Developing a One-Stop Government Data Model

Olivier Glassey

Developing a One-Stop Government Data Model

Olivier Glassey
Swiss Graduate School of Public Administration (IDHEAP)
Quartier Mouline UNIL
CH-1015 Lausanne
olivier.glassey@idheap.unil.ch

Abstract

In previous research work we defined a conceptual model of a one-stop public administration and we created a prototype to illustrate it. In parallel we participate in a European research project that is developing an integrated e-government platform. In order to do so the project consortium created a governmental markup language (GovML) that describes public services and life events in XML format. It also defined metadata schemas in RDF for searching, locating and retrieving governmental digital resources. We decided to integrate our personal research and the results of the consortium work in order to develop a data model for One-Stop Government. In this paper we explain how we extended our conceptual model to support GovML and RDF, and how we were able to model administrative services at the data and meta-data levels.

Keywords

E-government, electronic services, process model, data model, knowledge management, markup language.

1. Introduction

New technologies are used in many public administrations to deliver improved or new services to their clients. This field of work is commonly called e-government and is becoming increasingly important for administrations and their commercial partners, as well as in the academic world. In previous research work we defined a conceptual model of a one-stop public administration, studying both the structural and behavioral aspects of such a system. Indeed we believe that many researchers and public administrations need stable models of processes that can help them managing the complexity and the rapid evolution of new technologies. In the second section of this paper we explain briefly the steps we went through in order to build a model of a virtual one-stop public administration, from gathering users requirements, both amongst civil servants and citizens, to building a conceptual model. We believe there is a need for conceptual methodologies in that field and various researches were done in that field, such as the GAEL (Guichet Administratif En Ligne) project in Switzerland [1] or the BTÖV (Bedarf für Telekooperation in der Öffentlichen Verwaltung) method in Germany [2]. Moreover it has to be noted that a Swiss working group on e-government appointed by the federal government made similar remarks [3]. Furthermore this group identified the definition of patterns for such applications as one of three essential action domains for e-government. In that regard, let us also mention that we developed a prototype of one-stop public administration based on a component architecture distributed over the Internet, in order to validate and to refine the conceptual model.

However the goal of this publication is not to go into the details of the model nor of the prototype, although we will give a global view of the model in the next section. We rather intend to explain our current research that aims at expanding this conceptual model with work done in parallel within a European project named “An Integrated Platform for Realizing Online One-Stop Government (eGOV)”. This “Information Society and Technology” 5th Framework research project is lead by a consortium of commercial companies, public administrations and academic partners. Its main goal is to develop an integrated e-government platform and we will shortly describe it in section 3. In the fourth section we will present one part of the project that is relevant to this publication: the consortium defined a tailored XML vocabulary for describing public services and life events. Called GovML this language should enable data flow exchange amongst public administrations and provide a valuable tool to build electronic services. In our own research work we are currently extending our conceptual model to integrate GovML. In order to realize that we rely on existing CASE tools and we are developing specific stereotypes to model electronic services. In the fifth section we will present the work of the consortium on RDF (Resource Description Framework) and on metadata about electronic administrative services. Section 6 will give some indications on our own work where we use RDF schemas to develop data models for administrative services.

2. Conceptual Model

2.1 Field study and users requirements

We studied administrative processes in a large public administration in Switzerland, for there was a need to clearly identify and normalize all these different processes (internal processes, relations with its clients, services, procedures, etc.) before it was possible to dematerialize them [4]. We spent six months in the "Administration Cantonale Vaudoise" (ACV) and we met project managers, domain managers, departmental managers and users representatives for a total of 28 formal interviews. We also had many casual talks with different civil servants. This field research allowed us to collect a large amount of data on administrative processes, but it was rather heterogeneous since we met various people with very different focuses. We had to find a way to process that data and we will see below how we managed to do it. Moreover we conducted two online surveys (1998 and 2000) in order to discover the expectations of the citizens in the field of electronic administrative services and we had close to 500 questionnaires to analyze.

