beAWARE

Enhancing decision support and management services in extreme weather climate events

700475

D5.2

Basic techniques for emergency report generation

Dissemination level:	Public
Contractual date of delivery:	Month 18, 30 June 2018
Actual date of delivery:	Month 18, 30 June 2018
Workpackage:	WP5 Multilingual report generation
Task:	T5.2 - Setup of the report generation infrastructure T5.3 - Content selection and report discourse structure planning T5.4 - Multilingual linguistic surface report generation
Type:	Report
Approval Status:	Final
Version:	1.0
Number of pages:	42

This project has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No 700475

Filename:	D5.2_beAWARE_Basic techniques for emergency report_2018-06-30_v1.0

Abstract

This deliverable reports on developments in WP5 up to month M18 that contribute towards the completion of the first prototype of the beAWARE system. It focuses mostly on the multilingual surface realization aspects of the report generation procedure corresponding to T5.4. The setup of the report generation infrastructure (T5.2) is described briefly, as it is described in greater detail in D6.6. This report also describes the baseline content selection strategy developed as part of T5.3 and provides an overview of the work planned for the rest of the project -- to be reported in D5.3.

Co-funded by the European Union

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 700475

History

Version	Date	Reason	Revised by
0.1	4.6.2018	ToC, executive summary	Gerard Casamayor (UPF)
0.2	12.6.2018	First contents added concerning T5.3 and T5.4	Stamatia Dasiopoulou (UPF)
0.3	19.6.2018	Further contents related to T5.4	Simon Mille (UPF)
0.4	20.6.2018	Added description of progress in T.5.2	Jens Grivolla (UPF)
0.5	22.6.2018	Filled advanced generation methods section	Gerard Casamayor (UPF)
0.5	22.6.2018	Internal review	Tobias Helmund (IOSB) Stratos Kontopoulos (CERTH)
0.6	25.6.2018	Added abstract, introduction and summary	Gerard Casamayor (UPF)
1.0	27.6.2018	Changes following internal review	Gerard Casamayor (UPF)

Author list

Organisation	Name	Contact Information
UPF	Gerard Casamayor	gerard.casamayor@upf.edu
UPF	Simon Mille	simon.mille@upf.edu
UPF	Jens Grivolla	jens.grivolla@upf.edu
UPF	Stamatia Dasiopoulou	stamatia.dasiopoulou@upf.edu
UPF	Leo Wanner	leo.wanner@upf.edu

Executive Summary

This deliverable describes the functionality of the basic version of emergency report generation module developed for the 1st prototype of the beAWARE project. Advances in tasks T5.2, T5.3 and T5.4 are reported, covering the setup of the report generation infrastructure, the basic methods for content selection, and the methods for multilingual linguistic surface report generation respectively. This document focuses largely on the linguistic generation task. The setup of the generation infrastructure was largely covered in D6.6, while content selection and discourse structuring methods will be described in depth in the upcoming D5.3. However, an outline of future work in T5.3 is provided.

Abbreviations and Acronyms

BLEAU	BiLingual EvAluation Understudy
JSON	JavaScript Object Notation
KB	Knowledge Base
KBS	Knowledge Base Service
MRG	Multilingual Report Generator
MTT	Meaning-Text Theory
NLG	Natural Language Generation
OWL	Web Ontology Language
PSAP	Public Safety Answering Point
RDF	Resource Description FrameWork

Table of Contents

1 Introduction

Following an analysis of user requirements, the empirical study carried out as part of T5.1 and reported in D5.1 identified several types of reports for the various actors involved in emergency scenarios, i.e. authorities, first responders and citizens. The first prototype of the beAWARE system supports alerts and situation updates. Alerts are messages indicating changes in monitored indicators pertinent to the three pilots, flood, fire and heat wave emergencies. Alert messages are handled at the PSAP side enabling the authorities to choose from preselected messages or write their own, and consequently no content selection or linguistic generation support is needed for them.

The version of the report generation covered in this deliverable matches the functionality of the first prototype, which addresses the production of reports communicating situation updates during an emergency. Following several of the user requirements listed in D2.1 that pointed at the need or receiving real-time emergency reports, e.g. UR_103, UR_124, UR_327, the main goal of these updates is to enhance the situation awareness of the authorities overseeing an unfolding emergency. The reports being generated in this first prototype describe incidents detected following the analysis of multiple sources and modes of communication like social media, images, video and audio calls, thus addressing user requirements UR_217, UR_322, UR_324, UR_341, amongst others. The reports also provide information on the objects and people impacted by these incidents, as requested in UR_111, UR_112, UR_114, UR_201, UR_306, UR_318.

The current generation functionality is subject to certain limitations. First, its coverage is limited to specific types of incidents and participants (i.e. impacted or vulnerable objects) appearing in the pilots of the first prototype for the flood, fire and heatwave use cases. Albeit the linguistic resources for English have a broader scope and support additional input contents beyond those agreed for the first prototype pilots, linguistic support in other languages is more constrained to the pilots.

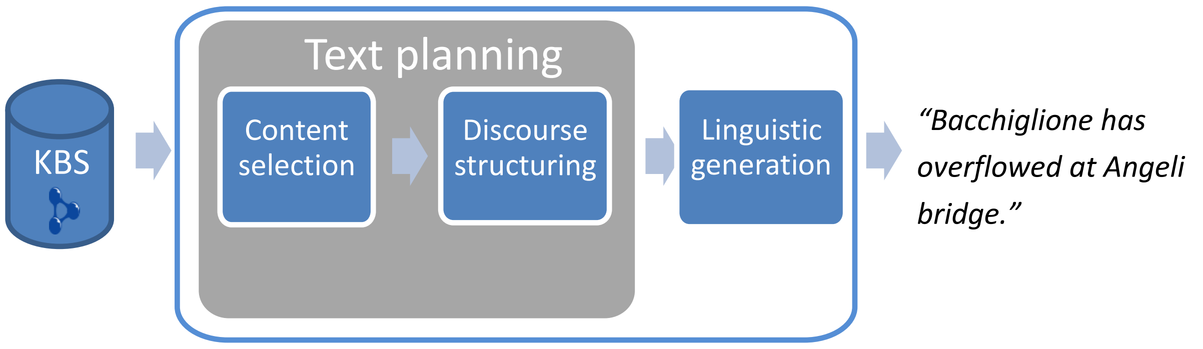


Figure 1: internal architecture of the MRG

As described in D7.2 and D6.6, report generation is implemented within the overall beAWARE system in the Multilingual Report Generation (MRG) service, a data analysis and processing component of the business layer. It receives requests for report generation on the message bus issued from the KBS service and containing ontological contents in JSON format. The contents received consist of a list of incidents belonging to nearby locations and with distinct timestamps. Each incident description details its type and the set of participating objects. These descriptions are obtained from the information extracted by the data analysis services, i.e. text analysis, speech recognition, image analysis, video analysis, and are semantically integrated by the KBS. Report generation processes the contents received in a request in two steps, first by addressing content selection and discourse planning, and then multilingual linguistic report generation.

For the following versions of the prototype, several types of additional information integrated or produced by the KBS are being considered, such as explanations, relevant weather data, and indications about the severity and priority of the reported incidents. In addition, report generation will also support the production of summary reports for authorities at the end of an emergency that provide a wrap-up of all incidents. Albeit not described in D2.1, summary reports have been brought up by partners as very useful in such contexts.

This document is structured as follows. Section 2 gives a brief description of advances in the set up pf the report generation infrastructure. Section 3 focuses on advances in content selection, while Section 4 describes the advances in linguistic generation. Section 5 details current and future work towards advanced methods for report generation. This deliverable concludes with a summary in Section 6.

2 Report generation infrastructure in beAWARE

As described in D6.6, the multilingual report generator integrates with the rest of the beAWARE platform by receiving report requests through the shared Kafka bus and sending the generated report in a Kafka message, according to the topics defined in D6.6. The incoming request messages contain all required information for generating the report, and the output is fully contained in the emitted Kafka message (as detailed in the following section), thus removing the need for direct communication with the databases used in the platform.

It is currently developed as a single component responsible for all languages and report types, internally selecting the appropriate modules for each scenario. It is implemented in Java and shares part of the integration code (Kafka communication, message parsing, etc.) with the text analysis modules developed by UPF. It is deployed as a container on beAWARE's Kubernetes infrastructure, using the project's Jenkins-based build and deployment system. While currently running as a single instance, horizontal scaling is easily achievable by simply deploying additional instances and using Kafka's consumer group functionality to balance requests between them.

3 Content selection in beAWARE

In this basic version of the emergency multilingual report generation (MRG) framework, where the focus lies on setting the grounds and consolidating the delivery of situation reports that describe the incidents detected from the continuous stream of incoming textual and visual inputs[1], content selection serves primarily as the means for ensuring that the generated reports reflect the latest updates. In particular, and as described in the following, as the analysis results of the incoming data are forwarded to the knowledge base, the knowledge service (KBS) aggregates these into incidents and issues respective report generation requests that indicate pertinent information about the incidents to be reported; based on this information, content selection determines which contents need to be forwarded to linguistic generation so that the verbalised reports afford the desired incidents' descriptions.

In the following, we go over the contents of the current KBS-generated report requests and describe which information aspects are used in order to determine which incidents are selected for inclusion into the emergency reports that are to be delivered through the PSAP. Representative examples, drawing upon the first prototype use cases, are also given.

