sawicki 1993 [2] (see Figure 2):

In the first step the policy maker **defines and details the given problem** by characterizing the social context in which the problem is embedded and identifying the independent variables that affect policy outcomes. It is the role of the policy maker to understand the positions and influence of various stakeholders and choose the definition that the problem owner/decision maker has control on. Clarification of the problem takes place with consultation, brainstorming, narratives, and scientific research.

In the second phase, the policy maker **identifies the evaluation criteria** that show when the problem is solved or a goal is accomplished. She/he will select those criteria that are central to the problem and most relevant to the decision makers in the implementation process [2].

Once the policy maker knows the values, objectives, and goals of the stakeholders and the evaluation criteria for validating the policies, she/he can **generate a list of possible policies.** This list is usually long since there are many variations and combinations. Benchmarking and past experience are common approaches for identifying policy alternatives [3] [2].

Among policy alternatives, the most appropriate options are selected using the already defined evaluation criteria, leading thus to the **formulation of the policy**. The different alternatives are assessed based on the potential effects and their chain of causation. Since not every policy can be tested with the same method, various methods (e.g., cost–benefit analysis, programming, institutional analysis, and quantitative analysis) to evaluate different policies can be used.

The next phase is the **policy implementation**. This is the most important phase, since is responsible for setting up the public authority's planned actions in the way it was intended, in order to achieve the expected impact and results. In most instances, the policy maker develops implementation guidelines and procedures rather than being directly involved in the implementation of the selected policy. It is important for the policy maker to know whether

a failed policy could not be implemented as designed or the policy did not produce the desired results because the underlying theory was incorrect [2].

The final phase is the **monitoring and evaluation** of the implemented policy. It requires the involvement of both formal and informal actors. In particular, this phase is about monitoring the use of inputs and the achievement of outputs, and evaluating the direct effects and medium-/long-term impacts of the policy provide evidence on whether the policies are achieving their objectives.

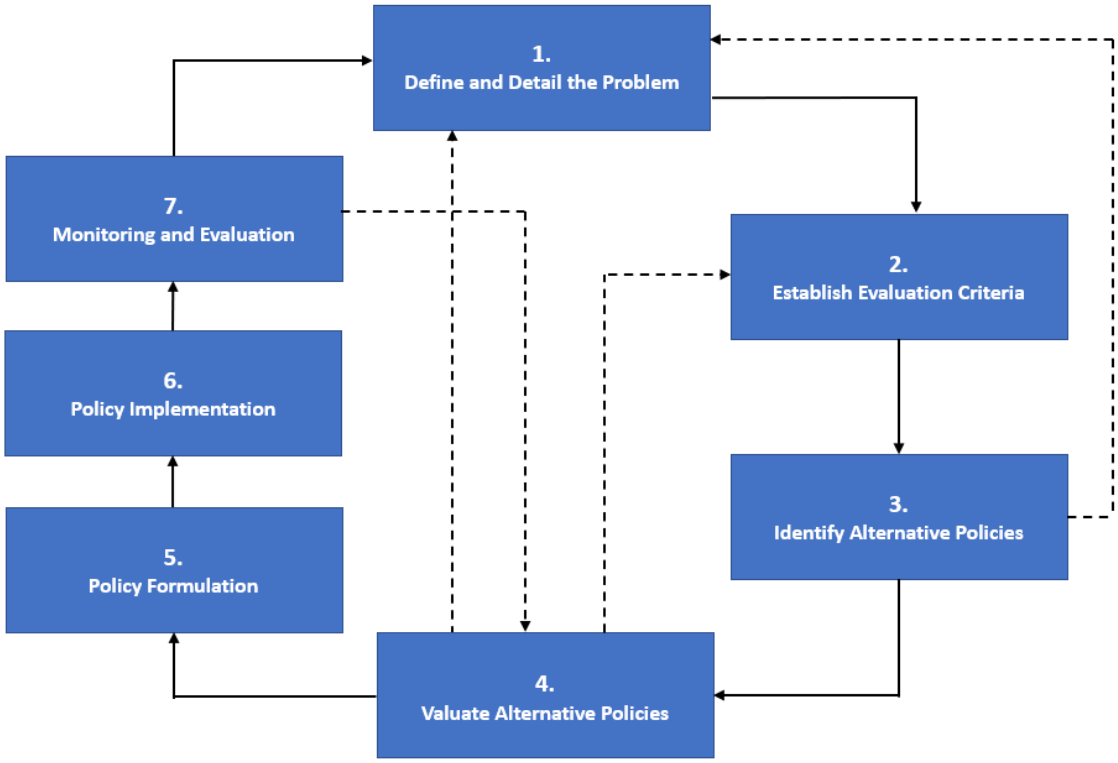

FIGURE 2 – POLICY ANALYSIS PROCESS

Although in common daily practice these stages lack clear separation, the PDT proposed by the PolicyCLOUD project is mainly devoted to directly helping the policy maker in the policy creation and decision-making stages, and, indirectly, in the policy implementation and policy evaluation stages.

2.5 Impact

Proper information management and monitoring is of utmost importance to enhance decision making and to monitor and track evolution of the PP that have been implemented. A useful tool in this regard is the definition and measurement of the KPIs that are already described in the paragraphs above. KPIs are metrics that are determined through scientific evidence or through the consensus of experts (when evidence is unavailable) to provide feedback to key actors about the results being achieved and to use this information to improve PPs.

Information is a key resource for developing specific KPIs that enable the measurement of each goal. KPIs are also a tool that needs to be reported for accountability reasons to those agents who may be responsible or who have to take informed decisions on existing PPs and their possible update. In addition, in order to develop good indicators, a set of common criteria could be used to aid the experts and final users to reach a consensus on which indicators

should be taken into account. To this end, the KPIs should be valid, specific, measurable, reliable, evidence-based, achievable, feasible, relevant, time-bounded, i.e. reported at regular intervals, and safe.

The benefits of using KPIs are several and they include mainly the ability to assess the outcomes of PPs, the ability to compare equal parameters among different organizations following a benchmarking process of sharing improvements, the promotion of accountability to relevant actors, the promotion of transparency by publicly reporting the results, and the identification of areas for further research.

An example of a KPI from an existing PP from the use case 3 is provided below.

Business KPIs	Success indicators
Increased brand awareness and reputation in social media	Number of posts, comments, retweets
Identified the most interesting concepts and trends in wine markets	Number of new terms identified Number of influencers identified
Brand analysis: quality vs price vs taste	
Prediction analysis: quality of the next crop	

TABLE 1 – DEFINITION OF THE KPIS FOR THE USE CASE OF ARAGON

3 Policy Modelling and Configuration

3.1 Modelling Methodologies

3.1.1 Ontologies

Policy modelling and decision making (DM) knowledge is often represented as a set of basic definitions such as alternatives, criteria, decision matrix, and decision itself. However, this domain is much richer, and many other related notions are useful to make right decisions: preferences, weights, thresholds, and so on. This knowledge must be formalized within a model. In addition, the practical needs for policy making as well as the number of researches dealing with decision-making increasingly grow as, for instance, in Information System (IS) Engineering.

This kind of modelling is justified by the necessity to show different semantic links existing between policy making concepts. We have elaborated the ontology in order to represent concepts of the policy making domain, as well as their properties and relations.

The starting point for analysing the ontology (see Figure 3) is the situation. The situation is an abstract concept that puts together the main elements and describes a set of specific conditions dealing with a given object. The object is an artifact being the subject of decision-making.

