Modeling Administrative Procedures to Improve Information to the Public*

José Félix Muñoz[1, 2], Carlos Bobed[2], Francisco Serón[2]

[1] Aragonese Foundation for Research & Development (ARAID)
[2] University of Zaragoza, Walqa Technological Park, E-22197 Huesca, Spain
{jfm, cbobed, seron}@unizar.es

Abstract. This paper deals with the theoretical foundations, standards used and experience gained in the design, development and deployment of a human-shaped intermediary agent on the website of a public administration, whose purpose is to inform citizens about files that are being processed there. This agent exploits different knowledge sources under a unified model, which allows it to integrate factual knowledge about what has already been done, provided by the workflow tool, and theoretical knowledge about the procedural steps that still need to be taken before finalizing it.

The knowledge model the agent uses is implemented by using XPDL. The benefits of using this model language along with the defined equivalences are explained. Finally, we indicate the main problems encountered in the deployment of the agent on the websites of two Spanish councils, emphasizing the lack of availability of knowledge in the system that would be important in order to give information to the public, especially regarding access permissions.

Keywords: e-government, open government, administrative procedures, intelligent agents, semantic web, ontologies, XPDL.

1 Introduction

1.1 Background

In recent years, Spanish public administrations have made a major effort to systemize and model their procedures. One very important factor was the 11/2007 Act on

* Work partially financed by the spanish government and co-financed by the ERDF (European Regional Development Fund) through the projects "Intelligent Agent for Assisting Users of the Electronic Municipal Administration" (AIAM), TSI-020100-2010-455, of the Avanza Plan; "Legal Guarantees Applicable to Electronic Documents", DER2009-09744; and "Replikants: Towards a New Generation of Humanalike Agents", TIN2011-24660, of the Spanish National RDI Plan.

electronic access to public services for citizens, which obliges public administrations to use the Internet in its communications with the public (a concept that includes both individuals and businesses), that is, in their front-office. One of the elements that all administrations must prepare in accordance with the 11/2007 Act is the "catalogue of procedures". Here, it is essential to make a distinction between two meanings of the word "procedures": on one hand, it is used to refer to each of the administrative actions that members of the public may initiate in an administration in order to achieve a given objective (renovating their home or appealing against a traffic penalty for example), and, on the other hand, it is used to refer to the process that is followed in the administration in order to reach a decision on these actions. The law refers to the expression "catalogue of procedures" in terms of the first meaning (for which we shall use the word "procedures" in this article). Consequently, the "catalogue of procedures" does not include procedural knowledge (about the processes), but is a document that contains a list of all the procedures that citizens can follow before a certain public administration.

As well as providing the means so that members of the public can communicate with them via the Internet, many administrations are also computerizing their back-offices through the implementation of workflow tools for the processing of procedures (the term "processing platform" is generally used to refer to them), and the gradual replacement of paper documents by their electronic equivalents. When carrying out this work, it should be remembered that the 11/2007 Act specifies that the automation of procedures must be preceded by analysis regarding their functional redesign and simplification. Regarding the tools to be used for this purpose, one is BPMN [4] (Business Process Modeling Notation), a process modeling language that was developed for the modeling and optimization of business processes in private companies, but it is also suitable for modeling administrative procedures [15]. Some public administrations in Spain have already started to use the BPMN language [3].

1.2 Three Information Sources

The information that is currently provided to the public on the websites of public administrations is mainly that relating to the "catalogue of procedures". This is public access information and basically includes the purpose of each procedure, the available channels —in person, via the Internet (with or without an electronic certificate), by post or by telephone— and the basic requirements for their initiation (documentation to be provided, for example). The second source of information is the one regarding files being processed (not finished). Access is restricted to interested parties and includes the procedural steps already taken and access to documents associated with them.

Finally, there is a third type of information available in the administration's information system, which is that regarding the processes and which is currently not included in the information that public administrations provide to the public on their websites. Therefore, at the end of 2010 our group, together with the company iASoft/Oesia, created the project "Intelligent Agent for Assisting Users of the Electronic Municipal Administration" (AIAM, for its acronym in Spanish), with the

aim to develop an intelligent agent [16] that will serve as an interface for communication with members of the public and be capable of integrating these three information sources. This paper focuses on the model that allows the agent to integrate and exploit these three sources of information.

The rest of the paper is structured as follows. In Section 2, the nature of the different knowledge sources that the agent has to deal with is analyzed. Then, in Section 3, the actual model that the agent handles and the different design decisions are explained. In Section 4, the scenarios where the agent has been deployed are presented, as well as some important experiences gained from the deployment. Finally, in Section 5, conclusions and future work are presented.