3.1 KBS report request

As incoming image, video, audio and text data are analysed the results of their analysis are fed to the knowledge base service. The latter processes them and populates the knowledge base accordingly, while aggregating the detected incidents based on their location. Once a report needs to be delivered (for details on the KBS decision making the reader is referred to D4.2), the KBS issues a report request that lists the incidents that are relevant for the given location, along with meta-information, including among others, priority and severity indicators, the modality of the incoming data (i.e., image, video, text or audio), the time stamp of the referred incident, as well as, though mostly relevant for linguistic generation, the language in which the report is to be generated. Figure 2 illustrates an excerpt of a KBS report request in the JSON format used for the communication of the KBS with the multilingual report generator (MRG). As illustrated, the report request contains two incidents, a flood incident that was detected from visual data, an image in particular, and an overflow incident that was

[1] As aforementioned, sensor and weather data, as well as logical associations reflecting KBS inferences and explanations (e.g., "High fire risk. Strong winds and extreme temperatures forecasted. "), will be investigated and incorporated, as needed, incrementally in the report generation versions planned for the second and final prototypes.

detected from textual data. Incident and participant types, as well as the relations linking incidents to participants, follow the beAWARE ontology as described in D4.1.

```
{
"incidentID": "example_incident",
"position": {"lat": 39.648914, "long": -0.300993},
 "language": "it-IT" ,
  ....,
"incidents": [
      {
         "mediaType": "image",
         "priority": "low",
         "incidentType": "Flood",
         "timestamp": "2018-06-05T09:42:23Z",
         "severity": "low",
         "participants": [....]
       },
       {
         "mediaType": "text",
         "priority": "high",
         "incidentType": "Overflow",
         "timestamp": "2018-06-05T09:43:46Z",
         "severity": "medium",
         "participants": [ ... ]
       }
     ],
   }
```

Figure 2 Excerpt of KBS report request invoving two incidents.

In its current implementation, content selection considers two information aspects of the listed incidents, namely the modality of the data from which a given incident was detected, and the time at which the incident was reported, as captured by the "mediaType" and "timestamp" fields respectively. The following subsections describe how this information is used for determining which contents should be selected.

Information about the priority and severity has not been considered for the moment, as on one hand, the respective KBS decision mechanism is still rudimentary, leaving, as a result, these fields mostly undefined; on the other hand, the scope of the prototypical use case implementations targeted for the first prototype, does not afford the broadness required for robustly assessing how these indicators can be effectively put to usage, besides prioritizing in accordance with the higher values. Likewise, the use of information about the type of incidents, which now is only rudimentary used, and their participants falls also within the investigations planned for the next versions.

3.1.1 Report contents selection

The generated reports consist of a description, which captures the system detected incidents, and a title, which is used to label the overall incident as visualised in the PSAP map. The respective selection processes are as follows.

Report description contents

The selection of the description contents currently depends primarily on the modality of the originating input data and the temporal sequence of the referred incidents.

Specifically, if the report request contains only incidents that have been detected from visual data, namely images or videos, then the most recent incident is selected. For example, and drawing upon the first prototype flood use case, if at a time point t_1, an image is received for a given location, for which no previous incident has been reported, its analysis will result in the population of the knowledge base and the issuing of a report request that will include this sole incident (e.g., a flood incident involving two cars); content selection then, will select this incident by default. When in a later time point t_2, and for the same location, a second image is received, analysed and used to populate the knowledge base (e.g., a flood incident involving four cars and two people), a report request containing both incidents will be issued; in this case, content selection will choose for inclusion in the report only the second incident.

The main motivation underlying this decision is the lack of further information on whether the visually detected involved objects of the incidents refer to the same real-world entities across the series of detected incidents or whether they correspond to distinct entities. For instance, following the running example, if the second image referred to the two cars of the first incident and two new ones, then selecting both incidents for report would result in report about six impacted cars, where there would be only four. The determination of the identity of real-world may fall within the KBS scope, however, even if assuming that all detected cars are distinct, there are still further refinements pending, as reporting the increasing counting of impacted objects of a certain entity type is hardly efficient. Ongoing investigations include the usage of qualitative and more complex numerical descriptions, such as "several", "many" "at least", "up to", etc. Assuming the active deployment of the unfolding of the situation, the use of comparative descriptions such as "more" and "many more" will also be considered, possibly in addition with explicit references to the newly detected incidents (e.g. "Several cars impacted by the flood." and "Further reports about more impacted cars in Matteotti square.").

If the report request only contains incidents that originate from textual data, then all incidents are selected for explicit verbalization. The reason is that once the knowledge base decides that a report is to be generated, and since for the moment the deployment of priority and severity information is not yet fully in place, the selection of all incidents ensures that no potentially relevant, and much more critical, information is lost or overlooked. Given the basic versions currently available of text analysis, which does not currently cater for semantic abstraction, and of the knowledge base service, which aggregates incidents primarily based on spatial considerations, there is a risk for redundancy, as for instance, if the system receives two messages about the square being flooded and the square being underwater, these will be considered as two distinct incidents, and hence both of them will be included in the generated report; likewise, if two messages about the river overtopping the embankments in a given location are received, these are currently interpreted in the KB as two distinct occurrences, resulting in two incidents in the report request.

Besides enhancements pertinent to text analysis and the knowledge base service, redundancy risks, such as the aforementioned ones, will be largely mitigated by advances in the content selection per se, through the incorporation of techniques for the incorporation of temporal history criteria and for the following up of the incidents temporal evolvement. These will enable us to avoid repetitions, while possibly communicating more succinctly and naturally the situation; for example, assuming the flooding of a square and the reception of several messages as the flood persists, temporal content selection will provide the means to refrain from verbalising over and over again a report about the flooded square and instead report that the square is still flooded in a regulated manner (no gain in avoiding repeating that the square is flooded, if the system keeps repeating the square is still flooded) or even omit in, in view of new information (e.g. nearby subway station being closed).

Last, when the report request contains incidents of textual or audio origin as well as incidents of visual origin, in accordance with the current paradigm of affording report that update with respect to the latest incoming detected information, the timestamps are used in order to select the most recent one(s). Figure 3 illustrates an example sequence of input data, the respective report requests, and the resulting selected content for inclusion in the report description. The example draws upon the storyline for the flood pilot use case addressed within the first prototype. Each row stands for a time point, the detected concepts (incidents and participants) from the incoming data, the incidents included in the triggered report requests, and the incidents selected for inclusion in the report. All incidents are referring the same location, hence, the incremental incidents lists.

Time	Incoming data (modality: detected concepts)	Report request (incident: participants)	Selected incidents
T1	• Flood: Car (2) [image]	• Flood: Car (2)	• Flood: Car (2)
T2	• Flood: Car (7), Human(2) [image]	• Flood: Car (2) • Flood: Car (7), Human(2)	• Flood: Car (7), Human(2)
T3	• Flood: Square [text] • Overflow: Sewer [text]	• Flood: Car (2) • Flood: Car (7), Human(2) • Flood: Square • Overflow: Sewer	• Flood: Square • Overflow: Sewer
T4	• Flood: Car (4) [image]	• Flood: Car (2) • Flood: Car (7), Human(2) • Flood: Square • Overflow: Sewer • Flood: Car (4)	• Flood: Car (4)

Figure 3: Example incidents sequence and respective report requests and selected contents.

Report title contents

As far as the selection of the contents for the title of the report is concerned, two approaches have been explored in this basic version. The first one consists in using the selected incidents types and present them as a list of keyword descriptions. For example, assuming that two incidents have been selected for inclusion in the report, namely a flood incident and an overflow incident, the title, after the verbalization provided by linguistic generation, would be "Flood, overflow". In the absence however of the use of priority and severity information or information about how the current incidents relate and compare with the ones reported previously, this-keyword based approach was replaced for a more generic one, in anticipation of the completion of the ongoing investigations towards a more representative selection of the title contents. The generic title consists simply in the generic incident mention (which the linguistic generation that comes after renders in the requested language), augmented by the system incident ID in order to facilitate the cross-reference during the sequence of incidents visualization.

Summing up, the selection of sufficiently informative contents for reports title and description is bound to the advancement of content selection from the current update paradigm, whose main concern is the delivery of the currently detected information towards a full-fledged temporal-aware paradigm that will inform with respect to the unfolding situation in view of what has already happened and has been reported, and the joint consideration of information from the various modalities and sources, while taking into account priority and severity considerations.

4 Multilingual report generation

This section focuses on the extension of UPF's multilingual discourse generators developed for multilingual report generation in a series of European projects to an incremental expressive generator that fits the needs of beAWARE.

The present deliverable contains a report of the Approach (4.2) and the Implementation (4.3) followed in beAWARE. It also contains an Evaluation section (4.4), which reports on the evaluations performed for Prototype 1, and in the framework of various Natural Language Generation shared tasks that took place in 2017.

4.1 Related work

NLG from abstract input structures has a long tradition in exploring mechanisms to tailor its output to the needs of the targeted user, relying on user models that guide the selection of the content relevant to the user in question, the ordering of this content and the choice of the appropriate wording and style (see Wanner et al., 2010 for an overview).

State-of-the-art linguistic generators from ontological representations cope, as a rule, with either ontology verbalization, namely generation of natural language descriptions of ontology axioms, see e.g., (Androutsopoulos et al., 2013) or with preselected types of information - air quality for citizens or administration (Wanner et al., 2015), patient monitoring as support for medical personnel (Portet et al., 2009), technical device operation monitoring for surveillance personnel (Yu et al., 2007), or football match commentaries (Bouayad-Agha et al., 2012) - using predefined mappings between the targeted ontological and respective linguistic constructions. Leaving aside text planning, for linguistic generation, some proposals operate with rule-based systems, and others with sentence templates, which impose, however, serious limitations on the flexibility and coverage of linguistic constructions. Recently, data-driven linguistic generation has been proposed (Ballesteros et al., 2015), which although more flexible than rule-based generation, require considerable resources for training. However, all data-driven generators still start from shallow semantic structures, not genuine predicate-argument, let alone ontological, structures. On the other hand, independently of generation, ontology lexicalisation, i.e., techniques for expressing classes and properties of ontological representations in natural language (NL), has been subject of growing research interest. Models for interfacing ontology elements with corresponding lexical items and pertinent syntactic and semantic aspects have been developed. Most of them address the discovery of candidate lexicalisations for DBpedia. As a result, useful resources for lexicalising RDF/OWL descriptions that use the DBpedia vocabulary are made available. However, these resources cannot be easily ported and reused in other application domains due to potential conceptual and scope disparities between the DBpedia ontology and the domain ontologies in question.