Service name	**Public health inspection**
Short service description	The department of Public Health has to inspect restaurants, shops, enterprises, etc. to verify if they comply with public health standards. It makes routine controls where the clients are aware that they will be checked and it also sets up unexpected controls. The results are made public.
Supplier	Department of health
Clients	Shops, restaurants, enterprises, medias, citizens
Uses types	Initiation, termination, notification, validation, publication, modification, consultation, search
Intermediaries/ partners	Consumers association, laboratories

Figure 1: Summary card

Figure 1 shows a summary card of a simplified example of a public health inspection service. After the interview stage, we filled in these summary cards to begin with our iterative modeling process. These cards are loosely based on CRC cards (Class-Responsibilities-Collaborators), an informal class modeling technique with some distinct advantages over simply starting with UML (Unified Modeling Language) class diagrams [5]. For detailed information on this technique, we recommend [6]. These cards helped us collecting the input from the various people we met and transforming it into an object model through an abstraction process. For that model we focused on the services that the ACV delivers to its different types of clients, through different intermediaries or partners.

With the data gathered during our field research we prepared over 90 summary cards, although we did not go into details for each of these administrative services. From these cards it was fairly straightforward to define class descriptions, such as shown in figure 2. We settled for 65 types of services that an administration can deliver to its clients and classified them according to the widely used typology of Information-Communication-Transaction services. We also defined 50 types of clients that we will present further on.

Service class	**Public health inspection**
Super-classes	Information - Communication
Sub-classes	
Methods	Initiated - Terminated - Notified - Validated - Published - Modified - Consulted - Searched
Attributes	Name - Description - Key_words - Supplier - Clients - Intermediaries - Supports - Payment
Constraints	
Collaborators	InstitutionalCl - BusinessCl - InternalCl

Figure 2: Class description

2.2 Structural model

We worked on this model using object-oriented (OO) technology and the graphical notation language UML (Unified Modeling Language) and we discussed this choice in details in one of our working paper. Here we assume that the reader has good notions of OO technology, but further reference on UML can be found in [7]. This object-oriented notation language has a very wide base of users, is an Object Management Group standard and combines the strengths of various object methods. It offers nine types of diagrams, amongst them the use cases diagrams that are very useful to formalize users needs. Besides the notation has been extended to model Web applications [8]. Such as stated in [7], we build models so that we can better understand the system we are developing. Indeed a good model allows us to specify the structure and the behavior of a system, that is static and a dynamic view of the system. In order to build the structural blueprint we used the class descriptions and the summary cards. We also worked on scenarios and use cases for the creation of the behavioral model, but we will not discuss that here, as it does not integrate with the GovML nor RDF extensions.

The structural model is made of class and object diagrams that show their attributes, their methods and the relationships between them. To build the hierarchy of the class diagrams from the class descriptions and the summary cards, we began with a fairly simple base diagram (Fig. 3). A client requests a service from an administration and the supplier of this service can call different intermediaries if necessary, before delivering the service on whatever support suits best.

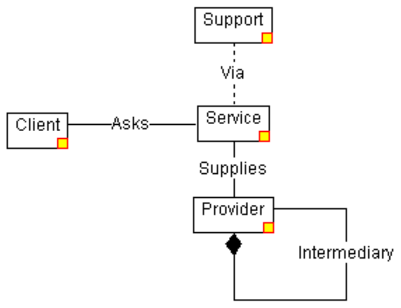


Figure 3: Base diagram

In order to build these class diagrams we used a CASE tool that allowed us to create cascading diagrams. That means that users can navigate through the class hierarchy by a simple click of the mouse. It is out of the scope of this paper to show the operations and the attributes for all the classes, however we designed two synthetic diagrams showing the services and the clients hierarchy (Fig. 4 and 5). Furthermore we created an index of over a hundred themes that can be used to classify services.

