Chapter 19
Ontologies for E-government

Knut Hinkelmann, Barbara Thönssen, and Daniela Wolff

19.1 Motivation

Serious efforts have been made for years to bring e-government forward. Although according to an e-government study 2006 nearly 50% of the examined services offered by EU states are fully available online (Capgemini, 2006) the majority of services on governmental web sites (especially of small municipalities) are in "stage 1" (information) or "stage 2" (interaction)[1]. That is: providing online information about public services, e.g. opening hours or addresses of departments or provision of downloading of forms. Processing of forms, including authentication, ("stage 3" - two-way interaction) and mainly "stage 4" (transaction, that is full case handling, decision and delivery e.g. payment) is reached only for a few services. Taking into consideration that only 20 basic public services (12 for citizens, 8 for businesses) have been evaluated in the study (Chevallerau, 2005) – out of more than 1,000 (e.g. in Switzerland approximately 1,200 services have been identified (eCH0015, 2006)) – the necessity for new approaches is palpable.

There are recent developments and trends on various levels that can have significant influence on the progress of e-government:

- *architecture*: In a service-oriented architecture processes are supported by independent services that are made available on a network instead of having complex monolithic systems. They can be accessed via defined interfaces while implementation details are hidden to the user. This is particularly useful in administrative processes where various authorities are often involved.
- *technology*: web services are a technology for implementing service-oriented architectures in an internet environment. They provide standards for identification

K. Hinkelmann (✉)
University of Applied Sciences Northwestern Switzerland FHNW, Olten, Switzerland
e-mail: knut.hinkelmann@fhnw.ch

[1]The classification is taken from IDABC (Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens) (Chevallerau, 2006)

R. Poli et al. (eds.), *Theory and Applications of Ontology: Computer Applications*,
DOI 10.1007/978-90-481-8847-5_19,

and invocation of services. In particular, they offer the advantage of using common services instead of each administration having to implement each service itself.
– *semantics:* With the semantic web new standards for ontologies are developed for a machine-understandable representation of the semantics of (web) services, thus enabling the finding of relevant information as well as an intelligent discovery, composition, invocation, monitoring and maintenance of (web) services on the internet. Thus, ontologies can be seen as an enabler bringing service-oriented approaches to their full potential.

As public services are based on legal rules and regulations binding for all public authorities, various service providers (e.g. all municipalities) have a (broadly) similar service portfolio. However, neither naming nor service presentation at the particular portals, nor least of all service execution are similar. This challenges the citizen to find the same service provided by different municipalities and hinders the administrations' provision of one-stop-services or even integration. Ontologies are a promising answer to particular challenges in e-government:

– *finding services and information*: Ontologies contribute to a common understanding of service description. Information about services can be found on web pages of public authorities and on specific portals (e.g. www.ch.ch in Switzerland). Although most of the web pages use life events as a structuring principle, they differ a lot in naming and structuring of life events. A "life event" is understood as a special situation in a person's life, like marriage, childbirth, house building etc. that requires a public administration's services; similarly, "business situations" are defined for companies, reflecting their requirements such as applying for work permits or importing goods. Ontologies can contribute to a common understanding of a domain and allow for mapping different information structures, e.g. mapping of life events differently presented on different e-government portals or web sites.
– *process design and implementation*: As various service providers have broadly the same service portfolio, the sharing and reuse of experiences and domain knowledge can significantly reduce effort and increase quality of services. Initial steps are made with repositories of reference models, best practices and use cases. Adding semantic metadata to these resources improves the identification of relevant resources. In addition, enhanced modelling methodologies with explicit representation of design rationales and design decisions help a service provider to customize a process for a new context. Both semantic metadata and design decisions can be modelled as ontologies to provide the needed control.
– *interoperability and composition of services*: One stop e-government (Wimmer and Tambouris, 2002) makes it possible for any public authority to act as a front office for delivering e-government services regardless of the administrations actually involved. Semantic Web Services enable interoperability of administration by automatic discovery and composition of appropriate web services. Consider, for example, the service "change of residence". In one-stop e-government the citizen can start the process (either at a portal or on the web site of his/her old

or new municipality) and the registration and deregistration services of the concerned municipalities have to be identified automatically. Exploring the semantic metadata is the initial step of that approach.
- *flexibility and adaptivity of process execution*: Web services are basic building blocks for building online processes. Semantic process models combined with inference rules and integrity constraints facilitate self-adaptivity and flexibility of process execution, allow for context-dependent resource allocation and ensure that constraints and guidelines are satisfied.
- *Maintenance and change of services*: As public services are based on legal rules and regulations, any change of these regulations can cause adaptation of processes. To ensure compliance of processes with regulations, the affected process and activities can be found easily if a law or regulation changes. This can be achieved by explicitly representing and inferring the reasons for design decisions.