However, to the best of our knowledge, no works are available as yet that would address the problem of the projection of complex ontological configurations onto lexicalized semantic structures, as needed in beAWARE.

4.2 Approach

Discourse generation starts from the ontological assertions that comprise the content inferred through the interplay of the interaction manager and the knowledge base. The generation is performed step by step, by successively mapping one level of representation onto the adjacent one: *Ontology -> Conceptual Structure -> Semantic Structure -> Deep-Syntactic Structure -> Surface-Syntactic Structure -> Morphologic Structure -> Linearised Structure -> Sentence.* The underlying linguistic framework is the Meaning-Text Theory (Mel'čuk, 1988), which foresees a very similar stratification in language description. In this subsection, we describe the role of each transition.

4.2.1 From Ontology to predicate-argument templates

The mapping of the ontological representation to a conceptual one (in the form of predicate-argument structures) is the first step towards the projection of the contents of the report request sent by the knowledge base service (KBS) to language-oriented structures. The grounding follows the frame semantics-based paradigm, which as described in D3.3, has also been adopted in the multilingual text analysis approach, under which events and situations are captured as *n-ary* relational contexts and the involved entities as participants with designated semantic roles. This ontological-to-conceptual grounding is in accordance with the Description and Situation (DnS) pattern of DOLCE+DnS Ultralite[2] (DUL), where *dul:Situation* objects correspond to reified relational contexts and the participating entities are associated with semantic roles via *dul:classifies* properties. Semantic roles correspond to instances of the *dul:Concept* class.

Following the aforementioned frame-based paradigm, each instance (occurrence) of the beAWARE ontology class Incident is projected to a linguistic predicate, while its participants (as captured via respective "participant_is_involved_in" property assertions) become its linguistic arguments. The projection is based on prototypical predicative templates that enable us to map configurations of incident contexts (i.e., frame-based configurations of specific types of incidents, with respective types of participants and their roles) to respective predicate-argument structures, while also reflecting, through corresponding features, information pertinent to linguistic generation. The latter include, among others, information

[2] http://www.ontologydesignpatterns.org/ont/dul/DUL.owl

about cardinality, which is projected accordingly using the feature "number" (i.e., singular vs plural); information about whether the entity under consideration is a named entity or not, which is reflected using the part-of-speech (proper noun vs common noun); information about the ontological type of the entities, which is reflected through the use of the "class" feature, and so forth.

Figure 4 shows an excerpt of an example report request that refers to a cracking incident of the embankment at Angeli bridge, where for simplicity only information relevant to the projection of ontological representations to linguistic predicate-argument structures is shown. The agreed upon JSON format used for communication between the KBS and MRG serializes the ontological information according to the following conventions: the "incidentType" statement reflects the existence of an incident instance of the respective class (e.g., "incidentType": "Crack" denotes the occurrence of an incident of the class "Crack"); the "participants" statements capture the entities involved in the incident (impacted objects, locations, etc.); each participant entry has a "type", which denotes its ontological class (e.g. "type": "Bridge" indicates a instance of the class Bridge), a "label", which in case of Named Entities holds the respective name (e.g., "label": "Angeli bridge") and is being null otherwise, a number, which captures cardinality information, and "role", which captures the semantic role of the entity (e.g., "role": "Location") and which in case of underspecified participating entities (e.g., when a car is detected in an image of a flooded square) has the value "null".

```
"mediaType": "text",
 ....,
 "incidentType": "Crack",
 "participants": [
  {
     "detections": [ { ...  } ],
     "number": "singular",
     "label": "Angeli bridge",
     "role": "Location",
     "type": "Bridge"
  },
  {
     "detections": [ { ...  } ],
      "number": "singular",
      "label": "null",
      "role": "A2",
      "type": "Embankment"
  },
 ]
```

Figure 4 Report request excerpt illustrating, in JSON format, the ontological representation of a cracking incident that is to be verbalized in the generated report.

Upon reception of an incident and its participants, provided of course that this has been selected for verbalization during the content selection phase described in Section 3, all information fields are parsed and the of incident and participants' types and roles are used for selecting the appropriate predicate template. In the running example, this corresponds to the template affording a location and a patient (i.e., a participating entity that undergoes the event denoted by the incident type); once the predicative template has been selected, then it is populated as needed with additional information regarding the number, presence of named entities, and so forth. The resulting predicative-argument structure is shown in Figure 5, with the semantic dependencies between the nodes (second argument *A2* and *Location*) and the feature/value structure associated to the node Angeli_bridge (which contains information about the fact that it is a proper noun and a member of the class *Bridge*).

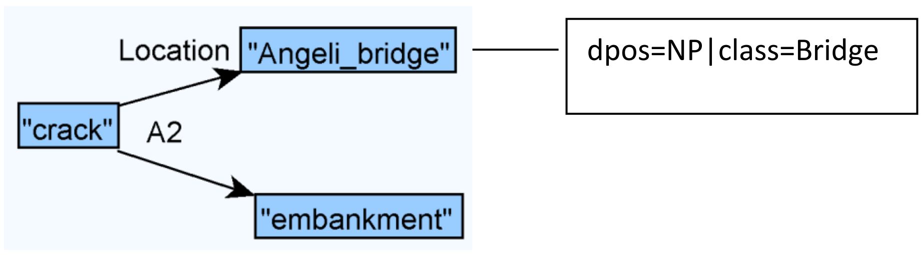


Figure 5 Conceptual structure for the cracking incident of the embankment at Angeli bridge.

4.2.2 From Conceptual Structure to Semantic Structure (SemS)

The conceptual structure is mapped to a language-specific structure according to the available meanings (*semantemes*) in the concerned language (English, Spanish, Italian and Greek). In order to illustrate the difference between a concept and a semanteme, consider the case of the concept of *measurement*. For example, the wind, as a physical event, can be measured, and this can be expressed in combination with the meaning *speed* in English. By contrast, in some languages, no meaning is available in order to realise *speed* in combination with *wind*. Mentioning *wind* with a rating is enough in order to understand that we are talking about wind speed. In English too it is actually common not to mention *speed*, and the organisation of the concepts must allow for choosing one way or another of combining the meanings. In theory, a semanteme can be lexicalised by many different words, see for instance the semantic dictionary entry 'CAUSE' shown in Figure 6.

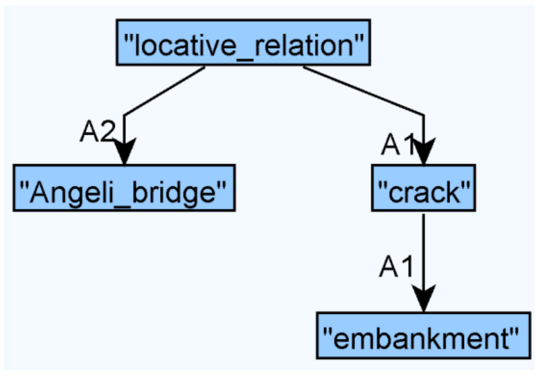
```
CAUSE { lex = cause_N | lex = cause_V | lex = contribute |lex = responsible |
lex = due | lex = because | etc.}
```

Figure 6: Entry for 'CAUSE' in the English semantic dictionary

However, in beAWARE, a simplified and more practical view has been applied, considering that lexical units such as "cause_V" (*cause* as a verb) are the basic meaning units in the semantic structure.

The semantic structure is unambiguous: each semanteme is the argument of a predicate and is numbered by the *valency* (or subcategorization frame) of the predicate, through the relation linking the two of them. Each language has its own set of predicates, and each predicate has its own valency.

In order to realize a sentence, it is necessary to give it a communicative orientation: *what are we talking about? What do we say about it?* The former is marked as theme of the sentence, the latter as rheme, based on the conceptual relations and the nodes in each connected graph. Each semanteme is included in a communicative span (theme, rheme, specifier), which can contain any number of semantemes.



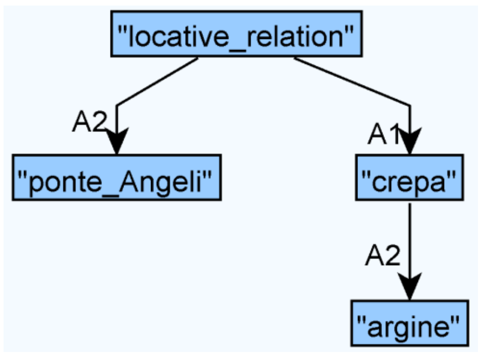


Figure 7: Semantic structures (left: English, right: Italian) that correspond to the conceptual structure in Figure 5.

In **Error! Reference source not found.**, the node "crack/crepa" has been identified as being the main node of the sentence. All the nodes have been assigned a part of speech and linked to an entry in a lexical resource: "crack" is linked to the entry "crack_VB_01" according to the PropBank nomenclature (Kingsbury and Palmer, 2002); this information is available in the feature/value structure associated to each node, as shown in Figure 5 with "Angeli bridge". In Italian, no verb is available for this meaning and "crepa" is considered as a noun, and linked to the entry "crepa_NN_01".

4.2.3 Discursive Unit aggregation

If several discursive units such as the one shown in **Error! Reference source not found.** (the group of nodes called “Sentence” is what we call here a discursive unit) are fed to the generator, they will each be realized by default as independent sentences. In order to group different units into complex sentences, we implemented a new graph-transduction module that performs aggregation in two steps. First, we look for predicates in the input: if the subject or object arguments of two unlinked predicates have the same relation with their respective predicates, they will be coordinated. For instance, if the generator receives two separate units corresponding to *Five cars are impacted* and *Four persons are impacted*, these two units will be rendered as one single sentence *Five cars and four persons are impacted.*

Second, we check if an argument of a predicate appears further down in the ordered list of discursive units. If so, the units are merged by fusing the common argument; during linguistic generation, this results in the introduction of postnominal modifiers such as relative and participial clauses or appositions (e.g. *A flood, in which five cars are impacted, has been reported.*). In order to avoid the formation of heavy nominal groups, we allow at most one aggregation by argument. Referring expressions are introduced during linguistic generation (see Section 4.2.5).