2 The Knowledge Used by the Agent

2.1 Substantive vs. Procedural Law

In the science of the law, a distinction is made between two types of law: *substantive law* and *procedural law*. In brief, we can say that substantive law establishes rights and that procedural law deals with the way in which actions are used to demand their effective enforcement before public bodies and with the course that established procedures follow in order to arrive at a decision. This distinction is also valid from the point of view of knowledge representation, as each one has very different characteristics. Substantive law refers to a large extent to elements of reality, whether moral (values) or physical. Therefore, ontologies [7] are usually used to model it, as they enable us to represent this reality by categorizing concepts and to establish relationships between them.

However, in procedural law both the agents responsible for executing the administrative actions and the procedures they use are creations of the law itself, that is, they do not have their own existence that is prior to and independent of the law. In this respect, procedures are abstract entities created by laws, which means that the inherent ambiguity of concepts that are used for their representation (or modeling) may be much less than the ambiguity that concepts used in the representation of substantive law. Although other types of knowledge might appear in the creation of procedures (for example in [8] distinctions are made between business activities, process workflow, social structure, knowledge created and used, and technology), the main object to be modeled regarding procedural knowledge is the process workflow itself. As a consequence of the above, instead of using an ontology to model the different objects that a procedure comprises, a more specialized modeling formalism should be considered in order to properly capture the semantics of the flow without duplicating modeling efforts.

Finally, a reflection about the scope of the model presented in this paper regarding access permissions has to be posed. The agent allows access to administrative files being processed and, in accordance with the law, each file can only be accessed by citizens who are interested parties. These include firstly the promoter, who is the person that initiates the action, but also include any people that have rights that could

be affected by the proceedings. Therefore, while the allocation of access permissions to civil servants may be carried out based on formal knowledge, as permissions derive from their role in the entity [18], in the case of members of the public it is necessary to study their rights, both in order to permit access (a legitimate interest in the proceedings) and to deny it (e. g. a conflict with the right to privacy of other people). Consequently, the modeling of knowledge regarding access permissions would introduce all the inherent complexity of substantive law into the system and it would be necessary to develop a specialized agent, which does not fall within the scope of our project, but it could be considered as future work.

2.2 Theoretical and Factual Knowledge

Apart from the distinction made in the previous subsection, there are two types of knowledge that the agent uses: theoretical and factual. The first deals with procedures and has its origins in the law —legislation (laws) and subordinate legislation (bylaws) [9]. Within this type, as we have seen, it is necessary to differentiate between:

- Knowledge that describes the different administrative actions that the public can take. In Spain, normally, the information that is given to the public about this type of knowledge has been previously compiled in the "catalogue of procedures", has been entered into a database and is provided to the public in text form via dynamic web pages.
- Knowledge that defines the processes that are followed to resolve the administrative actions. This one is being computerized to a much lesser degree because the work for the modeling of the processes was not included within the obligations imposed by the 11/2007 Act, and therefore depends on the policy of each administration, which carry them out as necessary for administrative simplification or for the automation of the back-office. Another reason is that, even when the systematization work is carried out, the use of specific modeling tools is not spread and in many cases the procedural flows are described only in text form.

As a matter of fact, there is a direct dependence between the theoretical and factual types of knowledge, as theoretical knowledge acts as a metamodel for factual knowledge. The theoretical knowledge provides the different processes, i. e. instances of *process*, that represent the different classes of *files* that can be used when dealing with factual knowledge. That is, the instances at the theoretical level are the classes at the factual one. In the functioning of the agent, when members of the public request information, the agent firstly finds their *files*. Then, for each *file*, an entity is selected from the model above (*processes*) that matches it —that is, the entity that is being used for its processing. After that, an instance from the procedural entity is created for each *file* and the part of the instance that relates to the part of the process already carried out is completed using the data from the "processing platform".

In the following subsection, we will present the details of the modeling of processes that supports the agent below: the conceptual model, the modeling language used, how to map the elements of procedures onto it and the extension possibilities that the model has.

3 The Modeling of Procedures

3.1 Modeling of *Procedures* and *Files*

As we have seen, the agent is able to work using procedural knowledge on two levels: theoretical and factual. Theoretical knowledge allows the agent to be aware of the flow that a certain file has to follow and factual knowledge is the information about the real progress of proceedings. Figure 1 shows the model of theoretical level types in detail, together with the relationship between the two main types from both levels. The theoretical model acts as a metamodel for the factual model, as instances regarding different procedures are in turn instantiated for each file. The factual model is not depicted completely in the figure because we focus on the classes that have deep conceptual differences between knowledge levels. The rest of the model is quite similar to the theoretical one.