In recent years ontologies have become common for semantically enriched formalization of knowledge. Chapter 2 provides a framework for modelling knowledge with ontologies in the e-government domain completed by examples of efforts already undertaken in our research projects covering all of the abovementioned challenges. Even though the e-government domain is a very important application area, our approach is not limited to it but can be used in any business domain. However, the proceeding introduced has been used in several e-government projects and proofed eligible.

19.2 State of the Art in E-Government Ontologies

Several European countries are currently working on a common service description aiming for a consistent presentation on portals run by the various service providers and establishing a basis for realising one stop e-government. With this approach e-government is no longer regarded as a portal where public administrations publish "passive" information but as an active source for knowledge and service sharing. In addition it is perceived that knowledge sharing is no longer limited to humans but include software agents. As ontologies are supposed to be representations of the agents' agreements about the set of concepts that underlie the information to be shared (Gruber, 1993b) they are an appropriate instrument of e-government knowledge representation (cp. (Gugliotta et al., 2006)).

In (Abecker et al. 1998) a methodology for organizational memories has been presented that distinguishes three ontologies (see Fig. 19.1):

- The *information ontology* describes the different kinds of information resources with their respective structure and format properties. The vocabulary for the information resource metamodels comes from the information ontology.
- The *enterprise ontology* defines the context in which information resources are used and generated. The top-level of the enterprise ontology defines a meta model for processes or organisational structure

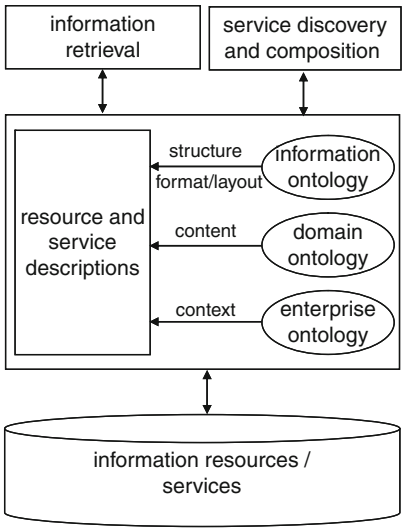


Fig. 19.1 Types of ontologies (Abecker et al. 1998)

– The *domain ontology* defines concepts modeling the content of information resources and services. It is obvious that in particular the domain ontology is specific for each new application area.

Throughout our lives, we all experience many events such as birth, marriage, building a house, retirement, and even losing one′s wallet. The "life" of a business also follows a series of predictable events such as the registration of a new business, relocation of a business or government forms and reports associated with various stages in a business' life. These events are called "life events". To help citizens and businesses deal with these kinds of events, most of the administration portals or web sites that have gathered information resources and services have a navigation structure that corresponds to these life events. In this sense, life events are organized as taxonomies – a simple form of ontologies (Daconta et al., 2003). Analysis of various portals and web sites has shown, however, that there is no consensus on what these life events should be called and structured.

In the SmartGov project a combined enterprise and domain ontology was built that provides a conceptual description of e-government services (Fraser et al., 2003). A representation in RDF is a cyclic graph and can become quite complex. This can cause problems both for public authority staff to maintain the ontology and also for user navigation. To avoid these problems in SmartGov the structure of the ontology was simplified in the form of a directed acyclic graph. This graph was extracted directly from the ontology, starting from a set of relevant top-level concepts that adequately describe public authority service provision. The top-level concepts are *activities, actors, issues, legislation, needs, process, requirements, responsibilities, results, rights* and *service types*. The subsequent structure has been engineered to enable searchability and function. Each concept has an optimal number of children,

which represent the domain as accurately as possible, while at the same time creating a logical search path to give an unambiguous route to the desired target (Fraser et al., 2003).

The Terregov project makes use of semantic technologies for achieving interoperability and integration between e-government systems. Rather than providing ontologies they implemented ontology creation and storage tools to allow ontologies to be created by domain experts (Wroe, 2005).

In the OntoGov[2] project we defined the top-level structure of e-government domain ontologies including life events and legal concepts as well as service and life-cycle ontologies (Stojanovic et al., 2006). The life event ontology was based on the life event structure of the national web site of Switzerland[3] (www.ch.ch). The idea, however, was not to define a standard ontology for life events but to provide a general structure that can be reused and adapted to specific applications and that enables information exchange and interoperation between web services of different service providers (for problems associated with reusing the life event ontology see Section 19.3.3).