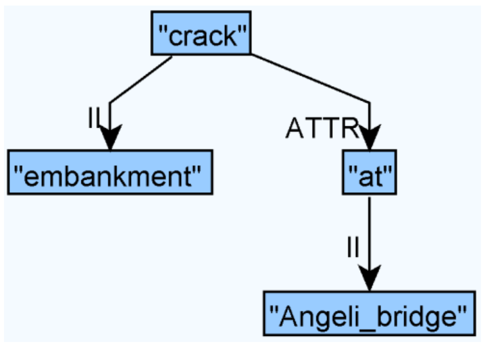
4.2.4 From Semantic Structure to Deep-Syntactic Structure (DSyntS): lexicalization and sentence structuring

During the transition from Semantic Structure to Deep-Syntactic Structure, the semantic graph with the communicative structure is mapped onto a tree: the main node of the rheme will be the head of the sentence, that is, the main verb, while the rest of the rheme generally corresponds to the objects and adverbs, and the theme to the syntactic subject. From this root, the whole tree is built node after node.

A lexicon indicates what a syntactic predicate requires to form a correct sentence in a language (syntactic combinatorial); for instance, the verb *cause*, as most verbs, requires a noun or a non-finite verb as its subject. The subject may also have arguments, also restricted by the syntactic combinatorial.

Only meaningful units (*lexemes*) are part of the DSyntS; in other words, there are no grammatical units at this point (bound prepositions, auxiliaries, etc.). The DSyntS can also contain abstract lexemes (*collocates)*, formalised as Lexical Functions (LFs). Those LFs are given a value (a concrete label) during the DSyntS-SSyntS mapping (see Section 4.2.5) based on the combination with other words. For instance, the abstract lexeme *Magn*, which means 'a high degree of', would be realised as 'heavy' in combination with 'rain', but as 'intense' in combination with 'heat'. Support verbs are also represented at this level under the form of lexical functions: the noun "crepa" (lit. 'crack') requires the introduction of the support (i.e. empty) verb "mostrare" (lit. 'show') if used as the main node in a sentence; this label corresponds to the value of the lexical function "Oper2" applied to the keyword "crepa".

Figure 8 shows that "crack" is the main syntactic node of the sentence, in English, while the abstract support verb "Oper2" is the main node of the Italian sentence, and that all other elements are organised around each of these nodes. Instead of a pure predicate-argument structure, the edge label reflects the syntactic structure of the sentence, in particular the opposition between arguments (*I*, *II*, *IV*) and modifiers (*ATTR*). Meanings such as *locative_relation* are verbalised as the prepositions "at" and "a", in English and Italian, respectively.



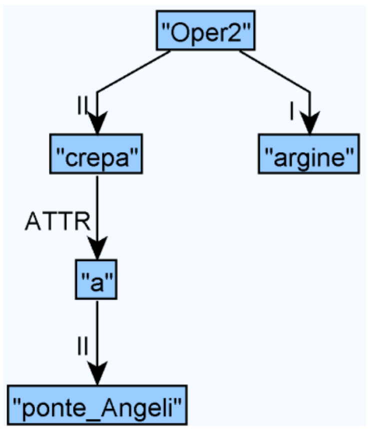


Figure 8: Deep-syntactic structures (left: English, right: Italian) that correspond to the semantic structures in **Error! Reference source not found.**.

4.2.5 From Deep-Syntactic Structure to Surface-Syntactic Structure (SSyntS): introduction of idiosyncratic information

Once the structure of the sentence has been defined and all the meaningful words chosen, non-meaningful, or *grammatical*, units have to be introduced. In the lexicon, an entry of a word indicates which preposition, case, finiteness, number, etc. has to be inserted on its dependent. For instance, if there is the DSynt configuration in which a verb only has a second argument, as it is the case in the English example, a passive auxiliary ("be" in English) must be introduced. Other non-lexical nodes are introduced, such as governed conjunctions and prepositions, other auxiliaries and modals, determiners, expletive subjects, etc.

Lexical Functions (LFs) must also be resolved during this transition: most words of the lexicon are the keyword of one or more LFs. The value(s) of the LFs is stored in the entry of a word: “mostrare” (‘show’) as the value of the LF *Oper2* in the entry of “crepa” (‘crack’), for instance.

Finally, the generic syntactic relations found in DSynt are refined into more idiosyncratic relations which convey very accurate syntactic information, instead of semantic, as it is the case with the argument numbers. For instance, the DSynt relation ‘I’ can be mapped to *SBJ* (subject) if the verb is active, *OBJ* (object) if the verb is passive, *NMOD* if the head is a noun, etc. A *SBJ* has the syntactic property to trigger an agreement on the verb, to undergo demotion in some conditions, and to be realised before the verb in a neutral sentence. An *OBJ*, on the contrary, appears by default after the verb, can undergo promotion, and is cliticizable with an accusative pronoun. A *NMOD* cannot be promoted or demoted, does not trigger any agreement and always has to be realised to the right of its governor if it is not a determiner.

Figure 9 shows the surface-syntactic structures with functional words and language-specific syntactic relations.

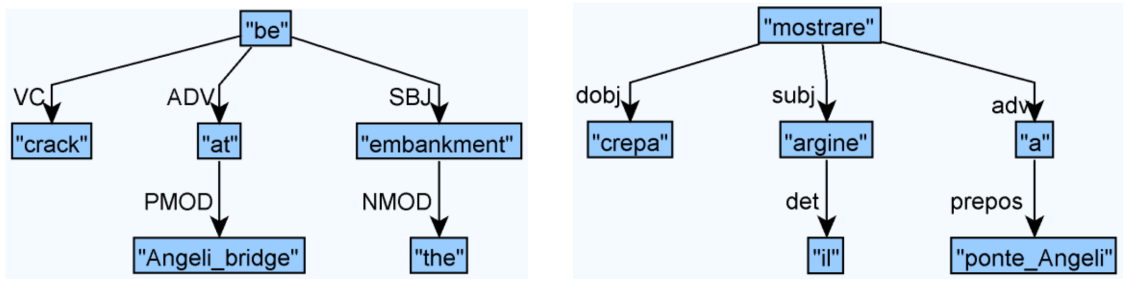


Figure 9: Surface-syntactic structures (left: English, right: Italian) that correspond to the deep-syntactic structures in Figure 8.

4.2.6 From Surface-Syntactic Structure to Morphologic Structure (MorphS)

Thanks to the idiosyncratic set of surface-syntactic relations, all agreements and orders between the components of the sentence can be resolved. Every word of the sentence contains all the indications to make the production of the final form possible. This can be done either by creating a full-fledged dictionary containing entries under the form, e.g., ‘be<VB><IND><PRES><3><SG> = is’, or by using some automata based on inflection schemas, such as Two Level Morphology, to automatically inflects forms. Capitalisation can be introduced when necessary.

With these idiosyncratic surface-syntactic relations, all orderings between the components of the sentence can also be resolved; for instance, in a given language, subject goes before its governing verb, a determiner before its governing noun, etc.

Figure 10 shows that at this level, the words carry all the necessary information for inflection (e.g. the attribute editor for the word “go”): part-of-speech, mood, tense, person, number. The precedence relations are shown in red.

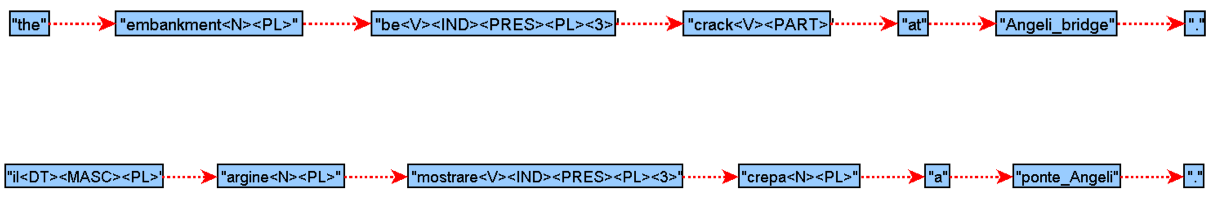


Figure 10: Linearized morphological structures (top: English, bottom: Italian) that corresponds to the surface-syntactic structure in Figure 9.

4.2.7 From Morphologic Structure to Sentence

Once all the words are ordered, punctuation marks are introduced (periods and commas around descriptive modifiers), the final form of the words is retrieved and the sentence is ready to be delivered to the next module. In the case of the running example shown throughout this section, the output would be “The embankment is cracked at Angeli bridge” in English, and “Gli argini mostra crepe a ponte Angeli” in Italian.

4.3 Implementation in beAWARE

In this section, the numbers within parentheses refer to the different mappings sketched in the previous section. We describe the methodologies followed for the deep and surface generation processes, and report on the development of a new environment for the implementation of the rule-based generator.

4.3.1 Deep Generation

For the deeper stages of generation, i.e. for going from the ontology to the deep-syntactic structures used for surface generation, rule-based modules have been developed. The first step consists in the frame-based translation rules for mapping the ontology representations of the incidents listed in the report request to respective conceptual structures in the form of predicate-argument structures for step (4.2.1); graph-transduction grammars for steps (4.2.2, 4.2.3, 4.2.4) follow.