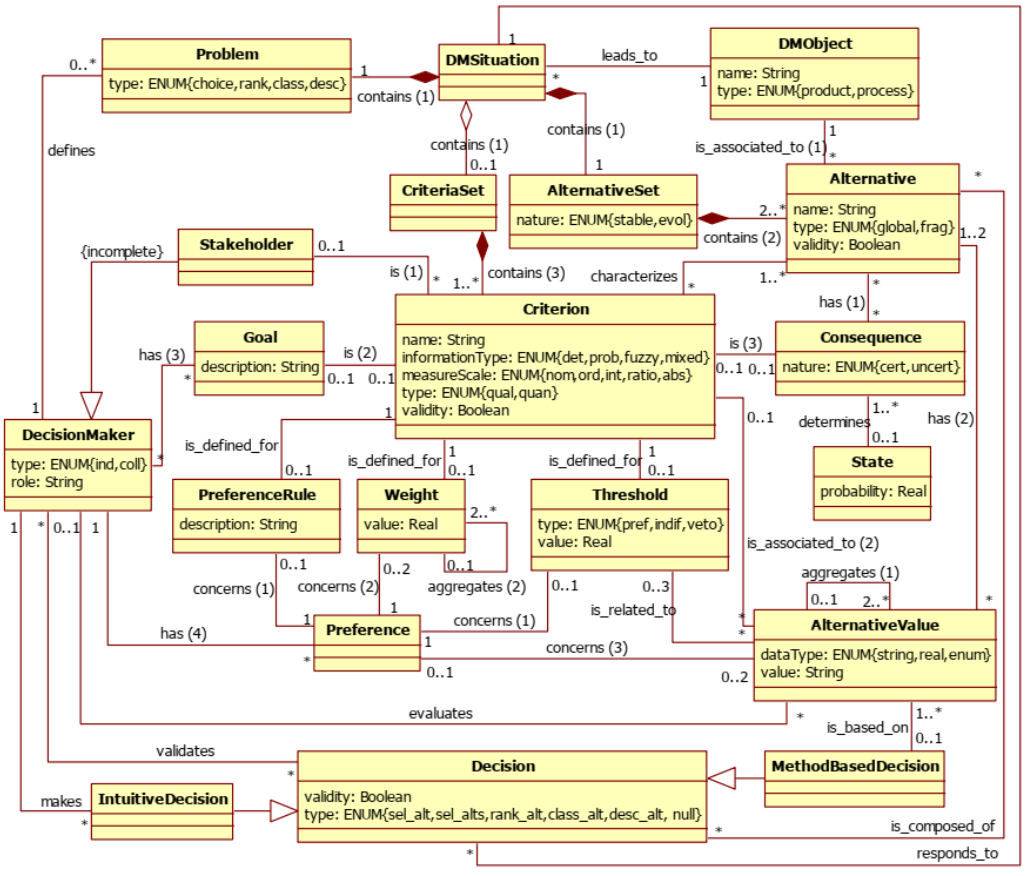

FIGURE 3 – DECISION-MAKING ONTOLOGY DIAGRAM

For example, a policy making about budgeting in a public organization like a government will have the key terms presented in the next table. The table does not include partial or total overlapping of concepts, synonyms, properties, relations and attributes.

Activity	Expense	Subpartial Item
Budget	Expenses Classifier	Subprogram
Budget Analytic	Expense Object	Program Executer Unit (UEP)
Budget Approved	Finality Function	Programmatic Category
Budget Project Draft	Financial Administration	Project
Budget Synthetic	Financing Source	Public Funds Administrative Service (SAFOP)
Budget States	Geographic Locate	Rector Organism
Budgetary Classifier	Institutional	Resource
Budgetary Fiscal Year	Institution	Resources Estimation
Budgetary Policy	Jurisdiction	Year Financial Administrative Service (SAF)
Budgetary Top	Program	
Executor Organism		

TABLE 2 – KEY TERMS

To properly understand the conceptual aspects in the context. Figure 4 presents a Unified Modelling Language (UML) diagram, with the main relations among defined concepts.

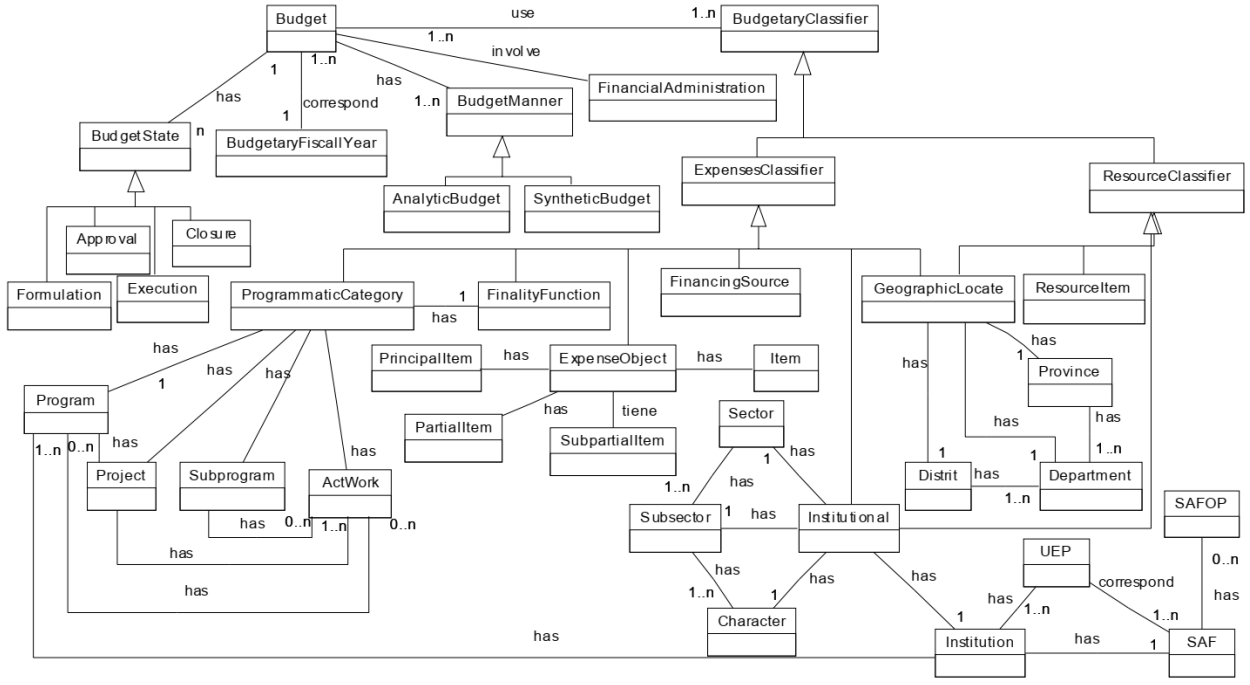

FIGURE 4 – EXAMPLE OF A UML DIAGRAM

3.1.2 Semantic reasoning and querying

Semantic queries allow for queries and analytics of associative and contextual nature. Semantic queries enable the retrieval of both explicitly and implicitly derived information based on syntactic, semantic and structural information contained in policy data store. They are designed to deliver precise results (possibly the distinctive

selection of one single piece of information) or to answer fuzzier and wide-open questions through pattern matching and digital reasoning.

Semantic queries work on named graphs, linked data or triples. This enables the query to process the actual relationships between information and infer the answers from the network of data. From a technical point of view, semantic queries are precise relational-type operations much like a database query. They work on structured data and therefore have the possibility to utilize comprehensive features like operators (e.g. >, < and =), namespaces, pattern matching, sub-classing, transitive relations, semantic rules and contextual full text search.

By implementing the JSON format to add semantic annotations output, the system will be able to automatically integrate data from different sources by replacing the context-dependent keys in the JSON output with URIs pointing to semantic vocabularies, that will be used to represent and link the data.

3.2 Policy Modelling

3.2.1 Data model

A *policy model* is an instantiation of the data model. At this phase of the project, we rely on existing approaches that have been already proposed for the modelling of policies in various domains. Our current definition of the policy model can be depicted in the following class diagram.