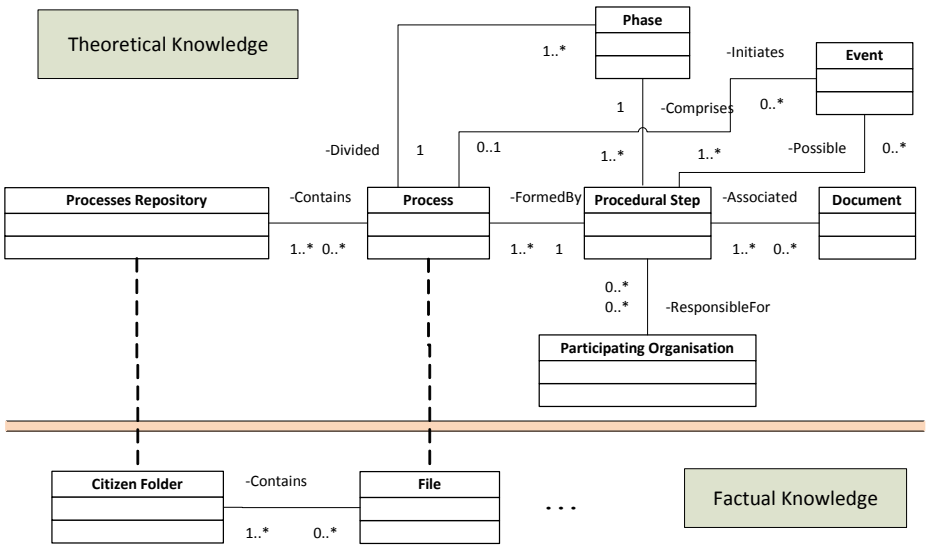

Fig. 1. Relationships between the different elements of the models

The model above (theoretical knowledge) represents the different processes that may occur and is made up of the following elements:

- *Process*. This is the workflow model of a process used in public administrations. These constitute the procedural knowledge and are stored in the *processes repository*. These are divided into *phases*, and are composed of *procedural steps*. For each *file* opened in the Administration a *process* is instantiated, depending on the procedure.
- *Procedural step*. This is each of the steps that comprise the *process*. They are grouped together into *phases*, which give a logical unity to a group of *procedural steps*.

- *Event*. This is any external fact that affects the *process*. These are not considered in the execution flow, but it is specified in the theoretical model which *events* may occur in a certain *procedural step*. These may initiate other *processes* and are always associated with the *procedural step* in which they occurred.
- *Document.* This is the type(s) of documents that are required in each *procedural step*.
- *Participating organization*. This is the body in charge of carrying out a certain *procedural step*.

This theoretical model is instantiated for each file, leading to the factual knowledge model:

- *File*. This is an instance regarding a procedure that is completed using the factual information from a *file* being processed. These are grouped together in the *citizen folder.*
- *Citizen folder*. This is the group of all the *files* being processed in which a user is interested part. To the extent that this is a repository for instances regarding *processes* (*files*), it can be considered as the counterpart to the *processes repository*, although it should be remembered that the criteria used for the groups are entirely different, in one case *processes*' families are grouped, in the other the *files* related to a particular citizen are grouped.

The other entities (*events*, *documents* and *procedural steps*) become instances of the classes defined in the theoretical model. In the factual model, instances represent real objects with real occurrences, while in the theoretical model the classes to which they could belong are considered.

3.2 Why XPDL? An Overview

In general, when we talk about modeling knowledge the use of ontologies springs to mind. However, the characteristics of the information that the agent has to process (procedural knowledge) imply that the most suitable option would be the adoption of modeling languages whose specific purpose will be the processes. This way, efforts to define the execution semantics of the processes are avoided, as they are given to us by the language used in the adopted model itself.

The modeling language chosen was XPDL [17], a process modeling language that was specifically designed to represent BPMN (Business Process Modeling Notation) models. XPDL, standardized by the Workflow Management Coalition (WfMC), allows us to specify the recitals of a process and provides us with three important characteristics in accordance with the specifications established for the AIAM project:

- It is an open standard.
- It allows us to model the processes that the agent has to interpret.
- It offers flexibility for the extension of specific added information from the project.