As far as the mapping to conceptual structures is concerned, and aiming primarily to afford coverage for the first prototype scope[3], an initial set of 30 predicative templates has been defined. These afford coverage for the key incidents and the vulnerable objects of interest that are addressed within the three use cases selected for the first prototype, i.e., flood, overflow, collapse, square, bridge, people, car, fire, traffic congestion, to name but a few. We distinguish between predicate-argument structures capturing incidents originating from visual data from those that capture incidents originating textual data, as for the former, due to the absence of information about the role of the detected involved participants to the incident, the predicative templates contain predefined information on their semantic roles. For example, in the case of cars detected in an image from a flooded area, the corresponding predicate-argument template, stipulates by default that the cars are the impacted entities (as opposed to e.g., being the agentive entities causing the flood incident). The mappings between ontological contents received from the KBS and predicative templates are manually crafted and realise the inverse transformation of the one applied during the analysis of the textual inputs.

In combination with the graph-transduction rules, a semantic dictionary, that includes equivalences between the entities in the semantic repository and the words of the different languages of the project (e.g. *locative_relation*, see previous section), is required.

4.3.2 Surface generation

For generation from deep-syntactic structures (4.2.5, 4.2.6, 4.2.7), two different approaches have been implemented, both based on a pipeline of graph transducers, which convert the output of the text planning stage into a well-formed text, and both being implemented as part of the MATE tools[4].

The first approach consists of manually crafted multilingual graph-transduction grammars for each transition between two consecutive layers. In combination with the rules, dictionaries of two different types are required: one that describes syntactic properties of these words (lexical dictionary), and one that contains the inflection patterns of each word (morphological dictionary). We manually crafted language-specific dictionaries for all the beAWARE languages that cover the dialogues foreseen for the final prototype.

[3] The conceptual predicate-argument structures are language independent, yet following the multi-layer generation framework, corresponding surface-realization coverage needs to be in place in order to deliver verbalizations in the four targeted languages.

[4] http://code.google.com/p/mate-tools/

In order to reach a large-coverage, we have been experimenting on the extraction of subcategorization patterns from English lexico-semantic resources such as PropBank (Kingsbury and Palmer, 2002), NomBank (Meyers et al., 2004) and VerbNet (Schuler, 2005), extending the approach described in (Mille and Wanner, 2015). The sample entry in Figure 10 shows the syntactic properties of the verb "give", which has three nominal arguments, the third one being introduced by the preposition "to".

```
"give_VB_01":_verbExtArg_{
        vncls = "13.1"
        pbID = "give.01"
        pbsenseID = "01"
        lemma = "give"
        gp = {
            I = {
               pos = NN
               rel = SBJ
            }
            II = {
               pos = NN
               rel = OBJ
            }
           III = {
               pos = NN
               rel = IOBJ
              prep = "to"
            }
        }
}
```

Figure 11: subcategorization pattern of the verb "to give"

The method has proven to be useful in English, but not successful enough to be applied to other languages so far, especially given the lack of parallel resources in PropBank and NomBank for these languages. This graph-transduction system has been named the Fabra Open Rule-based Generator - FORGe (Mille et al. 2017).

The second approach consists in performing some transitions with statistical modules trained on annotated data. Indeed, by aligning node by node a parallel corpus of two consecutive levels of representation, it is possible to apply Machine Learning techniques and obtain models for a statistical generator. For this purpose, multilingual corpora containing MTT-based linguistic annotations have been annotated manually and automatically by UPF (see next subsection).

In order to project a DSyntS onto its corresponding SSyntS in the course of generation (where both DSyntSs and their corresponding SSyntSs are stored in the 14-column CoNLL'09 format

(Hajič et al., 2009)), the following types of actions need to be performed by the statistical generator (see Figure 12 for an illustration of steps 1-5):

1. Project each node in the DSyntS onto its SSynS-correspondence. This correspondence can be a single node, as, e.g., *job*->[*NN*] (where *NN* is a noun), or a subtree (hypernode, known as syntagmain linguistics), as, e.g., *time*->[*DT NN*] (where *DT* is a determiner and *NN* a noun, as in *the time*) or *create*>[*VAUX VAUXVB IN*] (where *VAUX* is an auxiliary, *VB* a full verb and *IN* a preposition, as in *to have been created*). In formal terms, we assume any SSyntS-correspondence to be a hypernode with a cardinality ≥1.
2. Generate the correct lemma for the nodes in SSyntS that do not have a 1:1 correspondence with an origin DSyntS node (as *DT*and *VAUX*above).
3. Establish the dependencies within the individual SSyntS-hypernodes.
4. Establish the dependencies between the SSyntS-hypernodes (more precisely, between the nodes of different SSyntS-hypernodes) to obtain a connected SSyntS-tree. For Spanish, after the DSyntS–SyntS we apply transition rules for the generation of relative pronouns that are implied by the SSyntS. Since we cannot count on the annotation of coreference in the training data, we do not treat other types of referring expressions. The lemmas of nodes with 1:1 correspondence are the same in both structures.
5. Establish the order between words given the surface-syntactic structure.
6. Inflect all the words.

The models and JAVA source code for the statistical deep sentence generator can be found here: https://github.com/talnsoftware/deepgenerator.

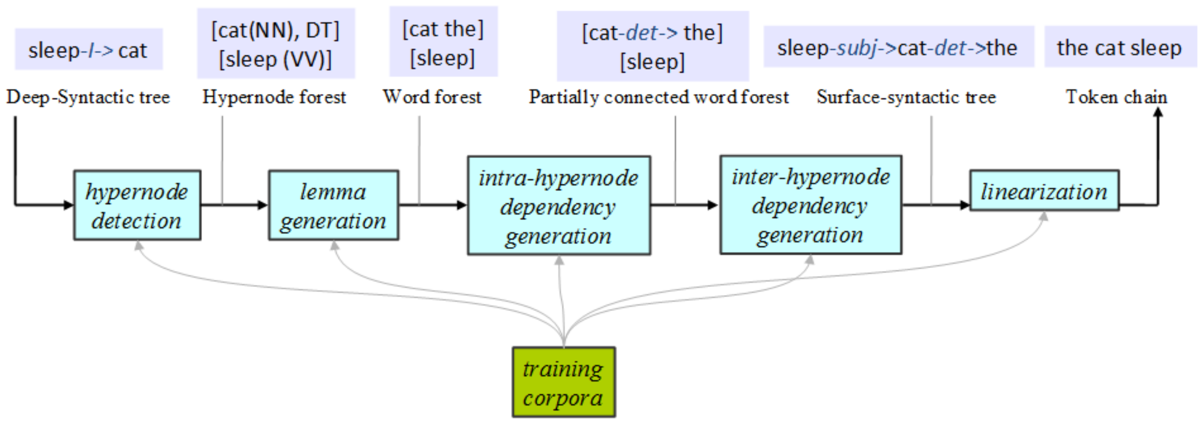


Figure 12: Workflow of the data-driven generator.

Both the rule-based and the statistical generators are based on the MTT (Meaning-to-Text) model.

Another strand of research in the context of this task concerns the development of a novel stochastic linearization strategy. In contrast to current state-of-the-art linearizers, our implementation takes the distinctive features of different types of syntactic dependency relations into account. We have not been able to complete this research during the first part of the project.

The generation module covers all languages involved in interactions with the Knowledge Base in the final prototype, that is, English, Italian, Greek, and Spanish.

4.3.3 Compilation of multilayered annotated corpora

For the needs of the advanced generation system, we create datasets that allow for learning on how to transfer Deep-Syntactic structures into Surface-Syntactic structures, and for linearizing the Surface-syntactic structures. For this, multilingual corpus annotation is being carried out, for training the required multilingual generators.

As mentioned in the previous subsection, the development of both statistical analysers and generators is based on the Meaning-Text Theory (MTT) (Mel'čuk, 1988), a linguistic framework that foresees various levels of representation, all based on dependencies. Having good quality data at hand is crucial for training efficient machine-learning systems, which is why Task 5.4 (*Multilingual linguistic surface report generation*) is partly centred on the annotation of resources.

Parallel annotation of morphological structures (MorphS), surface-syntactic structures (SSyntS), and deep-syntactic structures (DSyntS) is carried out in the different languages of beAWARE. MorphS consists of an ordered (linearized) structure, each node containing all the morphological information needed to be inflected, such as coarse-grained and fine-grained part-of-speech, gender, number, tense, aspect, finiteness, mood, person, etc. (Figure 10); SSyntS consists of a non-linearized structure containing all the words in a sentence (even the functional ones) related through language-specific relations (Figure 9); DSyntS consists of a structure in which just the lexemes (meaningful units) appear, related through language-independent relations. That is, all functional (i.e. non-meaningful) units are removed compared to the full sentence: definite and indefinite determiners and auxiliaries are replaced by attribute/value pairs on the concerned nodes, while punctuations and governed prepositions, and conjunctions are simply removed. In addition to attributive (ATTR, APPEND) and coordinative (COORD) relations, the dependency relations also encode predicate-argument information, through the assignment of an argument slot (I, II, III, etc.) in the valency (sub-categorization *framework*) of its governor predicate (Figure 8).

All the layers and the original texts need to be aligned sentence by sentence and node by node, which can be done thanks to unique identifiers associated to each sentence and node. We currently have available multi-layered corpora for Spanish, English and Italian at different stages of advancement; Greek will be tackled in the next months.

For English, we use both the Penn Treebank 3 converted to dependency trees (Johansson & Nugues, 2007), and the Universal dependency data (Nivre et al., 2016) which we use as surface-syntactic and morphologic annotations. We automatically derived a first version of the deep annotation from the surface annotation using graph transduction grammars implemented in the MATE environment (developed by TALN-UPF). During the mapping, we removed all determiners, auxiliaries, "*that*" complementizers, infinitive markers "*to*", punctuations and functional prepositions. The PTB-based treebank currently contains 41,678 sentences, and the UD-based treebank 14,981 sentences.