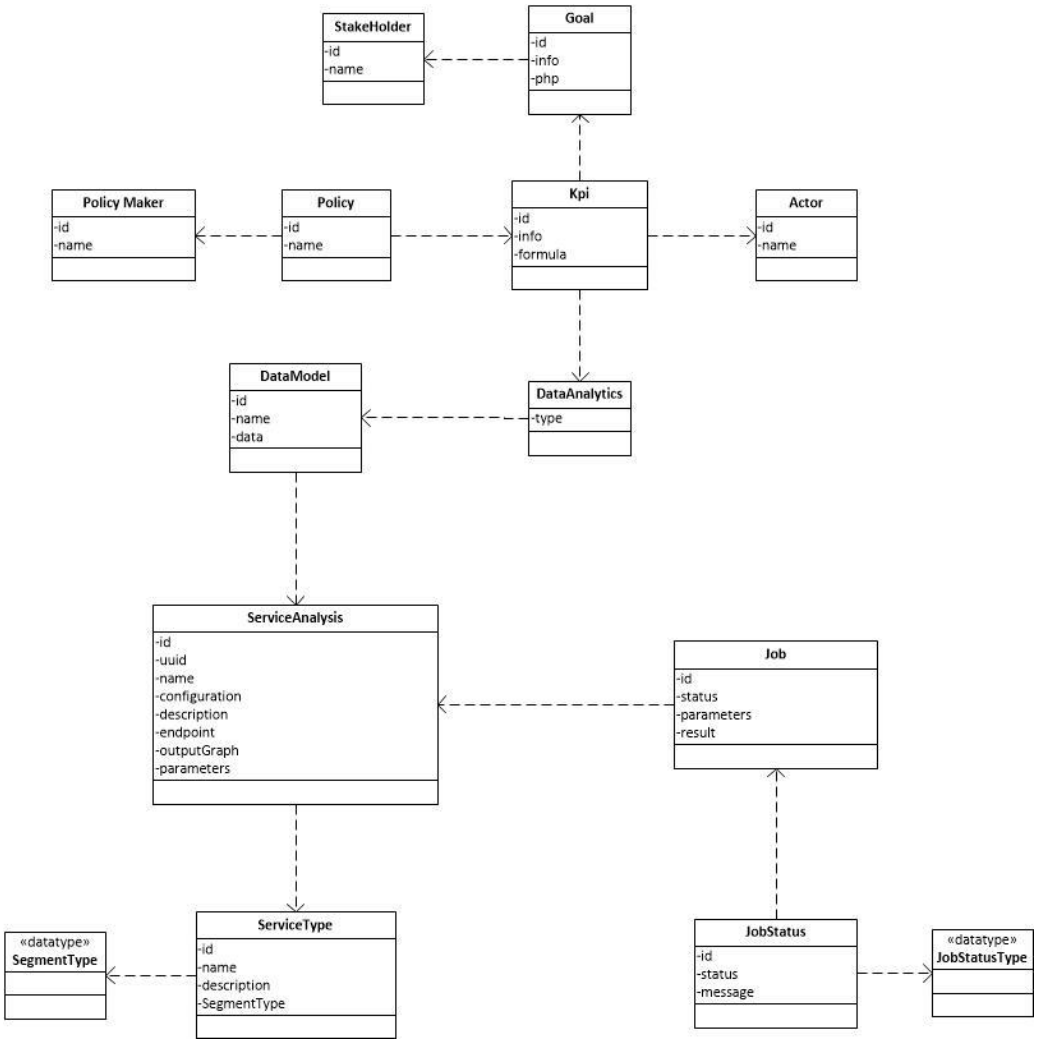

FIGURE 5 – POLICY MODEL CLASS DIAGRAM

From the class diagram, it can be concluded that there are several *entities* that are involved in a policy model, and each one of those can be reused by other models. The central entity is the *policy* which holds all information regarding the specific model. This might be assigned to a specific policy maker, thus making this instance private to her, or can be public, thus being a model for creating new policies. This also allows the policy maker to be able to search and explore her own policies. Each policy on the other hand, is associated with a list of available *KPIs,* which defines the key performance indicators of the policy. KPIs are related with specific goals per stakeholder. It is important to highlight that our design enables the re-usability of the KPIs, in a way that a well-defined KPI can be also used by other policies. As a result, there is a many-to-many relation between the *policies* and the *KPIs* that allows for this to happen.

Moreover, each KPI is being validated by a list of analytical tools that performs the processing and produces the results, for the KPI to be validated. The analytical tools might be of different type (i.e. sentiment analysis, risk assessment, etc.) and can be related with one or many different data sources. Therefore, an analytical tool is not only implemented and locked-in for a specific data source, but can be a general-purpose tool that can be applied to several data models.

Finally, when an analytical tool is being invoked, this submits a *job* for execution. Due to the fact that the processing of a dataset can be a long-running process, the invocation of a tool should be considered to take place in an

asynchronous manner. Therefore, a new *job* is created, and when the analysis is concluded, the result of the latter can be stored in that job.

To summarize, we can see that this data model for policies allows for great flexibility, as it avoids duplicating entities and instead, it allows for the re-usability and the addition of new elements that can be offered in the ecosystem. A policy maker can have a list of private *policies,* or explore the *policy models,* validate the existing KPIs that might be applied in different domains, and investigate why some of them can be beneficial across domain sectors, and why others are more related only to specific ones. The proposed solution also allows for the provision of additional analytical tools that can be developed in the future and can be easily plugged in into the platform and made available to any existing policy model. It gives an absolute freedom for the policy maker to validate different aspects and KPIs using a variety of different tools. We need to highlight at this point that the process of defining the data model of the *policies* is done in an agile approach, and we expect to further extend the proposed schema in the second version of this deliverable, once we get feedback by the policy makers and domain experts of our four use cases.

3.2.2 Policy modelling editor

There are some impediments to effective use of policies data stores by the end users that we need to address:

- Given the huge number of ontologies, it is hard for users to find and effectively apply them.
- End users should not be expected to have programming knowledge: they need a simpler way to get data out of the Policies data stores.

Hence, we need to develop and deploy a middleware which can be used as the adapter pattern, to retrieve data from the Policies data stores. The middleware will be a modelling editor based on .NET Core which will lie between the data stores (after the data acquisition) and the data visualization.

Essentially functioning as hidden translation layer, middleware enables communication and data management by users and allow them to perform requests by completing a simplified wizard (assistant).

The wizard will have 3 main steps as presented in the infographic bellow:

FIGURE 6 – POLICY MODELING EDITOR STEPS

By completing the above steps, the user will be able to create a policy easy and fast without any programming knowledge and provide the output to the data visualization component.

As for the existing policies the users will name a description with a set of rules (criteria) which will apply the values of a specific schema of data and Key Performance Indicators.

4 Policy Development Toolkit

The present section describes the functionalities of the Policy Development Toolkit (PDT). As a web application, the PDT consists of the frontend part that allows policy makers to create and evaluate Policy Models (PMs), and the backend that hides the complexity of storing the data models into a persistent store and implements the services that the front end makes use in order to display the content to the user and provide the necessary user experience (UX). PDT intentionally will cover the complexity of the system dataflow to provide to the user a Decision Support System (DSS) towards evidence-based Public Policies (PPs).

4.1 Architecture

The general interconnection of PDT with the other PolicyCLOUD components is illustrated in Figure 7. PDT enables the processing of data through the exploitation of collective knowledge that emerges from multiple information sources. The platform explores mechanisms that can be clustered across two main areas: analytics on heterogenous data at population and individual level and predefined policy models exploiting community knowledge across different ecosystems. PDT may be considered as the point of integration and interaction of the platform with the policy makers. Through the PDT, the policy makers will be able to question the platform data and exploit the analytics tools to perform policy creation and evaluation.

