An SMS-based System Architecture (Logical Model) to Support Management of Information Exchange in Emergency Stuations. poLINT-112-SMS PROJECT

Zygmunt Vetulani, Jacek Marciniak, Paweł Konieczka and Justyna Walkowska

Adam Mickiewicz University, Department of Computer Linguistic and Artificial Intelligence,
ul. Umultowska 87, 61-714 Poznań, Poland, {vetulani, jacekmar, pawelk, ynka}@amu.edu.pl

Abstract: In the paper we present the architecture of the POLINT-112-SMS system to support information management in emergency situations. The system interprets the text input in form of SMS messages, understands and interprets information provided by the human user. It is supposed to assist a human in taking decisions. The main modules of the system presented here are the following: the SMS gate, the NLP Module (processing Polish), the Situation Analysis Module (SAM) and the Dialogue Maintenance Module (DMM).

Keywords: artificial intelligence, computer understanding systems, human-computer interaction, crisis management tools, incoming information processing, text understanding, information integration, contradiction solving, decision making

1. Introduction

In this paper we present the architecture of the POLINT-112-SMS system: a computer system supporting the gathering, processing and interpretation of information on events and situations reported by different informers in the text message form (in future version also in the spoken form). The following assumptions are made:

- the information may be sent by different informers, who do not cooperate with each other,
- the information concerns a specific situation or event,
- the information is sent by means of an SMS message in natural language (Polish) or in controlled natural language (a subset of Polish); the system

has passive language competence, i.e. it can understand and process information,

- the information inputted into the system may be untrustworthy, incomplete or imprecise; also, contradictory statements may occur,
- the system is supposed to process the information to the extent allowing it to display (visualize) the state of the situation to the user and to answer user questions,
- the information is sent to the system and (possibly) also to a human operator.
- The communication mode has been limited to text due to the following reasons:
- there is no speech-to-text technology (even for English language) advanced enough to make it possible for the system to understand speech in extremely noisy conditions or when speaker's pronunciation is sloppy, distorted or highly regional,
- the system is to be used in the specific environment where noise or security reasons make the usage of text mode the most appropriate; e.g. in football stadiums during important events where the use of speech is not advisable because of both technical (noise) and logistic (safety of the informer) reasons.

A typical example of the system's application under the above assumptions is supporting the process of crisis situation management (a situation that poses danger to public safety), where decisions have to be made based on a large amount of information coming from different sources. In particular, this applies to mass events (sports games, big artistic events) and natural disasters. The common feature for this kind of situations is their variability and dynamics. This is why they need to be monitored by several observers who will report from various locations in different moments. As a typical case we selected high-stake football games (such as UEFA European Football Championship).

Compared to traditional methods of event reporting (telephone communication between the informer and the emergency center), a significant quality improvement is expected thanks to the computer-assisted integration of information coming from distributed informers, the on-line verification of the coherence of data, and the high credibility of the information sources. This goal can be achieved by using POLINT-112-SMS software to understand text and to process messages.

The practical goal of the work described in this article is to create a tool assisting a human in crisis management. By crisis management we mean undertaking preventive or repair actions by a human or a team, from now on referred to as Crisis Management Centre (CMC). We assume that CMC uses data acquired from informers, who can be ether professionally-trained experts or outside persons, who input information in a mode similar to 112 emergency telephone service, but using SMS messages instead of voice as the communication medium.

In traditional crisis management models, information is received and interpreted by professionally trained human operators. The fundamental problem here is the problem of communication bottlenecks when large amount of information tries to reach the CMC at the same time. When this problem is addressed by involving more operators, information may be lost, may remain unconnected (when two operators describe different aspects of the same situation) or even contradictory. The automation (or partial automation) of information gathering and processing may help solve this problem. Within this project we assume that the information is sent directly to the computer system that processes it for CMC use.

System's main functions are:

a) to collect information sent by the informer (SMS message),

b) to process (understand) the information to construct a coherent model of the situation/event in real-time,

c) to integrate data from different sources,

d) to perform some of the operator's duties when all the operators are busy.

Figure 1 presents a possible application context of the POLINT-112-SMS. The prototype being developed now will be tested in context of public order protection during a football match attended by a large number of fans.