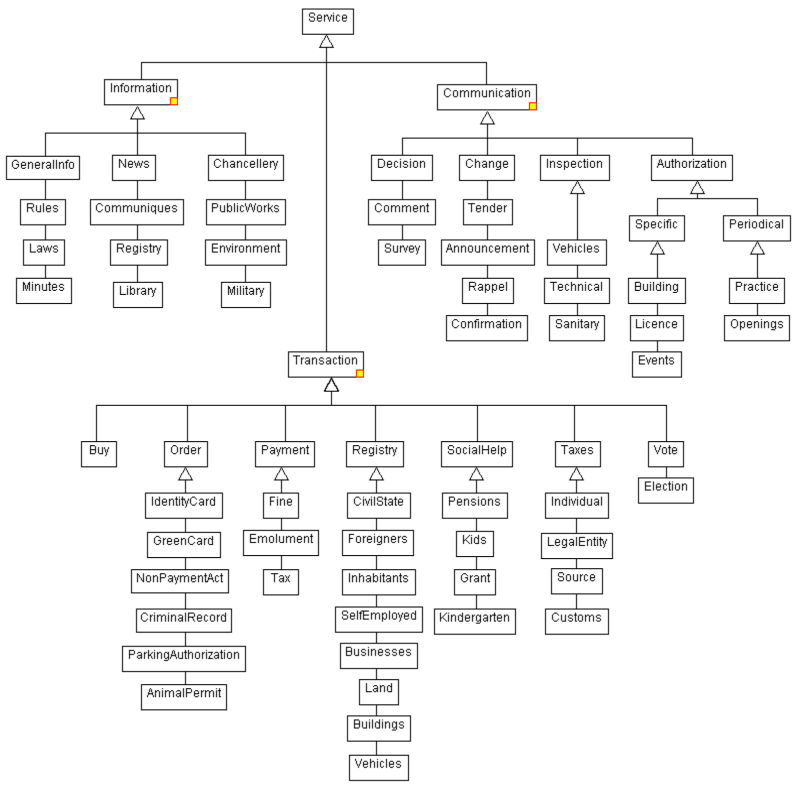


Figure 4: Typology of services

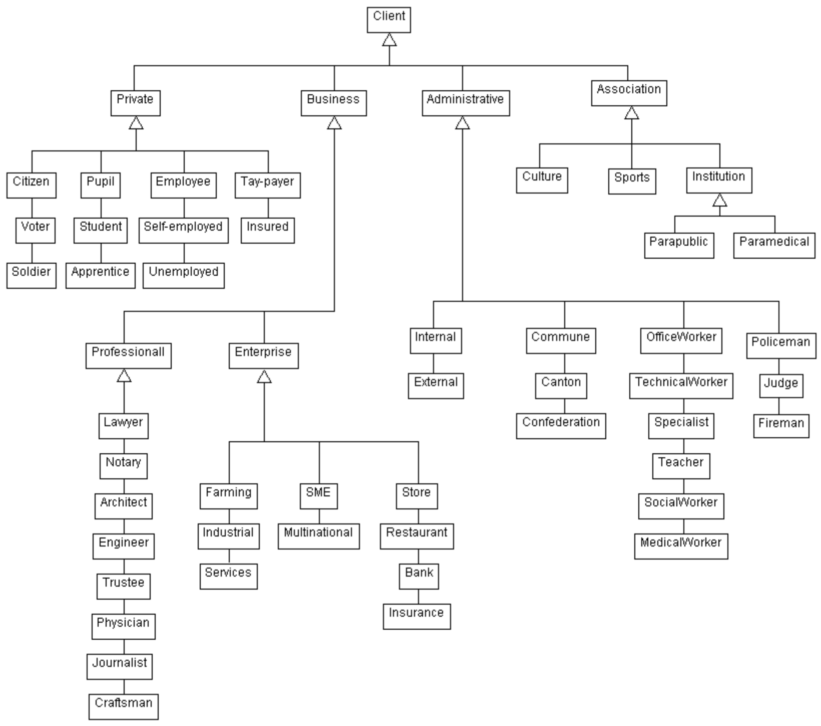


Figure 5: Typology of clients

We also identified the actors of the service delivery: suppliers, clients, intermediaries, etc. Then we established a typology of the suppliers: we decided to divide them into 6 general functions and 28 departments or offices. These divisions do not correspond to a real administration structure, but we think this functional visualization can well be used to represent real administrations with very different organizational structures. Last we defined the ways services can be provided (synchronous, asynchronous) and the supports to be used: mail, fax, electronic delivery, face-to-face, etc. Here we really want to emphasize the fact that this class model represents one point of view of a public administration in Switzerland and might not be of direct use elsewhere. However it can be modified or extended at any point and one does not have to use it as a whole. Although we realize this is only a partial representation of the reality and do not think that this is exhaustive, we believe that it does a good job covering what roles a public administration has to play in a modern society and that it constitutes a good backbone for our model.

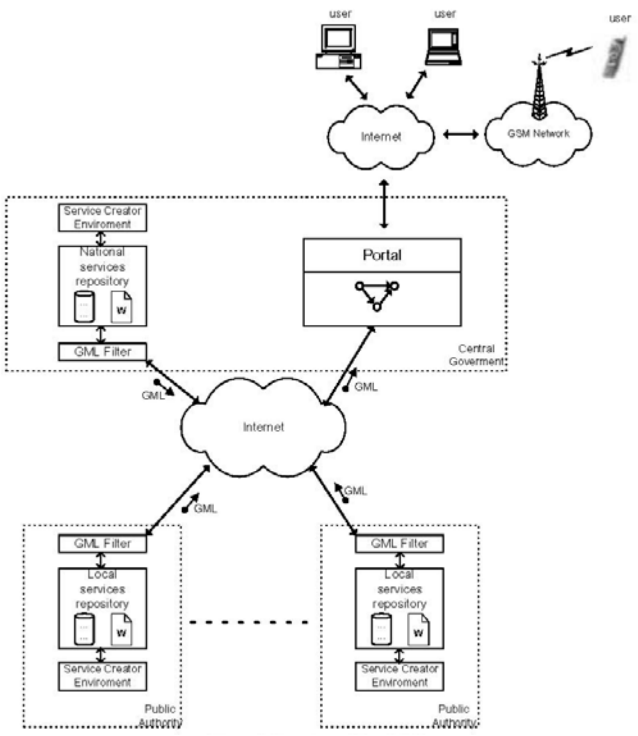
3. eGov Project

The eGov platform is being developed in a two-years research project (2001-2003) financially supported by the European Commission under the Information Society Technologies Program of the Fifth Research and Technology Development framework. The main objective of this project is to develop, deploy, demonstrate and evaluate an integrated platform for realizing online one-stop government. In order to achieve this, the consortium has to:

- Develop a governmental portal that offers advanced features to the citizens: personalization, multilingualism, authentication, accessibility, etc.