4.1.1 PDT Design

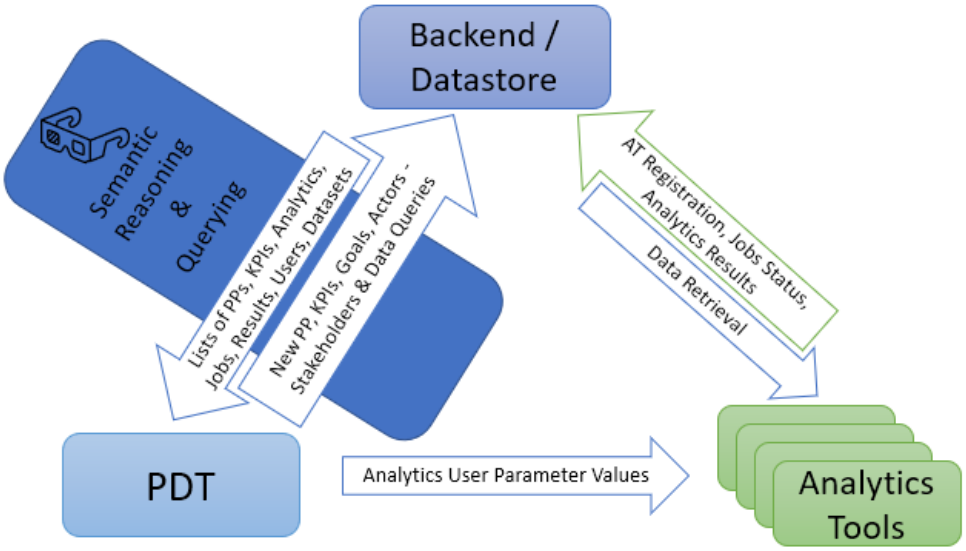

FIGURE 7 – PDT COMMUNICATION COMPONENTS

Figure 7 shows the two components with which PDT will communicate: Backend/Datastore and Analytics Tools. Both components will expose API Interfaces so that UI receives the data from Datastore and sends to Analytics the parameters that the policymaker will select to send for processing.

In the proposed architecture [4] each component is decoupled from the others. The modular structure allows versatility and extensibility, by means of analytics tools providers, analytics frameworks, cloud providers and deployment patterns, keeping a stable trustworthiness level though. For example, the Analytics Tools modularity permits the easy addition of various categories tools (such as simulation, social network analysis, semantic and linked data etc.) which are earmarked for policy-making purposes. This, in consequence, supports the policy maker

in the creation of policy models that are based on the composition of related KPIs. The -also- modular UI intentionally hides the big complexity for the users, as each component is decoupled and focused on their own properties and functions (in the object-oriented perspective). So, a Policy Model is composed and supported by related KPIs, which in turn are composed of related Analytics Tools which provide their own visualization graphs. The Service Oriented Architecture (SOA) pattern is followed by requiring the components to adhere to a common communication protocol (such as the Analytics Tools registration payload shown in Figure 7), and by exposing consistent RESTful APIs. This allows different Analytics Tools providers and big data technologies to co-exist and collaborate to create a policies related ecosystem. The modular structure also supports scalability. Each module, depending on the load it receives on a case by case basis, can resort to load balancing solutions.

Regarding the modular design of the Analytics Tools, means that each Analytics Tools developer may employ different Big Data Analytics (BDAs) frameworks using various deployment configuration mechanisms. This decoupled and distributed architecture provides fertile ground for the growth of an Analytics Tools Store, where Data Experts develop specialized Analytics Tools who register them in the PolicyCLOUD platform, and can be invoked by the PDT using a common data model that the tools must follow via a common protocol. In consequence, Analytics Tools can operate and provide their services under different financial procedures, including commercialized options. This way, new policy models can be formulated, based on KPIs that utilize the new Analytics Tools.

In detail, the arrows in Figure 7 shows the content of the communication on each side. The policy maker, via PDT, will be able to retrieve a list of existing policies, view the structure and content of a policy, edit it (if she/he has the appropriate permissions), and save it. The User will also have access to the registered analytics tools, so that when creating new policies, she/he can select the ones related to the content of the policies and link them to these policies (via the corresponding KPIs).

The display of policies via the PDT is based on the aforementioned policy model that is being fed by the data that are stored in the Datastore. Similarly, PDT, when the user modifies an existing policy by changing or creating new elements or creating a new policy, when she/he chooses to save it, the PDT sends the policy to the Datastore to be persistent, via a serialized message (in JSON format) invoking the REST API provided by the backend.

The blue level in Figure 7 represents the process of semantic reasoning and querying. Based on the process set out in Section 3.1, the semantic processing of emerging policies for lifecycle policy modeling is intervened. The metadata provided by this level, together with reasoning, enables the validation of the policy structure in terms of their proper construction. They also help policy makers to choose KPIs, avoid dysfunctional policies, and provide cross-sectional policy optimization information.

The following is a list of basic functions that PDT will provide to the policy makers:

- User Login
- User Profile
- Policy Model Viewer
- Policy Model Editor
- Policy Evaluations / Analytics
- Transactions History (This component offers a consistent way of displaying all the Analytics Results that the user has submitted to HATs along with the selected parameter values)
- Getting Help

Follows an initial Storyboard suggestion:

4.1.1.1 STORYBOARD

INTRODUCTION

After login, the initial page will initiate the policy 'creation/building'.

We call this component/module 'policy creator or builder' by convention, although it helps policy makers to create a real-world policy for example:

"Do something (like lower the taxes for sugar free products) for population on high risk regarding obesity",

for our system, a policy-query (and making/create it via controls/menus) is:

"Identify high risk population regarding obesity" (simplified what policy maker 'creates' via wizard)

(e.g. results may show age group 40-50 male, so targeting products that are consumed from that group will help policy maker create the 'real world' policy)

A mockup is shown in Figure 8.