Figure 2 shows a view of the main elements that XPDL provides. Only the most important ones are mentioned. For more details please see [17]:

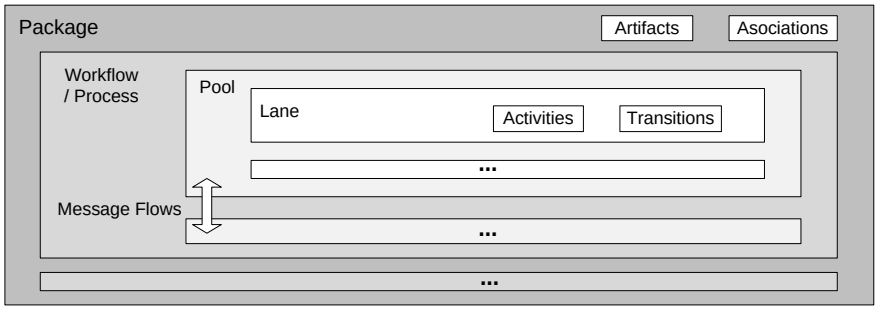

Fig. 2. Inner organisation of the XPDL elements

- *Package:* It is a container that groups together different *processes*/*workflows* and stores characteristics that are common to all of them (author, version, status etc.).The *processes* included in a package inherit the same characteristics, except those that might be redefined at the process level.
- *Process/Workflow:* It contains the definition of the process (workflows, activities, etc.) and information associated with its administration and execution.
- *Swimlane:* They are groupings that facilitate the distribution and viewing of a collection of *processes* and the activities that they contain. There are two types: *pool* and *lanes*. A *pool* contains one or more *lanes*. It groups a set of activities and transitions between them. Activities from different *lanes* in a same *pool* can be connected by transitions; however, transitions cannot be established between activities from different *pools*. *Lanes* subdivide *pools*. They are normally used to specify different roles in the *process* (the activities included in a *lane* inherit said roles, such as *participant* or *performer*).
- *Activity:* They are each of the stages of a *process*. Table 1 shows the different types that exist.

Type	Description
No Implementation	It does not have any associated execution method (there is no application or service to be called in order to execute this activity)
Block	It groups a set of subactivities, but there is no communication channels (neither input nor output parameters).
Sub-Flow	It represents a call to another *process*, with communication of input/output parameters. It permits the reuse of other models by including them as activities in our *process*.
Route	It does not entail a workload, but a decision about which direction the execution flow should follow.
Event	These are facts that occur during the execution of the *process* and can affect several of its phases.

Table 1. Types of Activities that XPDL offers

- *Transition:* It represents movement from one *activity* to another. They may have logical conditions which activate them to be followed or not, depending on their evaluation during the execution of the *process*.
- *Participant*: It represents the various agents that may participate in a *process*. It enables them to be represented at different levels: individuals, administrative roles, organizations etc.
- *Artifact*: They are necessary elements for the complete description of the process. However, they do not affect the flow of the *process*. There are three main types: *Text annotation, DataObject* and *Group*.
- *Association*: An entity that is used to associate the defined *artifacts* with its associated elements.
- *Vendor/User Specific Extensions*: One of the most important characteristics for the adoption of XPDL in the development of the agent: its extension mechanism. Nearly all of the XPDL entities admit the definition of *extended attributes*. Represented by pairs (key, value), they allow the XPDL to be extended using external information that must be interpreted by its corresponding editor (in our case, the intelligent agent).
- *Others:* XPDL is a language focused on the modeling of processes for their later deployment and execution in a workflow engine. This makes it an extensive and expressive language. The AIAM project is not interested in the execution, but in the interpretation of the status of the workflow. Therefore there are XPDL elements that we do not describe as they are not useful for the agent.

3.3 Elements of the Procedures and Mapping to XPDL

Now that we have seen why XPDL is a suitable modeling language for the project, we will explain how the modeling forms part of the processes followed in public administrations using XPDL elements. As we have seen earlier, the agent interprets both formal and factual knowledge based on XPDL models. Therefore, the principal mappings established between the elements of our data model and the elements described by the XPDL standard are as follows:

- *Processes repository – XPDL: Package*

It contains the generic *processes* to be instantiated. It is a *package* that contains the generic workflows that are associated with each *process*. The agent needs to access it to obtain the process model that is to be instantiated in order to generate a *file* that will then be included in the *citizen folder*.

- *Citizen folder – XPDL: Package*

It contains citizen's *files*. It stores *workflows* that are completed using the information about the status of the *file*. This information is obtained by accessing the underlying processing platforms.