As can be seen, there is no standard domain ontology – even for life events a consensus will barely be enforceable. Potentially, for every application a new ontology has to be built and maintained.[4]

19.3 Ontologies to Formalize a Shared Understanding of Meaning

Even though there is an agreed definition of an ontology as a "formal specification of a shared conceptualisation", as stated by Gruber (1993a) and refined by Borst (1997), the meaning of "conceptualisation" can be regarded differently (cp. (Pinto and Martins, 2004)). Here we follow the view of Pinto and Martins where conceptualisation is regarded as "an abstract conceptual model for the knowledge to be represented in the ontology" (Pinto and Martins, 2004). To bridge the gap between knowledge, phrased ambiguously in natural language (e.g. in regulations) and its unambiguous representation in an ontology we introduce the intermediate stage of semi-formalization.

Since the early 1990s there has been a lot of research on ontology design and creation (amongst others Gruber (1993b), CommonKADS (Schreiber et al., 1999) or Ontology Development 101 (Noy and McGuinness, 2002)). However, no standard methodology for ontology building exists.

---

[2] OntoGov (Ontology Enabled E-Gov Service Configuration) is a project funded in the IST Programme of the European Union (IST-2004-27090). For further information consult http://www.ontogov.com/

[3] The Swiss Federal Chancellery was a partner in the OntoGov consortium

[4] Application here is understood as any kind of software developed to execute public services (using semantic technologies)

As ontology development is part of an IT project, it has to be embedded into a project framework. We suggest using the waterfall model as it is widely accepted for IT projects in the public sector (e.g. HERMES, which follows the waterfall model, is the stipulated IT project management method in Switzerland (FSUIT, 2004)). Figure 19.2 depicts our ontology development process.

In the first phase, the pilot study, the purpose and goal of the project should be investigated with respect to the business strategy. The various possibilities of knowledge formalization should be considered to ascertain that ontologies are the appropriate model. This is variously called "specification" (Pinto and Martins, 2004) or "capture motivating scenarios" in TOVE (Gruninger and Fox, 1995) or "identify purpose and scope" in ENTERPRISE (Uschold, 1996, Uschold et al., 1998) or "requirement specification" in METHONTOLOGY (Fernández et al., 1997).

Within the concept phase ontology development is performed comprising three levels of formalisation: informal (knowledge is captured in natural language), semi-formal (knowledge is represented in a semi-formal way e.g. in structured templates) and formal (knowledge is strictly formalized in OWL,[5] SWRL[6] and OWL-S[7]). Therefore ontology conceptualisation, formalization and implementation are conducted in this phase.

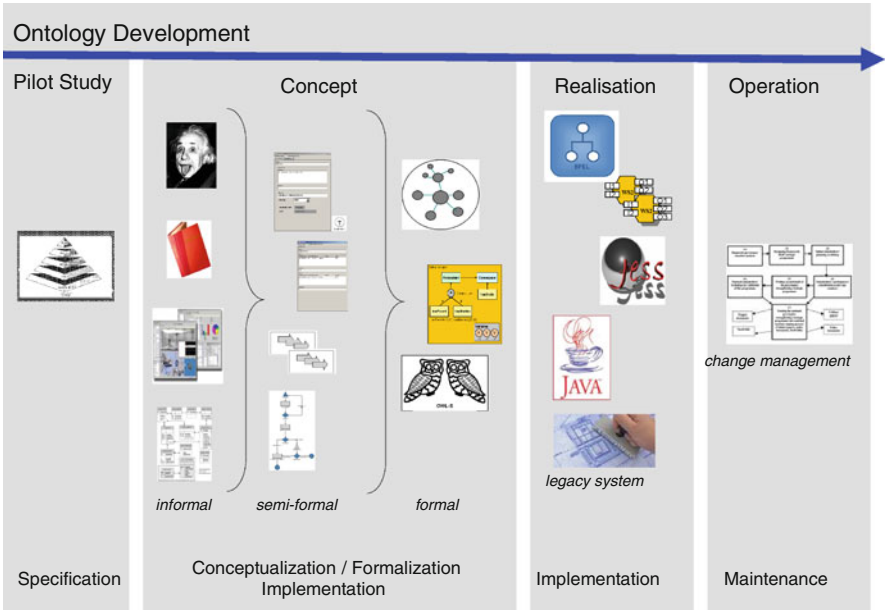


Fig. 19.2 Ontology lifecycle

[5]OWL: Web Ontology Language (McGuiness and vanHarmelen, 2004)
[6]SWRL: Semantic Web Rule Language (Horrocks et al., 2004)
[7]OWL-S: Web Ontology Language for Services (Martin, 2004)

The realisation phase is about implementation of ontologies, that is making ontologies accessible out of program code (e.g. legacy systems or web services). This can be achieved by migrating it into Java code, thus implementing application programming interfaces (APIs) to access the ontology via ontology management systems like Protégé[8]

Maintenance is performed in phased operations comprising ontology management like updates (add or delete concepts or properties), merging or integration of ontologies (q.v. Section 19.3.3 of this chapter).