- Develop content and service repositories (both at the national and local levels), as well as a service creation environment used to administrate the repositories.
- Implement a governmental mark-up language (GovML) that describes public services and life events in XML format and that defines metadata used for searching, locating and retrieving governmental digital resources.

- Create and supply a number of administrative services to be used during the testing phase.
- Deploy and evaluate the platform in Austria, Greece and Switzerland at the national and local levels.

The general architecture of the platform is illustrated in figure 6. Citizens and businesses access administrative services through the portal where these services are categorized with “life-events” and “business situations” metaphors. As the portal is linked through Internet and GovML to the national service repository and the local service repositories, users can obtain services from different administrations at various levels in a transparent and integrated manner. If they request information services they will get the data from the relevant repository, and if they need transaction services, these will be executed in the service runtime environment before the results are send back to the users through the portal. On their side, the public administrations maintain their service repositories with the service creation environment.


user
user
user
Internet
GSM Network
Service Creator Enviroment
National services repository
GML Filter
Portal
Central Goverment
GML
GML
Internet
GML
GML
GML Filter
Local services repository
Service Creator Enviroment
Public Authority
GML Filter
Local services repository
Service Creator Enviroment
Public Authority
Figure 6: Overview of eGov platform

Details on one-stop government and on the eGov project are available at *http://www.egov-project.org/* and in different publications, for example [9] and [10].