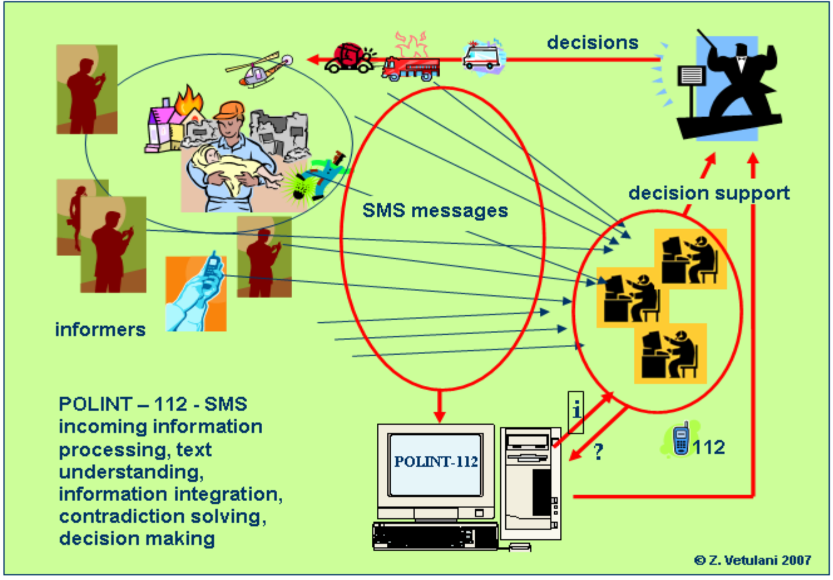


Figure 1. Application context of the POLINT-112-SMS system

2. LOGICAL MODEL

The logical model of the POLINT-112-SMS system, constituting the basis for the implementation in progress, has been developed with participation of experts with practical experience in the field of sport events security. As empirical background we considered a corpus of SMS messages collected in experimental setting. The messages were gathered in controlled conditions (experiments with participation of PPBW (Polish Platform for Homeland Security) to construct a linguistic model of SMS communication in Polish language in some precisely defined types of situations (cf. also (Fairon & Paumier 2006) and (Walkowska 2008)).

2.1. Assumptions about the logical model

The complexity of the processes that the system is expected to support has led to distinguishing 3 Entity Sets, that will be used to store data and reason about the events. These are Reports, Events and Situations.

Report – an entity designed to store data gathered during interaction with one specific informer. It is kept in the system for a determined period of time and moved to an archive when it expires. A Report is always assigned to at least one Event. If a Report cannot be assigned to any Event already existing in the system, then a new Event is created. After the Report is created, the system must decide about what Event(s) (a fight, a fire) it should be tied to. The type of the Event determines the mode of possible system-user dialogue to feed the system with required information. The system formulates questions to obtain as much information on Events tied to the Report as possible. E.g. or a Fight Event this may include questions about location, number of participants, danger for external people, etc.

Event – an entity representing a real-life situation (accident, crime, fight) reported by at least one informer. One Event has at least one Report assigned to it, which stores pieces of information. One Event organizes data from possibly various Reports. Each Event may contain:

- the most up-to-date (according to the system) information about the event,
- reliability assessment for each partial piece of information,
- information about contradictory elements of Reports,
- information about false data discovered in Reports.

The system tries to gain the maximal amount of information on the Event. It will query the informer when the introduced information is untrustworthy, contradictory or false. If the course of the dialogue (especially in the initial phase) does not make it possible to assess whether the inputted data concerns an Event already present in the system, an auxiliary unification question (e.g. about the location of the Event) may be asked.

Situation – an entity representing an important situation concerning many people (e.g. public order disturbance, riots, accidents with a significant number of victims). Such a situation is identified and the corresponding Situation entity is generated by the system automatically (using Situation Templates). This may also be done by the Analyst who has access to the events controlled by the system. At least one Event is assigned to each Situation. The Situation contains:

- information about Events tied to the Situation,
- information about the completeness of data in the Situation,
- information about how important the Events in the Situation are.

While accepting a Report concerning an Event that is tied to the given Situation, the system may generate further questions to the informer about different Events in connection to this Situation in order to complete its understanding.

2.2. Components

The system's architecture is presented in Figure 2. The components' tasks are as follows:

SMS Gate is a module allowing SMS communication with the informer by means of SMS messages. It is formed of two submodules, one of which is responsible for sending messages, the other one for receiving them. The SMS Gate communicates directly with the NLP Module.