- *Process – XPDL: Workflow*

It represents the process in a generic way. It provides the workflow that a *file* needs to follow and information regarding *participants, procedural steps* and *documentation* that must later be requested for each specific *file*.

- *File – XPDL: Workflow*

It represents the status of a *file*. It is comprised of a workflow that includes the information about the status of a *file*: *procedural steps* and *phases* already finished, responsible bodies and generated documentation.

- *Procedural step– XPDL: Activity*

They directly map the various *procedural steps* that make up a *process*. Three XPDL activity types are used:

- Normal procedural steps: They are represented as No Implementation activities as the agent has only to interpret them, not to execute them.
- Decision making/forks: They are naturally mapped to Route activities.
- Common processing blocks (CPB)[13]: A common processing block is a common workflow that appears in different procedures. They are modeled as Subflows as the agent might need information about the current status of the file to interpret them properly, and therefore there is a need of a communication channel in the model (XPDL Blocks do not offer it).

- *Phase – XPDL: Extended Attribute*

The *phase* which each procedural step belongs to is represented as an *extended attribute* of the *procedural step* in question. This figure does not exist in XPDL and therefore we have added it through the *Extended Attribute* mechanism that the standard provides.

- *Body – XPDL: Participant (Organizational Unit)*

It represents the bodies that may be associated with each *process* due to being responsible for its processing.

- *Event – XPDL: Artifact (TextAnnotation)*

It represents the various events that could occur during the lifetime of the *process*. Despite the fact that XPDL offers *events* in its model, there is a major difference between their meanings. The XPDL events are connected to the workflow execution (event-oriented execution), while those of the AIAM model refer to any external occurrence that could affect the process, but which is not included in its workflow (e.g. the death of the requesting party). In view of the difference in meaning between the *Event* elements defined in XPDL and those used in our model, we use a text annotation for each event that occurs, which refers to an external description of the occurrence.

- *Document – XPDL: Artifact (DataObject)*

It represents the documents that are generated during the *process* and associated with the *file*. It is associated with the *procedural step* in which they were added in the *process*.

3.4 Modularity of Our Model

A major concern while designing our model was to keep it extensible and modular. The knowledge modeling must follow a modular approach (as must the software development), as this facilitates the various tasks that take place during the lifecycle of the model (maintenance, revision, reuse etc.) [6], as well as their understanding and adoption by third parties.

In our case, the modularity achieved in the design of the AIAM project provides it with a great deal of flexibility. Indeed, although in this section we have focused on the information modeling of processes, the AIAM agent is not limited to handling procedural knowledge. It also obtains and handles information from other knowledge sources that have been modeled and integrated for their interpretation by the agent. In Figure 3 we can see the organization of the different models that the agent uses. This information includes the different types of documents that may be associated with procedural steps. The AIAM includes a taxonomy of them and of their associated metadata and also, based on the latter, manages the rights to access in the form of logical rules.

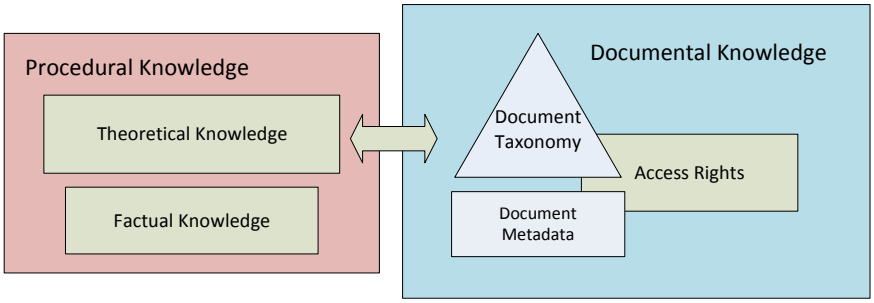

Fig. 3. Knowledge modules of the AIAM

4 The Deployment of the Agent in Municipal Administrations

4.1 The Environments for the Integration of the Agent

In recent years, two workflow tools for the processing of files by Spanish public municipal administrations have been developed. Both are distributed as free software. The first is SIGEM [12], developed through a Spanish Government initiative and the second is W@nda, by the Regional Government of Andalusia. Both tools have their own module for the definition of processes, but the most comprehensive is Model@, which is part of the second initiative and enables diagrams that define processes to be graphically created, managed and maintained, through the XPDL standard [14].