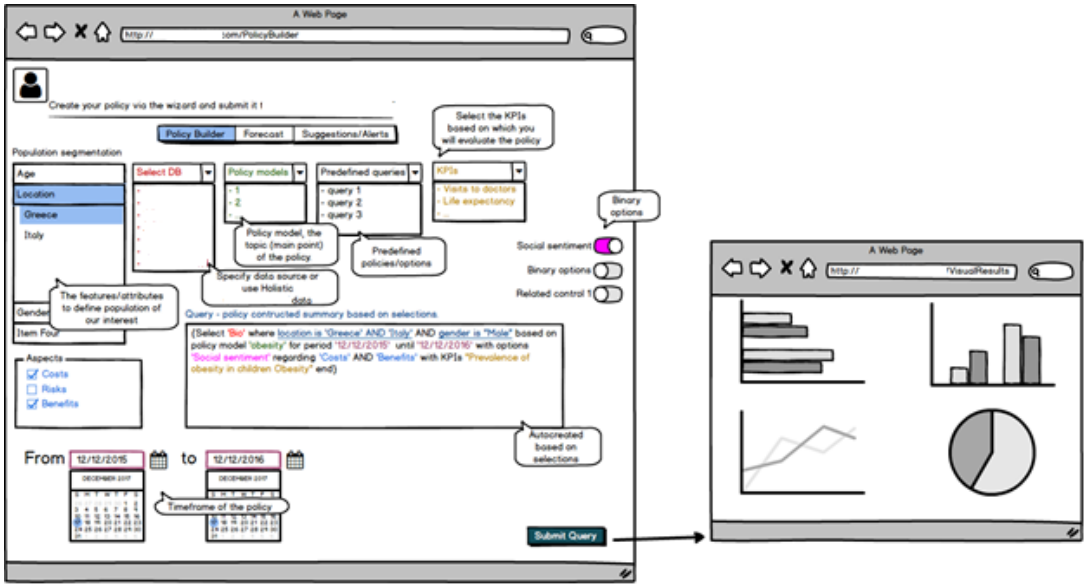

FIGURE 8 – MOCKUP: INITIAL PAGE AFTER LOGIN

POLICY WIZARD

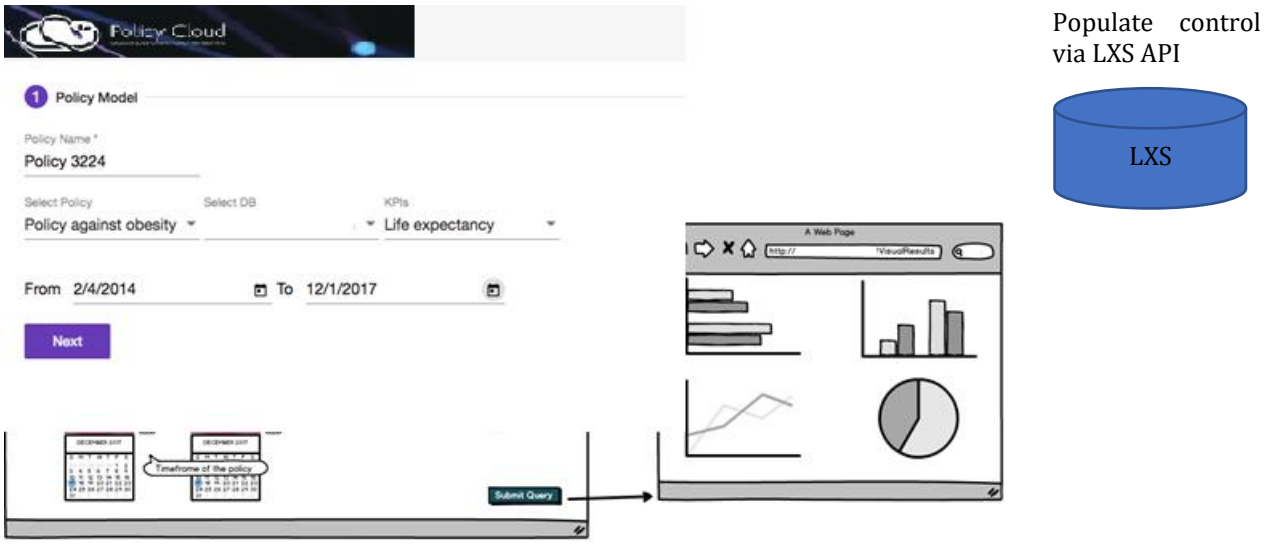

FIGURE 9 – POLICY BUILDER: WIZARD FOR THE END USERS (IMPLEMENTATION MOCKUP)

The user loads a Policy Model (PM) from the model's repository. Each model is associated with some goals/objectives, could include some predefined queries for population segmentation and a list of KPIs.

At page initialization, the KPIs will be fetched dynamically from the datastore via the backend exposed API (instead of being hard coded options in the code, this way KPIs are represented by different entities that can be reused and updated by just adding them in the database repo).

Some options will be static, for example the age, gender, and timeline.

For the rest of controls, we need APIs with the available options for the policy makers (see below).

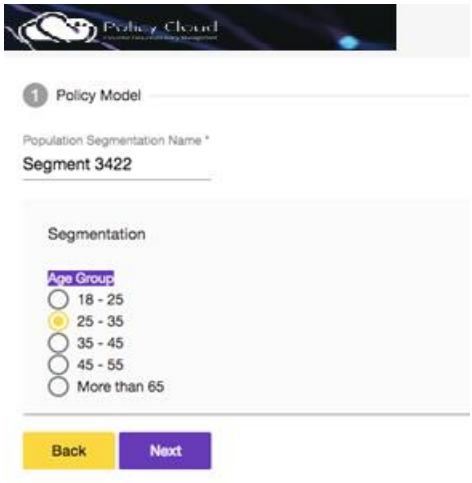

FIGURE 10 – POPULATION SEGMENTATION (IMPLEMENTATION MOCKUP)

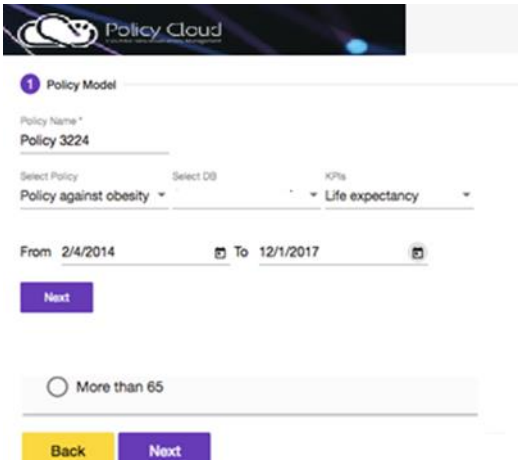

FIGURE 11 – TIMEFRAME SECLECTION OPTION

At initialization of the page, the controls that are dynamically populated (as mentioned above) should be populated with 'options' from component/modules like e.g. analytics.

The front-end gets the corresponding analytical tools that are related with a specific policy along with their meta-information, such as the parameters that are expecting, their types, allowed values etc. This information is retrieved by invoking web methods exposed by the backend as REST services. The response of such a request is a JSON object with this information and allows the front end to draw dynamically the content of the page accordingly. In order to do so, we have modeled the input parameters that each of the analytical tools will expect. This provides a generic and common way for all tools to describe these parameters so that the front-end of the PDT can invoke all registered tools in a common manner via well-established protocols, and become extensible to allow for future tools to be plugged-in and avoid being locked-in to specific implementations. The model definition of the parameters can be depicted in Figure 12:

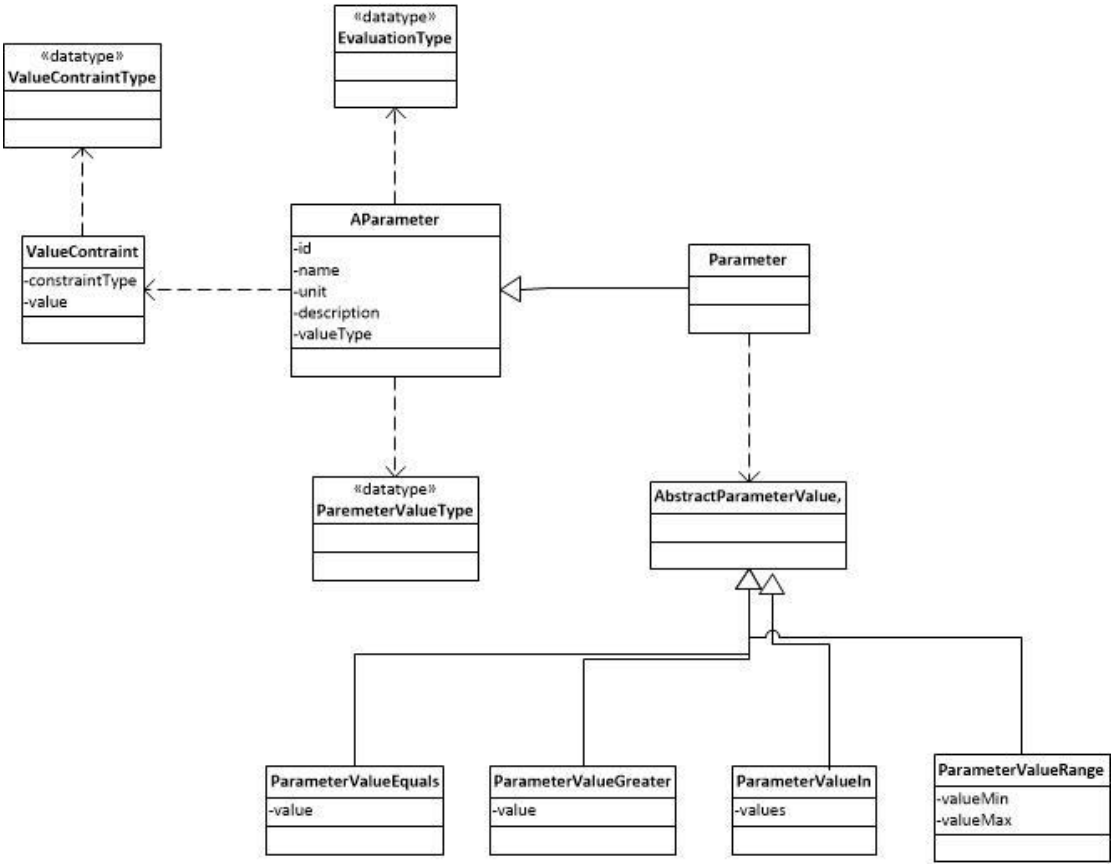

FIGURE 12 – CLASS DIAGRAMM OF THE PARAMETERS

An AParameter describes an available parameter required by the analytical tool. It is of a specific type, might contain a list of different constraints, and has a specific evaluation type. When this parameter has also a concrete value, assigned during the runtime, then this value can be of different types, according to the evaluation type.