The NLP Module is the main module responsible for processing text input to the System. In the final version of the system, the module will be responsible for question generation, but at this point the question generation functions are performed by DMM. The NLP Module communicates directly with the SMS Gate and with DMM.

Dialogue Maintenance Module (DMM) is responsible for dialogue with the informer. It takes into account the data controlled by the Situation Analysis Module. Thanks to the DMM, the NLP Module focuses on transforming single sentences into data structures without storing and processing these structures. DMM communicates directly with the NLP and SAM modules.

Situation Analysis Module (SAM) is responsible for reasoning. It acts as the „brain" of the system. It controls a number of subordinate modules, presented in points 5-12 (at this point some of them are integrated with SAM). SAM reasons about the structures without directly communicating with the informers. SAM communicates directly with DMM.

Knowledge About the World Module stores general knowledge and is used as the system's knowledge base. It may contain knowledge such as firemen reaction procedures, city maps in the GIS format and other information.

PolNet Module. PolNet is a WordNet-type ontology. Apart from the basic relations of hyponymy / hiperonymy it also contains relations that facilitate reasoning.

Reports Module stores information obtained from users in the form of Reports.

Events Module stores information about Events. Each Event is tied to at least one Report. The information stored in this module can be directly accessed by the Analyst

Event Recognition Module is responsible for creating new Events in the Events Module.

Situations Module stores information about Situations. Each Situation is tied to at least one Event. The information stored in this module can be directly accessed by the Analyst.

Situation Recognition Module is responsible for creating new Situations in the Situations Module..

Reaction Module's task is to inform the Dispatcher at the Crisis Management Centre that an action should be taken (e.g. sending an ambulance).

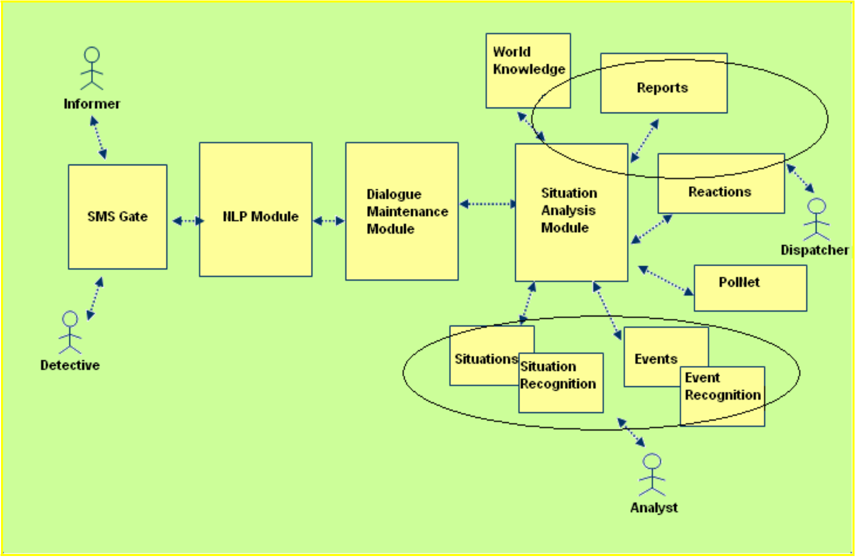


Figure 2. The logical model for the Polint-112-SMS system

3. Components implementation

3.1. System input – the SMS Gate

The SMS Gate is used for sending and receiving SMS messages. Currently it consists of a physical device (SMS terminal) and an application enabling communication with the device. The component's main tasks are: informer identification (by their phone No.), merging multipartite messages (i.e. messages too long to fit in 160 characters) into one message, detection and standardization of the encoding, and detection of the beginning and the end of a single dialogue session with the user. The informer's interaction with the system comes down to exchanging SMS messages (new information, questions, answers) .