For Italian, we follow the same method as for English; we start with the existing surface-syntactic corpora (also UD-based) from which we automatically derive the deep-syntactic annotation thanks to graph transduction grammars. Since the surface-syntactic annotation is quite similar to the English one, it is possible to re-use the English mapping, augmented with a few language-specific rules; the Italian corpus currently contains 13,838 sentences.

For Spanish, we are extending the AnCora-UPF dependency Treebank (Mille et al., 2013). In AnCora-UPF, each layer is independently annotated, and the annotation is manually validated. For the surface-syntactic annotation, there are several dependency tag sets available, with many or few tags. The different dependency tag-granularities obtained different annotator-agreement values, but they were between 89.4% and 92.26% from the most fine-grained to the most coarse-grained. Those different granularities will be tested in order to find the optimal tag set for the statistical generation. The treebank currently contains around 10,000 sentences, 6,500 of which have been annotated in the framework of beAWARE. We also performed an automatic conversion of the Spanish UD-based dataset.

Table 1 summarizes the total number of sentences comprising the annotated corpora compiled within beAWARE during the first half of the project.

Language	Representation Layer	Annotation	Number of Sentences
English-PTB	DSynt	Automatic	41,678

English-UD	DSynt	Automatic	14,981
Italian-UD	DSynt	Automatic	13,838
Spanish-UD	DSynt	Automatic	16,087
Spanish-UPF	SSynt, DSynt	Manual	6,500

Table 1. Summary of annotated corpora compiled within beAWARE.

4.3.4 A new graph-transduction environment

We have been re-implementing the graph transducer MATE (Bohnet and Wanner, 2010) in order for the transduction rules to be more expressive and compact, as well as for the tool to perform the transductions faster; we will refer to this new tool as "MATE-2" in this section.

MATE is a graph transducer programmed in Java. It contains different editors for graph construction, rule and lexical resource writing, a debugger, as well as a tool for regression tests. The rules (and their corresponding conditions) match a part of an input graph, and create a part of the output graph. The main problem with MATE is its speed, which is not adequate for a real-time system as beAWARE. MATE-2 is currently in a very advanced state but no publication describes it yet. Figure 13 shows a project open in MATE-2.

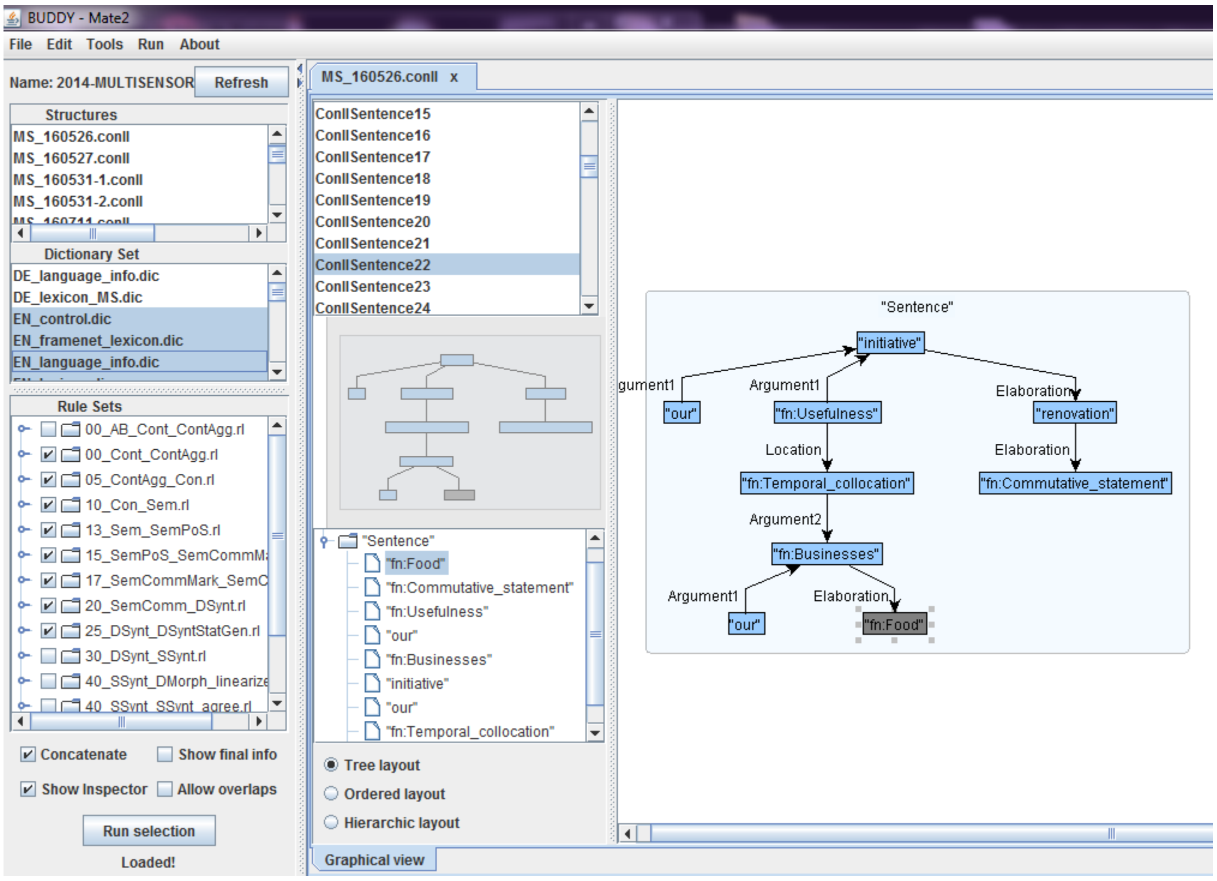


Figure 13: A screenshot of MATE-2 (Graph Editor view)

MATE-2 contains:

- A Project Browser

It is used to open and navigate in the different resources of a project. It's the part on the left, which contains 3 list fields, some options and the Run Selection button. Each list field corresponds to a resource used for the transduction:

  - Structures: a list of input structures on which the rules can be applied in order to create new structures.
  - Dictionary Set: a list of lexical resources used by the rules.
  - Rule Sets: a list of rule sets (= grammars); each rule set performs one transduction.

- A Resource Editor

Each of the three resources can be opened in an editor tab, by double clicking on it. The Graph editor contains 5 fields (see Figure 13):

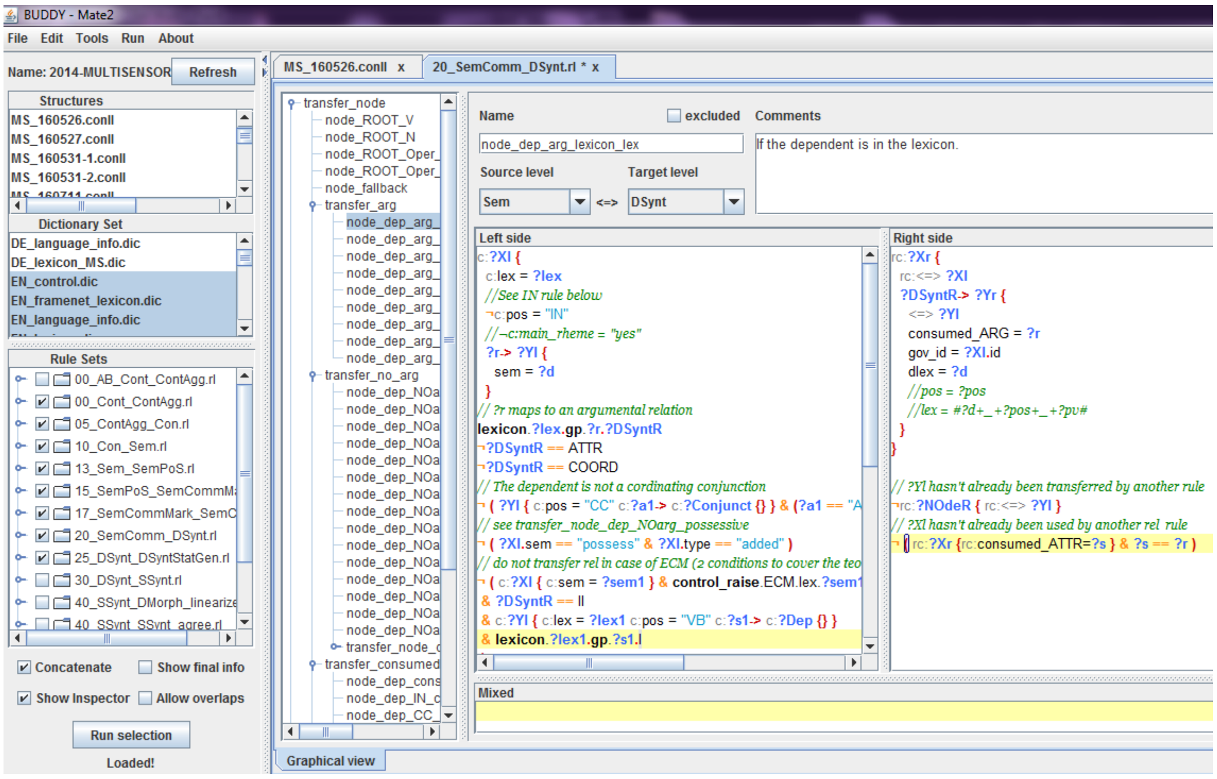
  - Graph List: the list of graphs contained in one file (top left)

- Graph Global View: the global view of the selected graph (middle top left)
- Graph Node List: the list of nodes of the selected graph (middle bottom left)
- Graph layout options (bottom left).
- Graph View: the complete graph view of the selected graph (right)

The Rule editor contains 2 main fields and some parameters (see Figure 14):

- Rule List: the rule tree (left)

Rule View: the complete rule view of the rule selected in the rule tree (right).



Name: node_dep_arg_lexicon_lex

excluded: ☐

Comments: If the dependent is in the lexicon.