4. GovML Description Language

As mentioned above, one of the main goals of the eGov project is to define a description language that supports the standardization of governmental information. In order to do so, the consortium conducted a survey of international best practices, both in the domain of XML and of governmental data exchange frameworks. It then identified the requirements of the project and the data elements necessary to create the GovML sub-vocabularies. With the input of all partners and after several iterations to refine the data models, the final vocabulary was specified according to the relevant ISO/IEC standard and the XML schemas were developed. Two categories were identified for content description. Common content for all public authorities that is created only once for all and that is centrally stored at the governmental content repository is called **Generic Data Vocabulary** for public services. Specialized content related to service provision and execution by a specific public authority is identified as **Specific Data Vocabulary** for public services. Moreover GovML vocabulary for **Metadata** elements related to information resources was created with RDF (*http://www.w3.org/RDF/*). The Dublin Core Metadata Initiative (*http://dublincore.org/*) is considered the core of the GovML metadata vocabulary. Moreover, GovML metadata vocabulary use some elements of the electronic Government Metadata Standard (eGMS), a part of the British e-Government Interoperability Framework (*http://www.govtalk.gov.uk/*).

Complete specifications of the GovML vocabularies are out of the scope of this paper and they are available on the project website. However we mention some GovML features in order to show the general ideas behind this language. For the **Generic Description**, 19 attributes were selected, such as *Title, Description, Language, Related Services, Audience, Periodicity, Time to deliver, Cost Information*, etc. They are used to describe an administrative service in a general way. For precise information on a given service, the **Specific Data Vocabulary** adds attributes such as *Public Authority Name* and *Department, Eligibility*, *Contact Details*, *Procedure* to follow, *State* of a service, *Delivery Channel*, etc. **Metadata** will be used in the eGOV platform to search and retrieve governmental resources across the Internet and to develop the navigation tools of the portal based on life-events and business situations. Twelve metadata attributes were selected, such as *Title*, *Creator, Publisher*, *Format*, *Language*, *Has Translation*, etc.

The eGov platform will provide a tool called the Service Description Tool that allows administrative users to create GovML content and documents with a GovML syntax that will conform XML schemas. These documents can be translated in different languages and will be used respectively in service repositories (generic and specific data) and in the portal (metadata). We believe that the GovML description language and the supporting architecture of the platform is one of the major strengths of the eGov project and therefore we want to integrate it in our own theoretical work.

5. RDF Schemas

In this section we will now see more details about RDF (Resource Description Framework), an emerging standard for defining metadata for encoding machine-readable semantics [11]. This W3C recommendation for metadata is based on XML and uses graph theory to represent knowledge. RDF statements can be in the form of three different representations that are mathematically equivalent. The first representation is a labeled directed graph that shows the relations between resources and literal values. A resource can only point to another resource or to a literal value (Fig. 7). By combining resources and labeled relations it is possible to build a RDF schema showing the class hierarchy and the type of the resources (Fig. 8). The second representation of an RDF statement is a triple *{Subject*, *Predicate, Object}* where the subject is normally identified by a URI (Universal Resource Identifier), the predicate is an attribute of the resource and the object is a literal value or another resource value. These triples can be used in software applications and stored in relational databases.