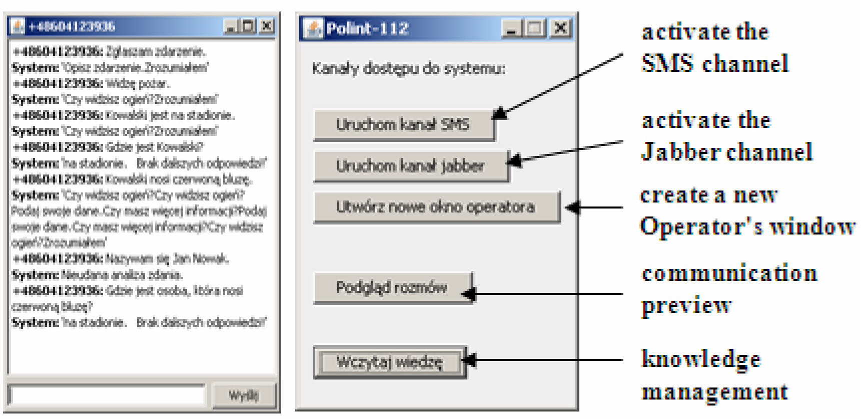
Technical information about the dialogue is kept in a dedicated structure called session. It stores information about the informer's phone number, the beginning of the dialogue, the time of the last message exchanged between the parties, message character encoding and the state of the session. The session becomes inactive after a preset time boundary has been reached. The time limit has been proposed on account of some specific features of SMS communication: determining the end of a dialogue session is not as straightforward as it is in the case of a telephone conversation. The system needs a way of determining whether it is worth to wait for an informer's answer, or whether it should process the information it has already gathered.

Determining the boundaries of dialogue sessions is crucial for solving anaphoric references. One solution would be to introduce special types of messages, in which the informers would determine the start and end of a dialogue session (but this solution would impose unnecessary constraints to the use of natural language and therefore would be in contradiction with the principle of unconstrained NL access). The session structure is also used by consecutive modules in the system, especially by the Dialogue Maintenance Module.

The prototype implementation of the SMS Gate is a Java application that communicates with PROLOG modules by SWI-PROLOG's API. This application is capable of accepting messages sent to the SMS terminal and carrying their content to further modules and of sending the module's output to the user. An additional window can be opened to monitor the informer's communication with the system.

Within the prototype implementation of the system it is also possible to input messages from the keyboard. This type of the communication may be considered as a regular communications mode and it can be used in the target version for facilitating interaction with the system by the operator (e.g. with the role of system analyst).

In Figure 3 below we show a screenshot of a sample session.



+48604123936: Zgłaszam zdarzenie.
System: 'Opisz zdarzenie.Zrozumiałem'
+48604123936: Widzę pożar.
System: 'Czy widzisz ogień?Zrozumiałem'
+48604123936: Kowalski jest na stadionie.
System: 'Czy widzisz ogień?Zrozumiałem'
+48604123936: Gdzie jest Kowalski?
System: 'na stadionie.   Brak dalszych odpowiedzi!'
+48604123936: Kowalski nosi czerwoną bluzę.
System: 'Czy widzisz ogień?Czy widzisz ogień?Podaj swoje dane.Czy masz więcej informacji?Podaj swoje dane.Czy masz więcej informacji?Czy widzisz ogień?Zrozumiałem'
+48604123936: Nazywam się Jan Nowak.
System: Nieudana analiza zdania.
+48604123936: Gdzie jest osoba, która nosi czerwoną bluzę?
System: 'na stadionie.   Brak dalszych odpowiedzi!'

Figure 3. A sample dialogue session in the SMS Gate's window

3.2. *The NLP Module*

The NLP Module is based on the POLINT system, initially developed as a text understanding system which answers the user questions (both questions and answers in Polish). Several prototypes of the POLINT system were implemented during 1990ies and in the past couple of years initially in Arity-PROLOG. Using these prototypes the user may ask questions about facts collected in a PROLOG data base. The methodological and linguistic foundations of POLINT have been published in (Vetulani 1989) (see also (Vetulani & Marciniak 2000) and (Vetulani 2004)).

For the needs of this project, POLINT was re-implemented in the SWI-PROLOG and adapted to new tasks, in particular gathering information from the user.

POLINT (Vetulani 1997) was a language understanding system including several modules:

- dictionary (lexicon-grammar),
- preanalysis module for parsing heuristics,
- interpretable grammar rule set (PROLOG clauses),
- semantic module,
- knowledge base (in form of PROLOG facts and clauses).

POLINT is a real time system due to appropriate use of heuristics which substantially reduce non-determinism of the top-down parsing mechanism (inherited from PROLOG). Heuristics take into account words in the input string, extract from the system dictionary the syntactic information contained in the lexical entries and calculate heuristics. These operations are executed in linear time. Heuristics permit to generate syntactic hypothesis about the input sentences and therefore substantially reduce the search (otherwise non-deterministic) (eg. (Vetulani 1997, 2004)).