Source level: Sem <=> Target level: DSynt

Left side

```
c:?Xl {
 c:lex = ?lex
 //See IN rule below
 ¬c:pos = "IN"
 //¬c:main_rheme = "yes"
 ?r-> ?Yl {
  sem = ?d
 }
// ?r maps to an argumental relation
lexicon.?lex.gp.?r.?DSyntR
¬?DSyntR == ATTR
¬?DSyntR == COORD
// The dependent is not a cordinating conjunction
¬ ( ?Yl { c:pos = "CC" c:?a1-> c:?Conjunct {} } & (?a1 == "A
// see transfer_node_dep_NOarg_possessive
¬ ( ?Xl.sem == "possess" & ?Xl.type == "added" )
// do not transfer rel in case of ECM (2 conditions to cover the teo
¬ ( c:?Xl { c:sem = ?sem1 } & control_raise.ECM.lex.?sem1
 & ?DSyntR == II
 & c:?Yl { c:lex = ?lex1 c:pos = "VB" c:?s1-> c:?Dep {} }
 & lexicon.?lex1.gp.?s1.
```

Right side

```
rc:?Xr {
 rc:<=> ?Xl
 ?DSyntR-> ?Yr {
  <=> ?Yl
  consumed_ARG = ?r
  gov_id = ?Xl.id
  dlex = ?d
  //pos = ?pos
  //lex = #?d+_+?pos+_+?pv#
 }
}

// ?Yl hasn't already been transferred by another rule
¬rc:?NOdeR { rc:<=> ?Yl }
// ?Xl hasn't already been used by another rel  rule
¬ ( rc:?Xr {rc:consumed_ATTR=?s } & ?s == ?r )
```

Mixed

Figure 14: A screenshot of MATE-2 (Rule Editor view)

The Lexicon editor's advanced view is still at an early development stage at this point.

4.4 Results and evaluation

For the purposes of beAWARE, advanced (statistical) generation systems were successfully developed; however, these systems tend to be greedy in terms of resources, and the texts they output cannot be tuned to the particular needs of the general pipeline. Thus, we dedicated a large part of our efforts to the rule-based system, which, instead of being a simple baseline for the advanced generator, ended up being the main generation system in the project.

In this section, we present three different evaluations:

- a basis for upcoming comparative evaluation of the (basic) rule-based system across prototypes, summarizing the improvements made for each prototype;
- several quantitative and qualitative evaluations of the rule-based system in 3 shared tasks;
- a quantitative evaluation of the advanced generation system, which will be based on questionnaires using the Likert scale and will be described in D5.3.

4.4.1 Basis for comparative evaluation of the (basic) rule-based system FORGe

		First Prototype
Languages supported		EN, IT, ES, EL
Number of rules and % of language-independent rules	ALL	- **Con-SMorph** (1,373) : 70%
		• Con-Sem (416) : 94% • Aggregation (212) : 91% • Sem-DSynt (177) : 75% • DSynt-SSynt (307) : 39% • SSynt-DMorph (172) : 48% DMorph-SMorph (89) : 55%
Main linguistic phenomena supported	EN	• Advanced sentence structure: ○ Argumental dependents ○ Circumstances ○ Coordinations ○ Embedded clauses ○ Complex syntactic structures • Lexicon-based introduction of functional elements: ○ Functional prep/conj ○ Modals and auxiliaries • Verbal agreements • Advanced linearization • Basic semantic and syntactic aggregations • Nominal compositionality
	IT	• Basic sentence structures: ○ Argumental dependents ○ Locative circumstancials • Lexicon-based introduction of functional elements: ○ Functional prep/conj ○ Auxiliaries ○ Determiners

		• Verbal, adjectival and det. agreements • Basic linearization
	ES	• Basic sentence structures: ○ Argumental dependents ○ Locative circumstancials • Lexicon-based introduction of functional elements: ○ Functional prep/conj ○ Auxiliaries ○ Determiners • Verbal, adjectival and det. agreements • Basic linearization
	EL	• Basic sentence structures: ○ Argumental dependents ○ Locative circumstancials • Lexicon-based introduction of functional elements: ○ Functional prep/conj ○ Auxiliaries ○ Determiners • Verbal and det. agreements • Basic linearization
Number of lexical units in lexicon	EN	41,955
	IT	63
	ES	1,852
	EL	20
BLEU score	**EN**	**35,53 (+12%)**

Table 2: Comparison between P1 and P2 generators. The table summarizes the improvements of the components of the generator during the course of the project

The rule-based generator had some of its components in place at the beginning of the project. Here is a summary of the improvements performed since the beginning of beAWARE:

Con-Sem (mapping from ontology-oriented to linguistic structures): we implemented specific rules to verbalize some particular concepts in each language (e.g. hours), to account for coordinated constructions, and to find the root of the syntactic tree in a more efficient way.

Sem-DSynt (definition of sentence syntactic structure): the variety of types of syntactic structures to be generated has been increased; in particular, different combinations of coordinations with other types of syntactic structures such as embedded clauses, shared arguments, or circumstancials.

DSynt-SSynt (introduction of idiosyncratic information): the number of rules doubled between the first two prototypes due to the fact that more languages are covered, and many rules are language-specific (cf. third row of the table - % of language-independent rules), as for instance the rules from the DSynt-SSynt mapping, for introducing the fine—grained syntactic relations and the functional elements (auxiliaries, modals, functional prepositions and conjunctions, determiners). Rules that perform a post-processing of the surface-syntactic structures have also been added, in order to correct possible errors during previous steps: for instance, disconnected nodes are reattached to the most probable governor, conjunctions or prepositions without dependents are removed, etc.

SSynt-DMorph (resolution of word ordering and morpho-syntactic agreements): the rules at this level are also highly language-dependent, since the ordering and agreements depend on the fine-grained syntactic dependencies which can themselves be different in each language. Because more constructions and languages are supported for the second prototype, the number of rules at this level also increased significantly. However, the percentage of language-independent rules (48%) may seem surprisingly high; the reason is that research has been carried out in order to shift as many idiosyncratic properties as possible in the lexicons, which are already language-specific. As a result, more rules, especially some basic linearization rules, can be generic and language-independent.

DMorph-SMorph (retrieval of superficial forms): the coverage of morphological dictionaries has been increased in order to cover strictly the beAWARE requirements. The size of the morphological dictionaries is proportional to the size of the lexicons, shown in the 5th row of the table (Number of lexical units).

The last row of Table 2 presents a quantitative evaluation of the English generator on a large-scale dataset. We used a classical Penn Treebank split of a development set of 39,279 sentences and a test set of 2,399 sentences, which we processed with the WP3 semantic analysis component of beAWARE to serve as an input to the generator. The BLEU score (Papineni et al. 2002) of the generator currently reaches 35.53, compared to 31.78 at the beginning of the project, which represents an improvement of the score of almost 12%. After analyzing the details of the generations, it turns out the BLEU increase is almost exclusively due to the improvements of the Sem-DSynt module, and more particularly to the improvement of the coverage of the rules that build syntactic modifiers; these rules were unable to process some complex structures in the previous version. Unfortunately, large-coverage lexical resources are currently only available for English. For the other languages of beAWARE, the shortage of lexical resources prevents the evaluation of the rule-based system on a large scale at this point.

4.4.2 Results in the 2017 shared tasks

In 2017, three generation shared tasks took place:

- SemEval: http://alt.qcri.org/semeval2017/task9/
  This task consisted in generating English text from Abstract Meaning representations; 6 systems were evaluated.
- WebNLG:http://webnlg.loria.fr/pages/challenge.html
  In this challenge, the teams were asked to generate English texts starting from DBPedia triples consisting of different combinations of 373 distinct RDF properties; 10 systems were evaluated.
- E2E: http://www.macs.hw.ac.uk/InteractionLab/E2E/
  In this challenge, the teams were asked to generate English texts starting from RDF triples consisting of different combinations of 8 distinct properties; 20 systems were evaluated.

UPF took part to the three tasks with the FORGe generator, developed in the framework of the beAWARE project. FORGe is consistently the only rule-based generator in the different competitions. Other systems are statistical generators, based on Support Vector Machines, Neural Networks, or Statistical Machine Translation techniques for example. Statistical systems traditionally score much better than rule-based systems according to n-gram-based evaluation metrics, which are used in all competitions.

For the **SemEval** Task, FORGe ranked third according to both automatic and human evaluations. It obtained a very low BLEU score of 4.74, below the (low) average of 11.36. The task was extremely challenging, as the inputs came from annotated written text, in which the sentences can be very complex. The output quality of our generator was not very good. For the human evaluation, however, FORGe obtained scores above average (44.9 for the main metric, TrueSkill, based on simple rankings of outputs of different systems, for an average of 40.67). See (Mille et al., 2017).

The setup of the other 2 shared tasks are much more similar to the beAWARE configuration, as they both consisted on generating from Knowledge Base representations. For **WebNLG**, FORGe ranked third according to BLEU (38.65) and TER (0.55), and first according to METEOR (0.39), which is also a n-gram based metric, but which takes into account phenomena like synonymy. Our system ranked first according to all human evaluations, achieving rating very close to the ratings given to human-produced sentences. See (Mille and Dasiopoulou, 2017a), (Mille and Wanner, 2017).

Finally, for **E2E**, FORGe obtained a similar BLEU and METEOR scores to the ones at WebNLG (42.1 and 0.37 respectively), but ranked in the 17th position, in a task clearly designed for statistical systems. In the human evaluation, FORGe ranked 2nd according to quality and 4th according to naturalness (Mille and Dasiopoulou, 2017b). For the results of the human evaluation, the systems have been clustered, according to their results, into five groups; the performance of the systems within the same group cannot be differentiated. The quality of the output of our generator was the same as for WebNLG, and it still needs to be analysed why the human evaluation for naturalness is so poor in the case of this challenge.