The evaluation types can be one of the following: EQUAL, LESS, LESS_OR_EQUAL, GREATER, GREATER_OR_EQUAL, IS, NOT, RANGE, IN

The ValueConstraintType can be one of the following: DEFAULT, VALUESALLOWED, MIN, MAX, STEP.

The parameter value types can be one of the following: BOOLEAN, INTEGER, FLOAT, STRING.

For instance, in aforementioned example, the KPI named *life expectancy* is related with a parameter of type *date,* which accepts a range of values. Therefore, the front-end will retrieve this information via the corresponding REST service to dynamically draw two fields to fill the *from* and *to* values of the *range*, and will provide 2 *calendar* objects for the end-user to insert her input. To summarize, the data store holds all this information that describes the analytical tools and the front-end provides a wizard to facilitate the start of an analysis. In each of the steps of this wizard, the front-end asks the backend for the corresponding meta-information and dynamically is being adjusted to improve the overall UX of the end-user.

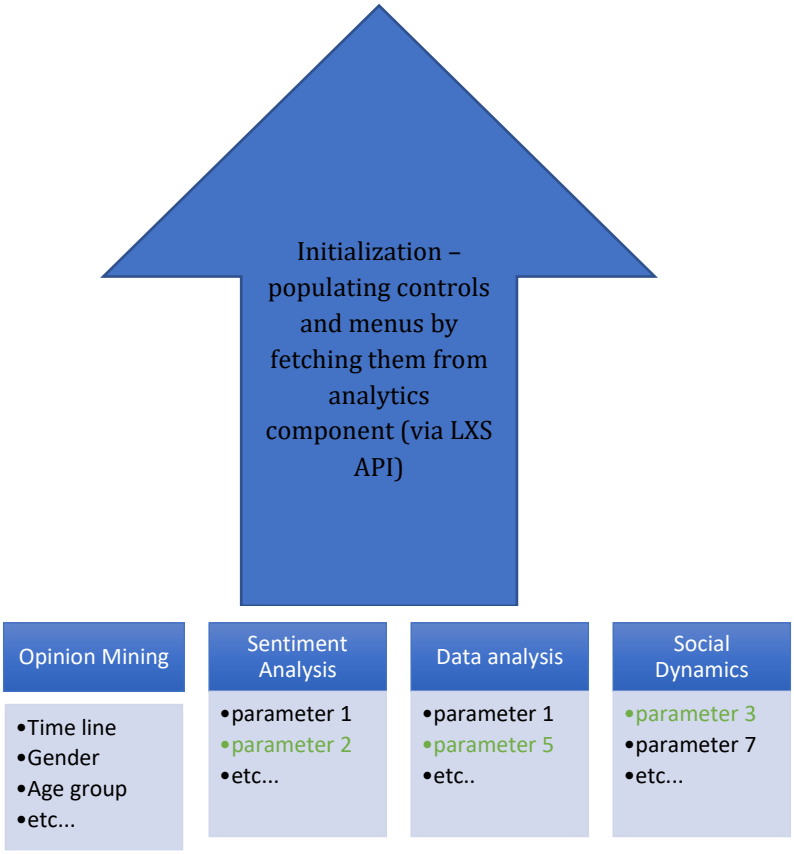

FIGURE 13 – INIZIALIZATION OF UI CONTROLS FROM THE DATASTORE POLICY MODEL

SUMMARIZE AND SUBMIT

After the initialization of the PDT according to the selected policy and the adjustment of the UI with respect to the meta-information provided by each analytical tool upon its registration, the policy maker is now able to experiment by providing her options to the toolkit and submit a request for execution of the analytical tool(s).

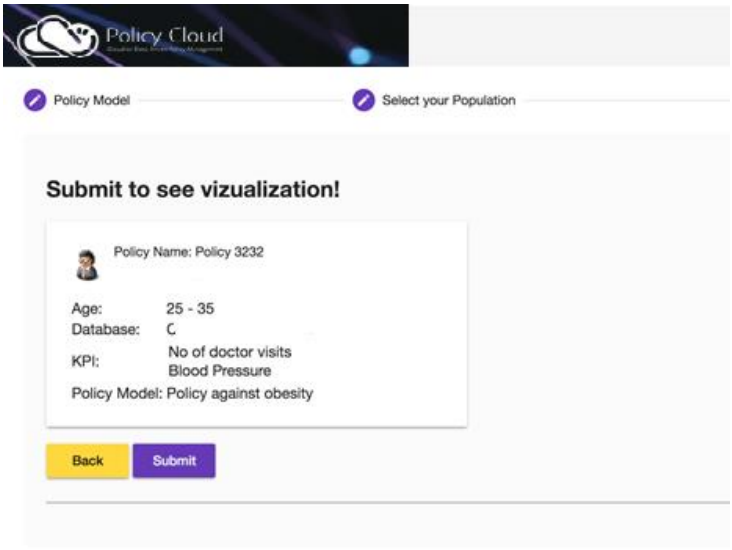

FIGURE 14 – SUMMARIZING BEFORE SUBMITTING

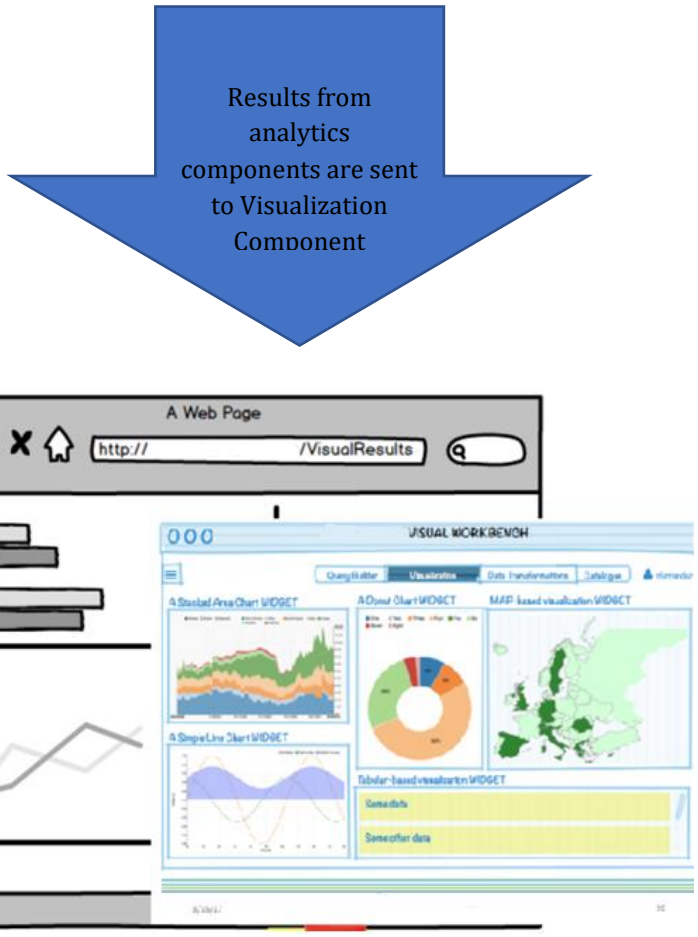

FIGURE 15 – RESULTS FROM ANALYTICS FETCHED FROM BACKEND AND VISUALISED IN PDT

To the bottom line the PDT does not make policies, but assists policy maker in policy development, offering a single and integrated entry point (one stop shop approach) to the PolicyCLOUD analytics and Visualization tools.