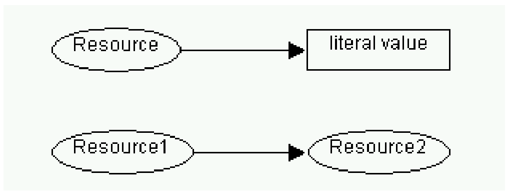


Figure 7: Resources and literal values

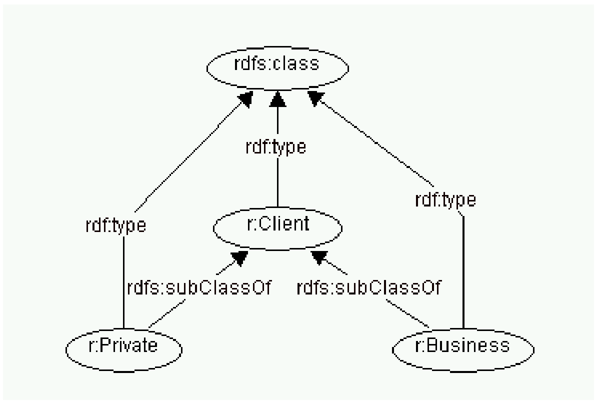
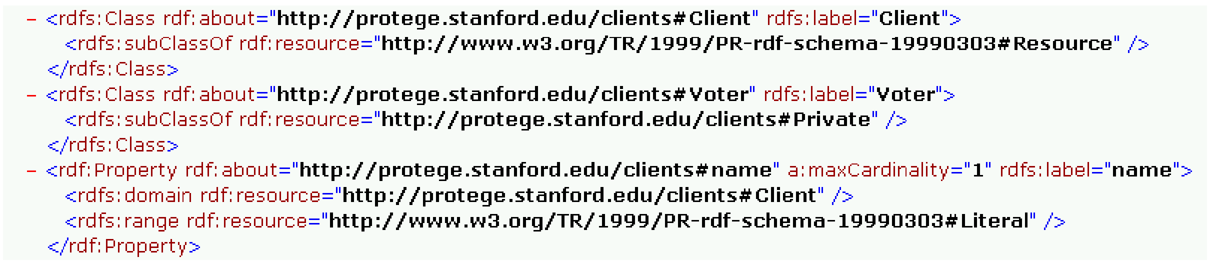


Figure 8: Labeled directed graph

The third mathematically equivalent representation of RDF statements is XML-based (Fig. 9) and can be easily exchanged between computers and applications.


```
- <rdfs:Class rdf:about="http://protege.stanford.edu/clients#Client" rdfs:label="Client">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Resource" />
  </rdfs:Class>
- <rdfs:Class rdf:about="http://protege.stanford.edu/clients#Voter" rdfs:label="Voter">
    <rdfs:subClassOf rdf:resource="http://protege.stanford.edu/clients#Private" />
  </rdfs:Class>
- <rdf:Property rdf:about="http://protege.stanford.edu/clients#name" a:maxCardinality="1" rdfs:label="name">
    <rdfs:domain rdf:resource="http://protege.stanford.edu/clients#Client" />
    <rdfs:range rdf:resource="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Literal" />
  </rdf:Property>
```
Figure 9: example of RDF definition in XML

As mentioned before we wanted to extend our conceptual model and to add knowledge in this structural representation. Indeed we agree with [12] and believe that this addition of knowledge will allow us to share a common understanding of the domain and to reuse this knowledge. Furthermore it will allow use to separate domain knowledge and operational information and thus to analyze this domain-specific knowledge. RDF being a suitable format for predicate logic and for ontology modeling [13], we chose to use this metadata framework to do so. We transformed the simple class model presented in figure 3 into a labeled directed graph that was a first (and simple) step towards building a RDF schema. Figure 10 shows the same information (without the *Support* class) and adds literal values for a purpose of illustration.