Our agent is developed in partnership with the councils of two cities — Huesca and Zaragoza – and its ultimate objective is its integration into the websites of these two public administrations. Huesca City Council is installing SIGEM for the management of its *files*, although there are still other platforms and the unification of all

departments under SIGEM has not been proposed yet. For its part, Zaragoza City Council has been developing an ambitious project over the last three years so that all of its *files* will be processed using W@nda. Regarding the "catalogues of procedures", the contents of both organizations are on a database, with the objective of constructing their websites dynamically with information for the general public.

Also, given that the agent will give access to restricted information, it is essential that users are identified before using it. Therefore, within the websites of public administrations, the agent will be located in what is called the "citizen folder". Although in our model we have used this name to refer to the group of *files* of a citizen, its usual meaning is an area of websites of public administrations where access can be gained after logging in by members of the public and where all the information regarding their relationship with the organization can be found. The use of a recognized electronic certificate is required in practically all Spanish councils for access to the "citizen folder" [5]. Consequently, AIAM does not include mechanisms for identification, as it receives the user's identity of the environment.

4.2 Architecture and Interface with Users

One of the main requirements in the design of our agent was that it could be integrated into the websites of different administrations minimizing the deployment efforts. The adoption of a shared model and the integration of the available information using XPDL allowed us to develop our agent in a *black box* way, as it only communicates with the underlying information system to obtain that file. Consequently, each organization interested in the deployment of the AIAM agent on its website must develop a service, needed to construct a well-formed XPDL file, in accordance with the definition that has been developed during the project and whose main characteristics were described in Section 3 (see Fig. 4).

When a citizen logins, the AIAM agent invokes the appropriate Web Service deployed by the administration (Fig. 4, step 1) to retrieve the XPDL file containing the corresponding c*itizen folder*. This service consults the *files* currently being processed related to the citizen (Fig. 4, step 2) and then, retrieves the associated *procedures* (their theoretical definitions) to use them to build the final c*itizen folder* (Fig. 4, step 3). With the information available in the underlying system (the administration's *back-office* system) about the citizen and the different aspects of the *files*, the agent builds an XPDL containing all the relevant data (Fig. 4, step 4). Finally, once the XPDL file is created, the agent interprets and explains it to the citizen by means of a graphical and audio-enabled interface (Fig. 4, step 5).

As we discussed in Section 2, and as is reflected in the Figure 4, the knowledge that the AIAM agent handles is based on both theoretical and factual knowledge. The XPDL *procedures* act as the classes of procedures that are instantiated to build the XPDL *files* file. However, although in this section we have focused on the procedure interpretation, we want to remark (recalling Section 3. 4) that the information that the AIAM agent manages is not restricted only to the content of the XPDL file, as the model has been designed to be extensible through the use of external URIs (following the principles of Linked Data [2]).