USER (POLICY MAKER) HISTORY OF POLICIES

The end user will be able to see, edit or delete any of her previously created policies.

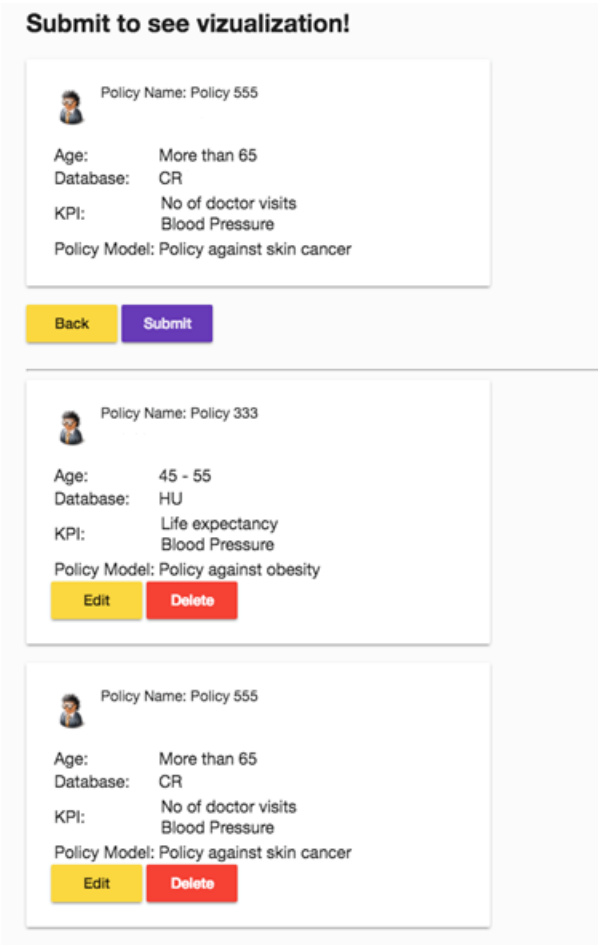

FIGURE 16 – HISTORY OF SUBMISSION TO ANALYTICS TOOLS WITH SUMMARISED OPTIONS/FILTERS

4.1.2 Data Visualisation Framework

The PolicyCLOUD Data Visualization Framework, part of the PDT, aims to help policy makers in the policy creating a process, as well as in its subsequent follow-up. This helps comes in the form of different charts, graphs, tables, and any other kind of visualization defined, enabling policy makers to check in a glance the results of selected data analytics queries outcomes.

On a first approach each pilot will define the best set of visualizations in order to fulfill its needs. These first defined visualizations can be changed after discussions and further analysis of available data.

One of the most attractive types of visualizations are “Maps”, that are a very illustrative and visual way to detect differences between regions and can be used into the Data Visualization Framework to present analytics results to the stakeholders. The type of map to be used is defined by the granularity of the data collected. Technologically a map that displays data at country, city or neighborhood level can be plotted, but this can only be done when the data source has enough information to do it.

FIGURE 17 – EXAMPLE OF A MAP VISUALIZATION

Among possible geographically representations, one that could be very useful to PolicyCLOUD Data Visualization Framework is the **Heat Map**, as they are a very intuitive, understandable, and user-friendly way to visualize data. "A data visualization technique that shows magnitude of a phenomenon as color in two dimensions", a heat map can help to quickly extract valuable information from a big amount of data.

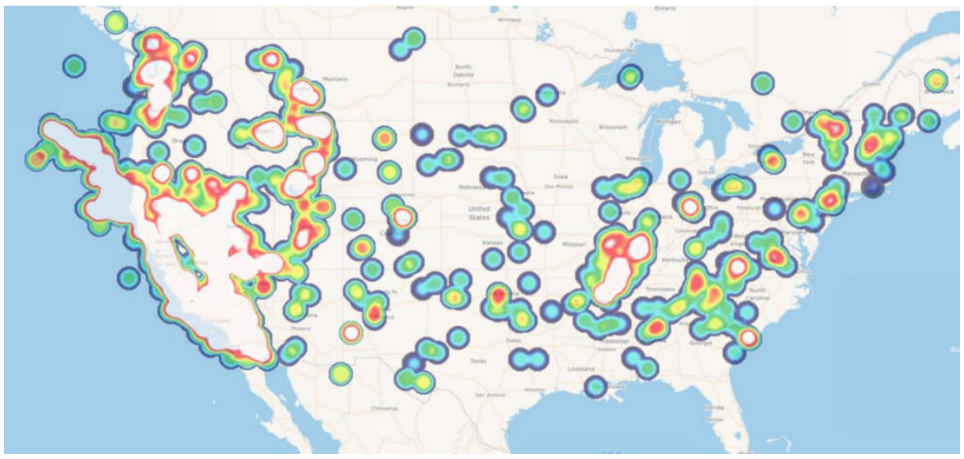

FIGURE 18 – EXAMPLE OF A HEAT MAP VISUALIZATION

But, although Heat Maps are undoubtedly a very good way to represent data, it is not always the best option. Looking for the best way to represent available data, two variables must be necessarily considered: the end user needs, and the data sources that will feed these visualizations. What is wanted by end users and what can be given by data sources should be in balance when selecting the visualizations.

In line with above mentioned, other types of visualizations, like line, pie, bar, gauges or radar charts, can also provide a lot of information and could be very helpful, when the data source used is social network based, for example.

In order to visualize data for a Sentiment Analysis, for example, it can be used charts like line, gauge, and bar charts, fed by any social network data source (Twitter; Facebook…). A line chart can be used to observe a specific behavior

over the time. In the example below it can represent, for example, the change of opinion or sentiment about a theme, between positive and negative, over the time:

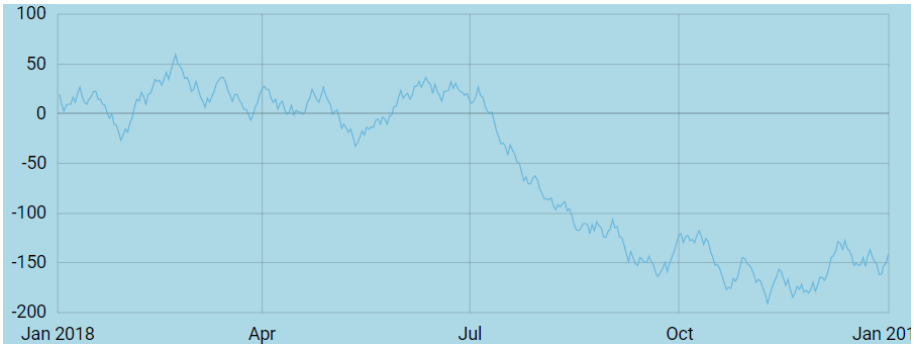

FIGURE 19 – EXAMPLE OF CHARTS USED FOR SOCIAL MEDIA ANALYSIS VISUALIZATION: LINE CHART

A gauge chart, due to its high customization, can be also very functional. In the example below, it is used to show quickly, the opinion media about a theme. It shows in a glance if the sentiment about a theme is mostly positive or negative, for example:

FIGURE 20 – EXAMPLE OF CHARTS USED FOR SOCIAL MEDIA ANALYSIS VISUALISATION: GAUGE CHART

A bar chart is used to express, given a set of values, the accumulate quantity for each value. In the example below, it is used to show the quantity of each sentiment found from Facebook, for example:

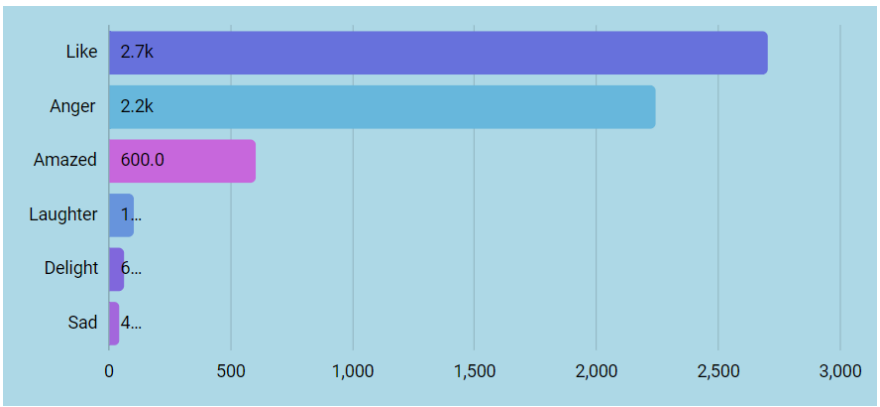

FIGURE 21 – EXAMPLE OF CHARTS USED FOR SOCIAL MEDIA ANALYSIS VISUALIZATIONS: BAR CHART

Another example of a chart that could be fed by any social network data source are tag clouds (also known as word clouds), used to check the frequency of texts, using different sizes (bigger with higher frequency) and colors:

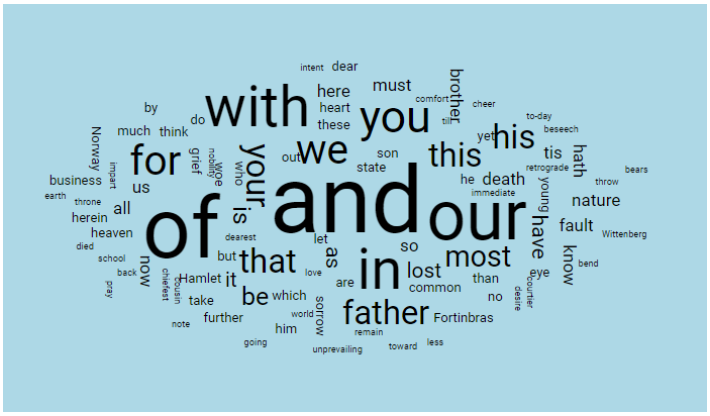

FIGURE 22 – EXAMPLE OF CHARTS USED FOR SOCIAL MEDIA ANALYSIS VISUALISATION: TAG CHART

Also, a radar chart could be useful, depending on the analyzed data. In the example below is shown an example with quality vs price vs integrity vs innovation vs speed vs consistency vs accuracy:

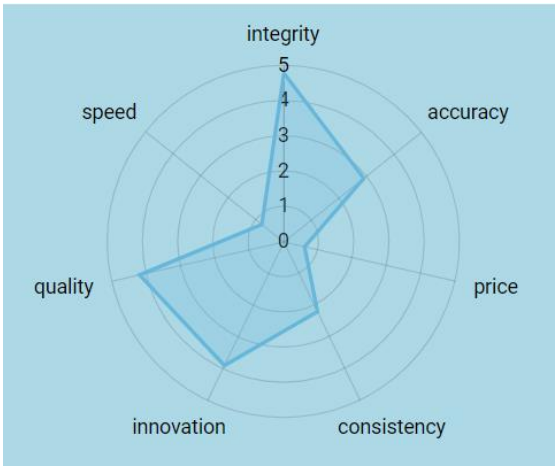

FIGURE 23 – EXAMPLE OF CHARTS USED FOR SOCIAL MEDIA ANALYSIS VISUALISATION: RADAR CHART

As can be seen, there are a lot of useful and functional charts that can be used in order to simplify and quickly show a big amount of information, that can come from any kind of data sources. The decision of which to use depends on the value they can provide to end users, taking into consideration their use cases, and the available data sources.

It is important to be highlighted that each of these charts will require the data of the result to be provided in a predefined format that the chart supports. Therefore, the analytical tools, after the processing of the data and the generation of the result, they need to provide the latter under the corresponding format in order for the charts to be compatible and in a position to draw the visualization graph. For that to happen, each tool announces upon registration the type of visualization that its results are compatible with, and store the results respectively. Then, the PDT will be capable to retrieve the results stored in the data repository, check the type of chart that these results are related to, and initialize the graph with the data in the agreed format. At this point of the project, the initial list of charts that need to be supported at this phase, along with the capabilities of each of the available analytical tools, has been identified. This process has not been finished yet, and therefore, the definition of the data format that each chart needs to be compatible with, is still in progress and will be reported in the second version of this deliverable.

4.2 Baseline technologies and tools

PDT design is based on a Service Oriented Architecture, and consists of the front-end part, along with its backend, both aiming to facilitate and ease the interaction of the user with all the analytics components and the Datastore. The front end will be built as a Single Page Application developed using the Angular framework [5]. Angular is "a JavaScript-based open-source front-end web framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications." The open-sourced full-fledged Angular framework offers multiplatform targeting without particular hardware or other software requirements. The specific framework is based on Typescript programming language and JavaScript (as output) which is compatible with all browsers and mobile devices. There is no need for backend components, as the business logic is integrated at the front end. For UI components and controls we will use the latest Angular Material library [6].

The Data Visualization Framework will also be developed with Angular Framework. Using a framework helps to achieve more functionalities in less time, as it provides components that can be reused, and makes developers work more efficient end easier.

With Angular framework, a lot of chart libraries can be used. One of these options could be, for example, amcharts library [7] that provides all possible charts mentioned in 4.1.2, among others. Any other chart library that can be used with Angular can be selected, and the final decision about it will be taken after the final set of charts is selected by end users to better fit their use cases.

Finally, the backend implementation of the PDT will be based on the Java 8 [8], which can be easily installed and run in different environments. Additionally, according to the Service-Oriented-Architecture paradigm, it will implement and expose various REST APIs that will be used by the other components of the PDT. The REST web services will follow the JAX-RS specification, as described in the corresponding JSR-331 [9], JSR-339 [10] and JSR-370 [11] Java specification requests. The library that the backend will rely on for the implementation of these specifications will be Jersey [12]. The deployment of the backend will be using an embedded Java servlet container, in order to avoid the need to maintain a whole application server, and package everything in a single Java Archive (Jar) that will ease the overall deployment. For the data repository of the PDT, we will rely on the LXS relational datastore that is the main pillar of the data management component of PolicyCLOUD.

5 Conclusions

This deliverable establishes the definition of the Public Policy (PP) in the context of the PolicyCLOUD project. It also presents the methodologies based on ontologies and semantic reasoning that can be used for modelling of PPs. We propose a structure for policy modelling based on Key Performance Indicators as the core of the structure, since KPIs enable a direct measurement of parts of the PP. Finally, we propose methodologies that can be developed to communicate the three different components: policy modelling, policy evaluation and population identification with the policy creation component, in order to feed the Policy Development Toolkit (PDT) and support policy-making.

References

[1] Gagnon, M. L., Labonté, R. (2010) "Framing Health and Foreign Policy: Lessons for Global Health Diplomacy", Globalization and Health, Vol. 6 (14). https://doi.org/10.1186/1744-8603-6-14

[2] Patton, C.V., and Sawicki, D.S., (1993) "Basic Methods of Policy Analysis and Planning", Vol. 7. Prentice Hall Englewood Cliffs, NJ.

[3] Scharpf, F.W., (1997) "Games Real Actors Play: Actor-centered Institutionalism in Policy Research". Westview Press.

[4] Moutselos, K., Kyriazis, D., Diamantopoulou, V., Maglogiannis, I., (2018) "Trustworthy Data Processing for Health Analytics Tasks", IEEE International Conference on Big Data, p. 3774-9.

[5] Angular, https://angular.io

[6] Angular Material, https://material.angular.io/

[7] AM charts, https://www.amcharts.com/

[8] Java 8, https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html

[9] JSR-331, https://jcp.org/en/jsr/detail?id=311

[10] JSR-339, https://jcp.org/en/jsr/detail?id=339

[11] JSR-370, https://jcp.org/en/jsr/detail?id=370

[12] Jersey, https://eclipse-ee4j.github.io/jersey/