The role of the NLP Module with respect to other elements of the POLINT-112-SMS consists in partial understanding of the input text. The NLP Module is responsible for processing isolated sentences (or sentence equivalents). In particular, resolution of problems which remain unsolvable without considering larger context and/or some extra linguistic knowledge will require involvement of the other system modules, i.e. SAM and DMM (anaphora, identification of event actors etc.).

At the present stage we were able to test the system's basic understanding functionalities. The following is a sample dialogue performed by the system and the user. The user was allowed to supply information and to ask questions (see Figure 4). In the dialogue presented in Figure 4, the involvement of the SAM/DMM functionalities was very restricted.

USER: Kowal nosi bluzę i czarne spodnie. Kowal i Wolski są na murawie. Kowal i Wolski należą do bojówki. *(Kowal is wearing a blouse and black trousers. Kowal and Wolski are on the football field. Kowal and Wolski are members of a hooligan squad)*
SYSTEM: Zrozumiałem'*(Understood)*
USER: Osoba, która nosi czarne spodnie, jest niebezpieczna. Kowal i Wolski atakują policjantów kamieniami. *(The person who wears black trousers is dangerous. Kowal and Wolski are attacking policemen with stones)*
SYSTEM: Zrozumiałem *(Understood)*
USER: Kto atakuje policjantów? *(Who is attacking the policemen?)*
SYSTEM: Kowal. Wolski. Brak dalszych odpowiedzi! *(Kowal. Wolski. No more answers!)*
USER: Kto jest niebezpieczny? *(Who is dangerous?)*
SYSTEM: Kowal. Wolski. Brak dalszych odpowiedzi! *(Kowal. Wolski. No more answers!)*
USER: Czy istnieje niebezpieczna osoba, która nie została złapana? *(Is there a dangerous person who has not been captured?)*
System: 'Tak, kilka, w szczególności istnieje: Kowal !' *(Yes, a few persons, in particular Kowal!)*

Figure 4. A sample dialogue session

3.2. *Dialogue Maintenance Module (DMM)*

The DMM's main tasks are as follows:

- DMM receives and processes data structures.
- If the data structure created from a sentence by the NLP Module lacks an important argument, then DMM consults the informer about its value be-

fore passing the structure to SAM. The importance of arguments is expressed by means of their priorities.

- If SAM needs to confirm or to fill a slot value in a structure (e.g. the color of clothes of a tracked person – slots correspond loosely to predicate arguments), then the module marks the slot and sends the structure to DMM. DMM chooses an informer that may have information on the subject based on the user dialogue history and asks the question.
- For any incoming data structure, DMM needs to decide whether the structure contains completely new information, or is a continuation of previous information (then structures should be merged), or is an answer to one of the previously asked questions.
- DMM tries to maintain and expand the user model. There are different types of users – information from registered police informers is to be treated with more trust than information from an anonymous user. User type influences DMM's communication mode. Some types of informers cannot communicate with the system too often, because their function cannot be revealed to people surrounding them, so questions to them should be grouped and sent rarely. If a user takes too much time answering questions, DMM will most likely choose somebody (at similar position) else to ask for urgently needed information.
- One of DMM's most important functions is to solve anaphoric references. If the NLP Module discovers a reference of this kind (as in "He hit her."), it marks the corresponding slots. When DMM receives such structure, it tries to find the referenced value in the recent user dialogue history.
- There is a number of structures that DMM passes to SAM with almost no processing. These include structures used to indicate that the informer has asked a question or asked for notification when a described situation occurs (e.g. "Please inform me when Piotr Kowalski enters sector 5."). For such structures, DMM's task is only to remember which user asked for information, in order to sent them the answer/notification later.
- As the NLP Module is limited to understanding (questions, affirmative sentences and orders), DMM is charged with question generation. It keeps partially predefined questions for values of each slot in each structure, also in nested structures. It chooses the question form based on the slots that are already filled (e.g. a question for a person's first name can be formulated using their surname or nickname) and on the questions that have already been asked. DMM tries not to repeat question forms or questions. If a informer does not answer a question or says that he does not know the answer, DMM remembers the fact and does not try to ask again.