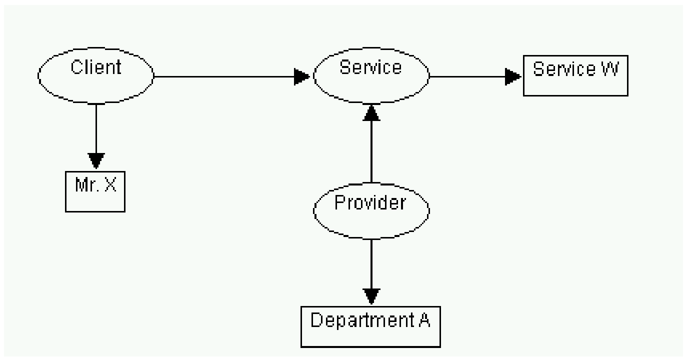


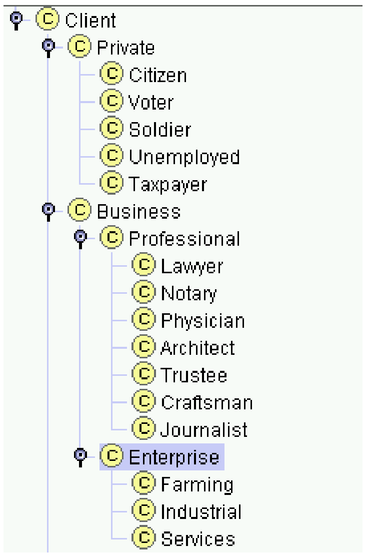
Figure 10: Simple RDF schema

A simple model as shown in figure 10 can be built with any drawing tool, but in order to develop a solid data model and to generate proper RDF documents we used the Protégé-2000 tool (*http://www.smi.stanford.edu/projects/protege/)*. This work is what we will see below.

6. Simple Data Model

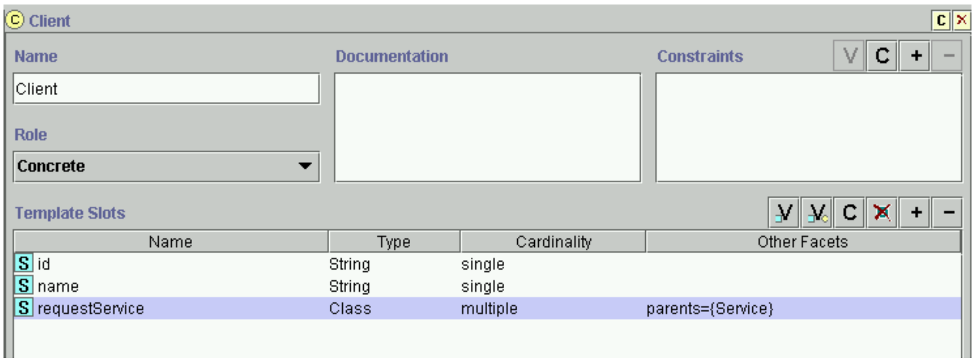
Protégé-2000 is a meta-tool that helps users to build domain-specific knowledge bases within a simple graphical interface. It allows them to concentrate on the concepts and the relationships of a given domain without having to worry about correct syntax or mark-up tags [14]. Developed at the Stanford Medical Informatics Institute, this tool is component-based and platform independent. According to [11], Protégé's own knowledge model is frame-based and consists of classes (concepts), slots (properties), facets (allowed slot values) and axioms (additional constraints). Projects created with Protégé can be saved in the form of text files, JDBC databases, RDF/XML documents and OIL files. The OIL (Ontology Inference Layer) proposal was developed within two Information Society Technology projects of the European Community. OIL consists of different layers that are added on top of RDF schemas. For this work we used the RDF format, but it is interesting to note the flexibility of Protégé models. Let us also mention that the Dublin Core Metadata Initiative is fully compatible with Protégé models and that [13] developed a Dublin Core metadata ontology in Protégé. The source files of this work are available at *http://www.smi.stanford.edu/projects/protege/.*

In order to build a formal explicit description of a one-stop government platform we reused the class models developed earlier in UML and with Protégé we redefined the classes and their attributes. Part of the class hierarchy of this knowledge base is shown in figure 11. The window used to create the slots (attributes) of a given class is illustrated in figure 12. At the time being we have not done any further development than this formal declarative ontology, but we found that Protégé-2000 was a very efficient editor to create RDF documents. We were also able to render our RDF model with the SiRPAC (Simple RDF Parser & Compiler) tool that is available at *http://www.w3.org/RDF/Implementations/SiRPAC* and that lets the user see RDF/XML documents in the form of triples and labeled directed graphs.