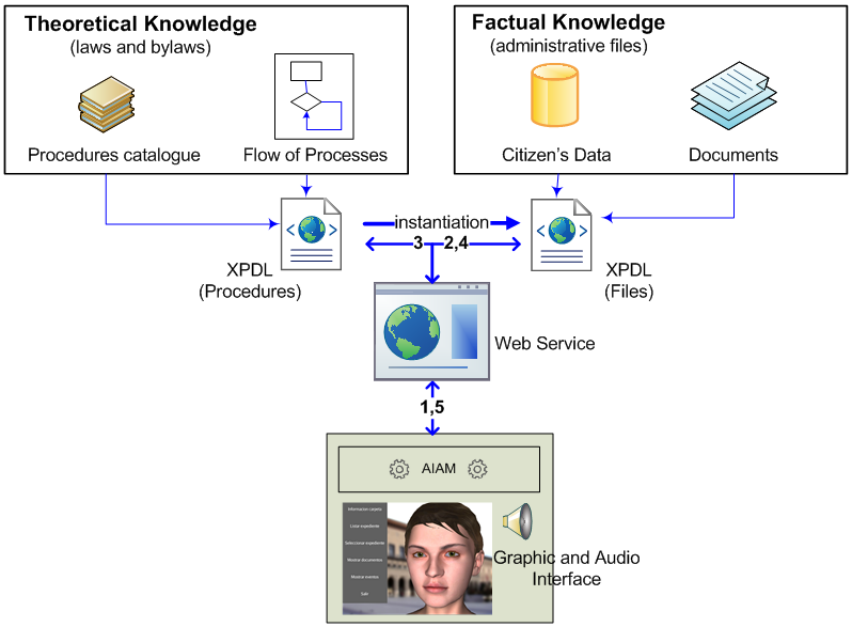

Fig. 4. Architecture of the AIAM agent

4.3 Experiences Learned

4.3.1 The Lack of Information Available for the Public

Apart from usability, the most important question when it comes to evaluating the functionality of the agent is the quality and quantity of information provided by it. However, this obviously depends on the existence of knowledge about the *files* in the information system of the council and on the way this knowledge is organized. We have seen that there are three information sources used by the agent. The first is the "catalogue of procedures", used for members of the public when they have a need and are searching for the appropriate procedure for their request and which is normally the most developed source in organizations. As future work, the possibility of extending the functionality of the agent to this phase prior to the initiation of *files* has been proposed, but at the moment, the agent focuses on procedural knowledge and uses information from the catalogue only as an accessory to describe some characteristics of the procedure followed in each file.

The second type of information is the one about files. Metadata are its main source, both about the file and the documents associated with it. Although very comprehensive sets of metadata are defined in the consulting phase prior to the development of the processing platforms, the ones that were really included in the development were the essential set for the management of *files*, and even then many lacked content. The result is that the information that may be provided to users is quite bare. Despite the above, there are interesting possibilities such as adding

geographical information associated with files [1]. In our case, Zaragoza City Council has a very comprehensive geographical referencing system, named iDEZar [11], and the metadata from this type of information are completed in the majority of files. This has made it possible for the agent to connect to the geographical information of the organization, placing the *file* on a map when it is associated with a spatial reference. The encoding used for the exchange of this information is UTM30N, as this is what is used in IDEZar.

Finally, the third type of information is that regarding the procedural flow, including the various steps to be taken, bodies responsible for them and the specified periods for their execution. In this case there is also a lack of available information as the organizations chose to define a generic process model for the setting up of the processing platforms. This model is used for all procedures and council workers adapt each *file* process through the options to add new steps, which includes the workflow tool. Therefore, currently the AIAM can only report on common processes, which in many cases do not match the process that the file ultimately follows. However, the perspective is that this situation will improve as councils plan to model new processes, with Zaragoza City Council preparing to model the processes corresponding to town planning procedures, which is one of the largest areas of the council. This work is carried out in order to incorporate the procedural flow into the processing platform. However, it can be re-used immediately to improve public information, without requiring additional work.

4.3.2 The Need of Metadata about Access Permissions

One especially serious consequence of the lack of definition and completion of the metadata is that there are many people who have the right to access *files* and cannot do so. The reason is the lack of metadata regarding the interested parties in *files*, which would be the means that the agent would have to decide who is authorized to access them and who is not. In the two councils where the agent is being installed, the only information from files regarding access permissions for members of the public is the identity of the promoter, who is the person that initiates the *file*. Consequently, even when the promoter acts as the representative of a person it is not possible to give file access to this person, as there is only information in the system about the representative.

There should also be metadata about permissions in documents, indicating, for example, whether they contain protected personal data or not. Also in this case, it should be remembered that, once files are finalized, some documents should be added to repositories accessible to the public as the law specifies free access to the documentation of finalized files. This legal provision was not complied with to a large extent when documents were on paper, but electronic management would enable automation and consequently would make the exercise of this right truly effective. Consequently, documents must include metadata regarding the existence or not of limitations to public access in each specified document from the time of their creation [10]. However, experience tells us that it is not enough to include metadata in the structure of documents, but that they should be assigned a content by default when

preparing the templates of administrative documents, to ensure that document's metadata are not left empty. This is possible because in many cases it is possible to forecast, by being aware of the document's content type, if it will or will not be affected by these restrictions. Therefore the right to free access to the information of public administrations would be guaranteed, along with any rights that might be affected by access, because if access is allowed on template by default and there is an exception, a council worker would always be able to change the appropriate metadata in the corresponding instance of the document.

5 Conclusions

In this paper, we have proposed a model for both the theoretical and factual dimensions of the procedures that are part of the relationship between public administration and public. This model allows representing this information to be exploited to make it more accessible to the citizens. Regarding this, the AIAM project is defined as a line of research of our group dedicated to Open Government, whose objectives are the study and development of legal and technical mechanisms that enable the right of citizens to free access to information from the public sector to be supported as far as possible.

The experience gained through the AIAM project has enabled us to conclude that XPDL, which is a standard designed to automate the processing of procedures through workflow tools, can also be used as a basis for an agent that informs members of the public about the status of their *files*, by integrating the information about procedural steps already taken in their processing with that relating to what still needs to be done. Also, using the strategy followed in the project, this can be done by using the work carried out for the automation of the back-office of public administrations, which is thereby reused in order to inform citizens in the front-office. This agent, which integrates theoretical and factual knowledge, can also be applied in order to help civil servants in the processing of files, which is seen as one of the future lines for the wider application of the results of the project.