DMM is being implemented in PROLOG. It operates on data structures that also play the role of transport structures between the NLP Module and DMM, and between DMM and SAM. The NLP Module fills the structures with as much information as it can extract from single sentences. As stated in the previous section, some of DMM's most important tasks are merging structures received from the NLP Module, solving anaphoric references and asking (i.e. choosing and generating) questions about missing pieces of information that are considered crucial.

Structure merging is based on PROLOG list unification. After receiving a structure generated from a sentence sent by an informer that had communicated with the system a short time before, DMM tries to merge the structures. If the same kind of structure has been received (e.g. one describing a person), but the information inside is different (or partially different), DMM checks for conflicts. If there is a conflict, meaning that a value in a particular slot differs among the structures, then DMM either considers the structures separate (e.g. when different surnames are given) or asks for clarification (e.g. when different nicknames are given, it can ask if they both refer to the same person). When there are no conflicts, it is assumed that the information carried by the structures is complementary, and the structures will be merged. After having merged the structures, or adding a new structure if unification was not possible, DMM checks for missing information. Every slot in the structures (representing every possible piece of information that the system is capable of processing) is given a priority value at the start of the system. If a slot with priority above some predefined threshold remains not filled, then DMM asks for its value before sending the structure to SAM (it will send the structure anyway if the informer does not know the answer or if it does not respond within a given period of time). Thresholds may be different depending on types of informers.

DMM keeps a list of all structures sent by the NLP Module and of all questions it has asked. Because of this, it is able to assess that an incoming structure is the answer to one of its questions in a manner similar to structure merging described in the previous paragraph. Keeping a history of user input allows DMM to choose the best informer to ask when SAM forces a question by looking through structures sent by informers and finding one containing similar information (e.g. a structure in which an informer described an event close to a place about which SAM want to receive more information).

The situation is very similar when it comes to anaphoric references. If the first structure sent by the NLP Module, generated based on the informer's input, describes an event in which two people participate („Kowalski is throwing stones at Nowak.") and the second structure describes a person, but the person is not named („He is very aggressive."), the NLP Module will mark the corresponding slot (representing the person's data) as anaphoric. DMM will then test the latest structures sent by the same informer, looking for structures representing persons that might match the event. If in the last event more than two people are present (as is the case in the example), it will choose the person that was more active. Coming back to the example: DMM will decide that „he" in „He is very aggressive" is Kowalski. (Of course, if Kowalski is throwing stones at Nowak, the Situation Analysis Module will mark him as aggressive anyway.)

The DMM only reasons within one dialogue session. It does not try to merge structures sent by different informers on its own. It is the Situation Analysis Module that is responsible for combining information sent by different informers. When SAM forces a question upon the DMM, the DMM starts a new dialogue session with the informer.

3.3. Situation Analysis Module (SAM)

The Situation Analysis Module is responsible for collecting knowledge based on the information from the informers and for reasoning with the knowledge in order to deduce new facts. Its main tasks are: management of information about individuals, linking new information about entities with information stored in the knowledge base, recognizing Event types (using built-in context-related Event templates) and managing knowledge about Events and Situations (linking participants of Events and Event type structures, detecting dangerous objects, etc.).

There are three categories of SAM data structures:

a) Structures describing entities and relations between them.

Two types of entities have been distinguished: animate (individuals and groups) and inanimate (objects, artefacts). Special structures have been added to represent the relation of possessing objects by individuals and groups. Examples of entity structures are shown below:

- PERSON(id, name[], surname[], alias[], function, sex, appearance(heightCm, hair_colour, hair_length, eye_colour, skin_colour, has_moustache, hair_beard, is_dressed, identifying_articles[], clothes(article, pattern)[]), mood[], physical_state[]),
- GROUP_MEMBER(group_type, person_id, group_id, position_in_group)
- ARTICLE(id, idPolNet, size, colour, is_dangerous),

b) Structures describing Reports, Events and Situations

The most important structures from the point of view of SAM are those representing Events. Such structures contain current information about real-life, potentially dangerous situations reported by the informers. Examples of Event structures are shown below:

- EVT_BATTERY(id, aggressors[], victims[], articles[]),
- EVT_UNREST(id, participants[]),
- EVT_DESTROYING(id, participants[], destroyed_place, articles[]),
- EVT_FIGHT(id, participants[], articles[]).

c) Structures describing referential points in space and space-time relations between entities, Events and places.