- Client
  - Private
    - Citizen
    - Voter
    - Soldier
    - Unemployed
    - Taxpayer
  - Business
    - Professional
      - Lawyer
      - Notary
      - Physician
      - Architect
      - Trustee
      - Craftsman
      - Journalist
    - Enterprise
      - Farming
      - Industrial
      - Services

Figure 11: Class hierarchy



Client

Name: Client

Role: Concrete

Documentation

Constraints

Template Slots

Name	Type	Cardinality	Other Facets
S id	String	single	
S name	String	single	
S requestService	Class	multiple	parents={Service}

Figure 12: Class and slots

7. Conclusions

Although we did not present the one-stop government architecture we developed nor the GovML and RDF specifications in full details, we think that this paper showed the general concepts of our proposed data model although we would yet have to develop potential applications for a one-stop government platform. However we believe that it is of importance to develop such data models and to use appropriate supporting tools to help public administrations creating and delivering electronic administrative services. Indeed we think that the combination of RDF schemas and XML dialects such as the GovML language will allow the exchange of data and knowledge between different software components of a one-stop government system and between public administrations and their clients. Hopefully the development of ontology such as the one presented here will allow the provision of “intelligent” electronic services by supplying relevant knowledge to the users of a one-stop administrative portal. However we realize that this model stays rather theoretical and cannot be used as is in public administrations that need mature and solid domain ontology. It is rather to be taken as a simple example for e-government projects.

8. Notes and References

[1] Chappelet, J.-L. & Le Grand, A. (2000). Towards a Method to Design Web Sites for Public Administrations. *Proc. IFIP Working Conference on Advances in Electronic Government.* Zaragoza, Spain.

[2] Gräslund, K., Krcmar, H. & Schwabe, G. (1996). The BTÖEV Method for Needs-driven Design and Implementation of Telecooperation Systems in Public Administrations. In: In: B. Fitzgerald & N. Jayaratna. *Information Systems Methodologies*. London: British Computing Society.

[3] GSCI (2000). Guichet virtuel: la communication électronique avec l'administration, le Parlement et les tribunaux. Groupe de Coordination Société de l'Information, Switzerland.

[4] Baquiast, J.-P. (1999). Le guichet unique dans l'administration française, *New interfaces between Administration and Citizens, One-stop-government through ICT*. Bremen, Germany: International Workshop.

[5] Harmon, P. & Watson, M. (1998). *Understanding UML: The developer's guide with a Web-based application in Java*. CA: Morgan Kaufmann.

[6] Bellin, D., Suchman Simone, S. & Booch, G. (1997). *The CRC Card Book*. MA: Addison-Wesley.

[7] Booch, G., Rumbaugh, J. & Jacobson, I. (1999). *The Unified Modeling Language User Guide*. MA: Addison-Wesley.

[8] Conallen, J. (2000). *Building Web Applications with UML*. MA: Addison-Wesley.

[9] Tambouris, E. (2001). An Integrated platform for Realising Online One-Stop Government: The eGov Project. *Proceedings of the DEXA International Workshop "On the Way to Electronic Government*. CA: IEEE Computer Society Press.

[10] Wimmer, M. & Traunmueller, R. (2002). Towards an Integrated Platform for Online One-Stop Government. *ERCIM News, Special Theme: e-Government, Issue 48*.

[11] Noy, F.N., Fergerson, R.W. & Musen, M.A. (2000). The Knowledge Model of Protégé-2000: combining interoperability and flexibility. Available at: http://www.smi.stanford.edu/projects/protege/. Accessed January 12, 2003.

[12] Noy, N.F. & McGuinness D.L. (2000). Ontology Development 101: A Guide to Creating Your First Ontology. Available at: http://www.smi.stanford.edu/projects/protege/. Accessed January 12, 2003.

[13] Kamel Boulos, M.N., Roudsari, A.V. & Carson, E.R. (2001). Towards a Semantic Medical Web: HealthCyberMap's Dublin Core Ontology in Protégé-2000, *Fifth International Protégé Workshop*, Newcastle, England.

[14] Noy, F.N., Sintek, M., Decker, S., Crubézy, M., Fergerson, R.W. & Musen, M.A. (2001). Creating Semantic Web Contents with Protégé-2000. *IEEE Intelligent Systems*. CA: IEEE Computer Society Press.