Finally, within this line of research, we have seen in several projects that it is important that requirements regarding public access to information are incorporated right from the initial developmental phases of electronic administration systems. Usually, only requirements proposed by the civil servants responsible for procedures are specified, but they do not prioritize the exercise of these rights and usually interpret them in a restrictive way. The result is that computer programs are constructed in such a way that it is not easy, and in many cases not even possible, to comply adequately with the rights of the citizens in this regard.

References

1. Almeida Y., Feitosa H., Gonçalves R., Ramos A., de Souza C.: Visual Modeling of Workflow with Support to Multimedia and Spatiotemporal Indexing, in: Andersen K. N.,

Francesconi E., Grönlund Å., Van Engers T. M. (eds.) EGOVIS 2011, LNCS, vol. 6866, pp. 184-196, Springer, Berlin (2011)
2. Bizer C., Heath T., Berners-Lee T.: Linked Data-The Story So Far, in: Heath T., Hepp M., Bizer C. (eds.) International Journal on Semantic Web and Information Systems (IJSWIS), vol. 5, 3, pp. 1-22(2009)
3. Burgos P., Simino M.T., Rodelgo M.A., Ranz A.: Gestor de Expedientes Normalizado GEN, in: Tecnimap 2010, Zaragoza (2010)
4. Business Process Model and Notation (BPMN), http://www.omg.org/spec/BPMN/
5. Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures, article 6
6. Gómez-Pérez A., Fernández M., Corcho O.: Ontological Engineering, Springer, Londres (2004)
7. Gruber T.R.: Towards Principles for the Design of Ontologies Used for Knowledge Sharing, in: Guarino N., Poli R.: Formal Ontology in Conceptual Analysis and Knowledge Representation, Kluwer (1993)
8. Hawryszkiewycz I.T.: Process Modeling Semantics for Complex Business Environments, in: Janssen M., Lamersdorf W., Pries-Heje J., Rosemann M. (eds.) EGES/GISP 2010, IFIP AICT, vol. 334, pp. 155-166, Springer, Berlin (2010)
9. Nešković S., Paunović O., Babarogić S.: Using Protocols and Domain Specific Languages to Achieve Compliance of Administrative Processes with Legislation, in: Andersen K.N., Francesconi E., Grönlund Å., Van Engers T.M. (eds.) EGOVIS 2011, LNCS, vol. 6866, pp. 284-298 Springer, Berlin (2011)
10. Oliver-Lalana A.D., Muñoz J.F.: Collisions Between Data Protection, Transparency and Efficiency, in: Galindo F., Traunmüller R. (eds.) E-Government: Legal, Technical and Pedagogical Aspects, pp. 349-364, Universidad de Zaragoza, Zaragoza (2003)
11. Portolés-Rodríguez D., Álvarez P., Béjar R., Muro-Medrano P. R.: Idezar: an Example of User Needs, Technological Aspects and the Institutional Framework of a Local SDI, in: EC-GI&GIS/ESDI 2005, pp. 56-58, Utrecht (2005)
12. Sigem, http://www.planavanza.es/avanzalocal/soluciones/paginas/sigem.aspx
13. Verginadis G., Gouscos D., MentzasG.: Modeling e-Government Service Workflows Through Recurring Patterns, in: Traunmüller R. (ed.) EGOV 2004, Springer, Berlin, 483–488 (2004)
14. Wanda, http://www.juntadeandalucia.es/repositorio/usuario/listado/fichacompleta.jsf?idProyecto=33
15. Wimmer M., Palkovits S.: Processes in e-Government–A Holistic Framework for Modelling Electronic Public Services, in: Traunmüller R. (ed.) EGOV 2003, LNCS, vol. 2739, pp. 213-219, Springer, Berlin (2003)
16. Wooldridge M.J., Jennings N.R.: Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, 10 (2), 115-152 (1995)
17. XML Process Definition Language (XPDL), http://www.wfmc.org/xpdl.html
18. Zafeiris V., Doulkeridis C., Belsis P., Chalaris I.: Agent-mediated Knowledge Management in Multiple Autonomous Domains, in: Jurriaan van Diggelen J., Dignum V., van Elst L., Abecker A. (eds.) AAMAS/AMKM 2005, Utrecht, 97-102 (2005)