The location of entities and Events is represented in the knowledge base by means of special structures called landmarks. They describe the entities' and Events' location in the space of referential points. Referring to them by their names, the informer inputs data about their location, specifying temporal and spatial relations between entities ("individual A is standing next to individual B"), between entities and Events ("individual A is involved in a fight in sector X"), between Events ("fans of opposite teams met, and then they started fighting"), between landmarks and entities ("the suspect is in the guests sector), between landmarks and Events ("riots in the guest sector"), or between different landmarks (sector A is to the right of sector B). Relations of this type can be static ("to the north of X") or dynamic ("right behind X's back"). Spatial relations in the Situation Analysis Module are translated into directional matrices, on which calculations are carried that determine new space-time relations.

Examples of space-time structures are shown below:

- LANDMARK_SECTOR(place(id, idPolNet, name, place_state(is_peace, is_smoke, is_fire, people_inside)), team, capacity),
- TIMESPACE_RELATION(object1_id, object2_id, matrixes[], space_distance, time_distance, orientation).

After consultation with experts a document describing business processes was prepared. The processes were used to create an experiment during which participants sent SMS messages reporting observed course of events. Analysis of the collected corpora allowed for creation of predefined Event templates for situations in the chosen context. Situation details important from the point of view of knowledge processing have been identified. The type of the Event influences the way in which the dialogue is managed and determines the set of questions asked by the system. If during a dialogue session it turns out that the informer is describing a type of Event different from what was assumed at the beginning, the dialogue mode changes.

When new information is inputted into the system, SAM checks if it is possible to tie the incoming information (forming a Report) to an Event or Situation already present in the knowledge base. If SAM suspects that such relation occurs, it asks questions (i.e. forces the Dialogue Maintenance Module to ask questions) that may confirm it. After confirming the relation SAM will try to gather missing information about the Event/Situation.

4. Conclusions

A system based on the presented architecture can satisfy the needs of a number of user categories. When used in a crisis situation or a potential crisis situation (e.g. large scale football event or another type of mass event) the system might have the following users:

The Informer, is a person (not necessarily professional) which reports on an accident, a crime, an incident, etc. The Informer might be in a state of vexation, stress or fear. Hence, the data obtained from them may be incomplete or imprecise, and the mode of conveying the information may be chaotic or clumsy.

The Detective is a employee of emergency services (police, fire service) working in the field, supposed to report an accident, a crime, an incident or a significant change in the monitored situation, etc. It is assumed that he/she has been trained to work under crisis conditions.

The Analyst works for emergency services trained to recognize critical situations.

The Dispatcher is an emergency services employee responsible for making decisions about the actions that should be taken.

Identification of the above user categories introduces new quality into the technologies of crisis situation management. Systems with text based communication competence (as it is the case of POLINT-112-SMS) apt to collect and process knowledge, open new possibilities to obtain information about events from a large number of informers in situations that call for quick decisions.

ACKNOWLEDGEMENTS

The research presented in this paper was partially covered by the on-going Polish Government research grant R00 028 02 "Text processing technologies for Polish in application for public security purposes" (2006-2009) within the Polish Platform for Homeland Security.

References

Fairon C., Paumier S. (2006): A translated corpus of 30,000 French SMS; in: Proceedings of LREC 2006. Genova.

Vetulani Z. (1989): Linguistic problems in the theory of man-machine communication in natural language. A study of consultative question answering dialogues. Empirical approach. Brockmeyer, Bochum.

Vetulani Z. (1997): A system for Computer Understanding of Texts, in: R.Murawski, J. Pogonowski (eds.), Euphony and Logos (Poznań Studies in the Philosophy of the Sciences and the Humanities, vol. 57) Rodopi, Amsterdam-Atlanta, 387-416.

Vetulani Z., Marciniak J. (2000): Corpus Based Methodology in the Study and Design of Systems with Emulated Linguistic Competence; in: Dimitris N. Christodoulakis (ed.), Natural Language Processing - NLP 2000, Lecture Notes in AI, no 1835, Springer, 346-357.

Vetulani Z. (2004): Komunikacja człowieka z maszyną. Komputerowe modelowanie kompetencji językowej, Akademicka Oficyna Wydawnicza EXIT, Warszawa.

Walkowska J. (2008): A corpus of real-life and experimentally collected Polish SMS messages (manuscript).