4.4.3 Evaluation of the statistical (advanced) generator

We performed our first experiments with Spanish (using only the first section of the corpus, containing 3,513 sentences) and English (using the PTB-based annotation), evaluating steps 1 to 5 of the statistical generator. Since there is no existing generator that takes deep-syntactic structures as input, we used as baseline a very simple version of the rule-based graph-transduction generator.

The Spanish treebank, AnCora-UPF (Mille et al., 2013) has been divided into: (i) a development set of 219 sentences, with 3,437 tokens in the DSyntS treebank and 4,799 tokens in the SSyntS treebank (with an average of 21.91 words by sentence in SSynt); (ii) a training set of 3,036 sentences, with 57,665 tokens in the DSyntS treebank and 84,668 tokens in the SSyntS treebank (with an average of 27.89 words by sentence in SSynt); and a (iii) a held-out test for evaluation of 258 sentences, with 5,878 tokens in the DSyntS treebank and 8,731 tokens in the SSyntS treebank (with an average of 33.84 words by sentence in SSynt).

For the English treebank (Johansson and Nugues, 2007), and a UPF annotation of the deep-syntax), we used a classical split of (i) a training set of 39,279 sentences, with 724,828 tokens in the DSynt treebank and 958,167 tokens in the SSynt treebank (with an average of 24.39 words by sentence in SSynt); and (ii) a test set of 2,399 sentences, with 43,245 tokens in the DSynt treebank and 57,676 tokens in the SSynt treebank (with an average of 24.04 words by sentence in SSynt). In Table 3 and Table 4, we show the system performance on both treebanks in terms of standard n-gram-based metrics, BLEU (simple) and NIST (weighted), and by calculating the proportion of sentences that match exactly the gold standard sentence (Exact). For all the measures, a higher number corresponds to a better result.

Test Set	BLEU	NIST	Exact
surface gen.	0.91	15.26	56.02%
baseline deep gen.	0.69	13.71	12.38%
deep gen.	**0.77**	14.42	21.05%

Table 3: Overview of the results on the English test set excluding punctuation marks after the linearization.

Test Set	BLEU	NIST	Exact
surface gen.	0.762	12.08	15.89%
baseline deep gen.	0.515	10.60	2.33%
deep gen.	**0.542**	11.24	3.49%

Table 4: Overview of the results on the Spanish test set excluding punctuation marks after the linearization.

5 Towards advanced report generation methods

So far, the bulk of the efforts in WP5 have been devoted to developing the multilingual surface generation subcomponent of the MRG, while content selection and discourse structuring are addressed using a simple strategy that distinguishes between the source of the contents, i.e. visual or audio/text.. In the upcoming months the situation will be inverted, and strong emphasis will be placed in devising and implementing a strategy for the selection and ordering of contents received from the KBS.

The main line of research in text planning will be temporal narrative-aware selection of incidents for status updates, and a relevance-based selection of contents for summary reports. For the former we will look into more elaborate mechanisms to select from multiple incidents received in a series of requests belonging to the same location, so that each new request results in an update where repetitions are avoided and where contrasts between incidents in the temporal line of the disaster scenario are reflected. The main criteria for this strategy will be the specific times of each incidents, the history of incidents reported in previous updates, and the descriptions of objects impacted.

To produce summaries, the whole set of events belonging to a crisis will be relatively assessed to the overall scenario and ordered in a temporal narrative. This assessment will require that, in addition to the criteria enumerated for the situation updates, the summarization strategy will have to also consider the locations of each incident in order to report those locations - and time spans - where the most important incidents have occurred. In addition to these advanced strategies for report planning, the MRG will also be extended to support the additional contents made available to it via the KBS, e.g. priority and severity aspects, weather-related data, etc.

With respect to linguistic realization of the selected contents, the generation grammars and lexicons will be extended as needed for providing the coverage required by the final use case dialogues. The evaluation of the generation module on the beAWARE data will take place within the evaluation of the final prototype. Furthermore, by the end of the project, we will perform experiments on training advanced modules in Italian and Greek. These advanced modules are the results of research experiments that aim at ensuring a broader coverage for the generator, at the expense of quality; they could be integrated into the beAWARE pipeline in the eventuality that a coverage not allowed by the rules is needed.

6 Summary

Multilingual generation plays an important role in beAWARE by contributing to fulfil several user requirements where updated information coming from multiple media and sources regarding an emergency is required by the stakeholders to improve their awareness of the current situation or gain an overview of the whole crisis at the end of its course. It does so by producing textual reports in any of the languages supported in the project, namely English, Greek, Italian and Spanish. Producing reports involves planning the text from contents provided by the semantic integration and reasoning services and then rendering the contents as natural language text, these contents being in turn the results of the analysis of text, images, video and audio received via multiple channels.

This deliverable reports on the basic version of the MRG component in charge of producing multilingual reports for the first prototype of the beAWARE platform. Most of the advances reported belong to the multilingual realization step, while the previous planning remains the main focus of upcoming work. While coverage of the MRG as a whole is at this point largely restricted to the contents considered for the first prototype pilots, some of the linguistic resources have been developed to offer larger coverage, particularly for the generation of reports in English.

The advanced methods will invest a considerable amount of effort to offer a time-aware selection and ordering of contents that results in temporal narratives that omit redundancies and can be applied to the generation of both update reports and wrap-up summaries. Coverage of the linguistic resources will be extended for languages other than English.

7 References

Androutsopoulos, I., and D. Galanis (2013). “Generating Natural Language Descriptions from OWL Ontologies: the NaturalOWL System,” *J. Artif. Intell. Res.*, vol. 48, pp. 671–715.

Ballesteros, M., B. Bohnet, S. Mille, and L. Wanner (2015). “Data driven sentence generation with non-isomorphic trees”. *In Proceedings of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Denver, CO.

Ballesteros, M., B. Bohnet, S. Mille, and L. Wanner (2016). “Data-driven deep-syntactic dependency parsing,” *Natural Language Engineering*, vol. 22, no. 6, pp. 939–974.

Bohnet, B. and Wanner, L., (2010). “Open Source Graph Transducer Interpreter and Grammar Development Environment”. In *Proceedings of LREC*.

Bouayad-Agha, N., et al. (2012). “From ontology to NL: Generation of multilingual user-oriented environmental reports,” in *Lecture Notes in Computer Science* (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 7337 LNCS, pp. 216–221.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L. andStraňák, P. (2009). “The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages”. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task* (pp. 1-18). Association for Computational Linguistics.

Johansson, R., and Nugues, P. (2007). “Extended constituent-to-dependency conversion for English”, *Proceedings of 16th Nordic Conference of Computational Linguistics*, p. 105-112.

Kingsbury, P., and Palmer, M. (2002). “From Treebank to PropBank”, *Proceedings of the Third International Conference on Language Resources and Evaluation*, p. 1989-1993.

Mel'čuk, I. (1988). *Dependency syntax: Theory and practice*, State University of New York Press.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., and Grishman, R. (2004). “The NomBank Project: An Interim Report”, *Proceedings of the Workshop on Frontiers in Corpus Annotation, p. 24-31*.

Mille, S., A.Burga, and L. Wanner (2013). AnCora-UPF: A Multi-Level Annotation of Spanish.In Proceedings of the 2nd International Conference on Dependency Linguistics (DepLing).

Mille, S., Burga, A., Carlini, R.,Wanner, L. (2017): FORGe at SemEval-2017 Task 9: Deep sentence generation based on a sequence of graph transducers. In: Proceedings of SemEval '17. Association for Computational Linguistics, Vancouver.

Mille, S., Wanner, L. (2015). “Towards large-coverage detailed lexical resources for data-to-text generation”. In *Proceedings of the first workshop on data to text generation*, Edinburgh, Scotland, UK.

Mille, S. and S. Dasiopoulou. (2017a). FORGe at WebNLG 2017. Technical Report 17-09, Dept. of Engineering and Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain. Retrieved from **http://talc1.loria.fr/webnlg/stories/upf-forge_report.pdf**

Mille, S. and S. Dasiopoulou. (2017b). FORGe at E2E 2017. Technical Report 17-12, Dept. of Engineering and Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain. Retrieved from **http://www.macs.hw.ac.uk/InteractionLab/E2E/final_papers/E2E-FORGe.pdf.**

Mille, S. and L. Wanner. (2017). A demo of FORGe: the Pompeu Fabra Open Rule-based Generator. In Proceedings of the 10th International Conference on Natural Language Generation (INLG), Santiago de Compostela, Spain.

Mille, S., R. Carlini, A. Burga and L. Wanner. (2017). "FORGe at SemEval-2017 Task 9: Deep sentence generation based on a sequence of graph transducers". In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 920-923, Vancouver, Canada.

Nivre, J., de Marneffe, M.C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C.D., McDonald, R.T., Petrov, S., Pyysalo, S., Silveira, N. and Tsarfaty, R. (2016). "Universal Dependencies v1: A Multilingual Treebank Collection". In *Proceedings of LREC*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. (2002). "Bleu: a Method for Automatic Evaluation of Machine Translation". *Proceedings of the 2002 Conference of the Association of Computational Linguistics (ACL)*, p. 311-318.

Portet, F., et al. (2009). "Automatic generation of textual summaries from neonatal intensive care data," *Artif. Intell.*, vol. 173, no. 7–8, pp. 789–816.

Schuler, K. K. (2005). *VerbNet: A broad-coverage, comprehensive verb lexicon*, Ph.D. thesis, University of Pennsylvania.

Wanner, L., et al. (2015) "Ontology-centered environmental information delivery for personalized decision support," *Expert Syst. Appl.*, vol. 42, no. 12, pp. 5032–5046.

Yu, J., E. Reiter, J. Hunter, and C. Mellish (2007). "Choosing the content of textual summaries of large time-series data sets," *Nat. Lang. Eng.*, vol. 13, no. 1, pp. 25–49.