In the following we focus on the concept phase with its three levels of formalization (informal, semi-formal and formal). We need the formal level to make ontologies machine understandable. In most cases, however, a domain expert will not be able to transform informally described knowledge (e.g. guidelines written in natural language) into a formal representation like OWL, using Protégé as ontology management tool. To bridge the gap we introduce the semi-formal level where domain knowledge, services and rules are formalized in a way domain experts are able to phrase and IT staff is able to transform without missing or misinterpreting business logic.

19.3.1 Starting with Terms

In general there are two ways to build ontologies: either from scratch or by reusing existing ones. Gugliotta (2006) gives a detailed overview of current (research) activities in this field. Please refer amongst others to Peristeras and Tarabanis (2006) for "top-level reusable models for the overall e-government domain".

In the document at hand we talk about ontology development from scratch but starting not immediately with concept modelling but with *term*[9] *definition*. What Oscar Chappel stated for rule modelling can be adapted for ontology modelling: "everything starts with terms" (Chappel, 2006), that are candidates for main concepts. Sources for terms can be "anything" that reflects business behaviour:

- documents
  - lists of services and their descriptions (eCH, 2006)
  - theme-catalogues (e.g. lifecycle aspects, business situations (eCH, 2008)
  - handbooks
  - guidelines
  - policy papers
  - glossaries

[8]Protégé is a free, open source ontology editor and knowledge-base framework (http://protege.stanford.edu).

[9]Merriam Webster Online: Definition 4a: a *term* is a word or expression that has a precise meaning in some uses or is peculiar to a science, art, profession, or subject <legal *terms*>, URL: http://www.m-w.com/cgi-bin/dictionary

– data
  - meta-data, descriptors, identifiers
  - key-words for search engines
  - database entity and attribute names

– mind
  - expressions passed on orally.

When questing for terms, the goal and scope of the ontology to be built should be kept in mind. Not every term written in a handbook should become a concept candidate even though every main concept supposed to be in the ontology should be explicitly represented.[10] Therefore *relevance* is an eligibility criterion for ontology inclusion as the list of terms could quickly become complex.

Within the public sector – especially in federal states – finding and defining relevant terms is even more complex as there is no "authorized decision maker". Several terms will be used for the same circumstances and one term will be used to describe different facts. As we will see later, ontologies provide a good solution for handling synonyms and homonyms without offending sensibilities. In the term building phase the various terms will be collected, their relation or specific meaning identified and documented.

There are several approaches for automated ontology creation (amongst others TextToOnto,[11] supporting semi-automatic creation of ontologies by applying text mining algorithms (Maedche and Staab, 2004)). This could be a good starting point for *term capturing* as ontology building is time consuming. As ontologies are complexity prone, ontology creation can be machine supported. However, a lot of manual labour is still necessary.[12]

Regardless of which approach is chosen the following attributes[13] should be defined for every term (they will become data properties in the ontology):

– Manually defined attributes
  - label (the human-readable label including language information)
  - definition (a statement that represents the concept and essential nature of the term)
  - source (where the term has been taken from, e.g. out of a handbook)[14]

[10] *completeness* is a criteria very difficult to prove (cp. (Gómez-Pérez, 2001))

[11] The system is freely available and can be downloaded at http://sourceforge.net/projects/texttoonto/

[12] An interesting topic for research is how term and fact modeling could be automated (as it is less formal) and how the semi-formal representation could then be used as input for automated ontology creation

[13] The attributes "label", "definition" and "date issued" are attributes, "source" and "creator" are defined as meta-data terms by the Dublin Core Metadata Initiative (2006)

[14] In case the terms are extracted automatically this attributes can be created automatically, too

– Automatically created attributes

  – date issued
  – creator (e.g. domain expert who records the term).

The result of this step is a collection of terms with their attributes. Again, there are several methods of presenting the terms (as well as concepts), either as mind-maps (as introduced by Sure et al. (2002)) or simply as a (flat) list – where appropriate in alphabetical order.

We propose a repository where concepts can be stored in a structured but semi-formal way understandable by IT staff *and* by domain experts. An example of a structured term representation is given in Fig. 19.3[15] for the term "Application". A term is represented graphically as a named circle (labelled with a "T" for term);

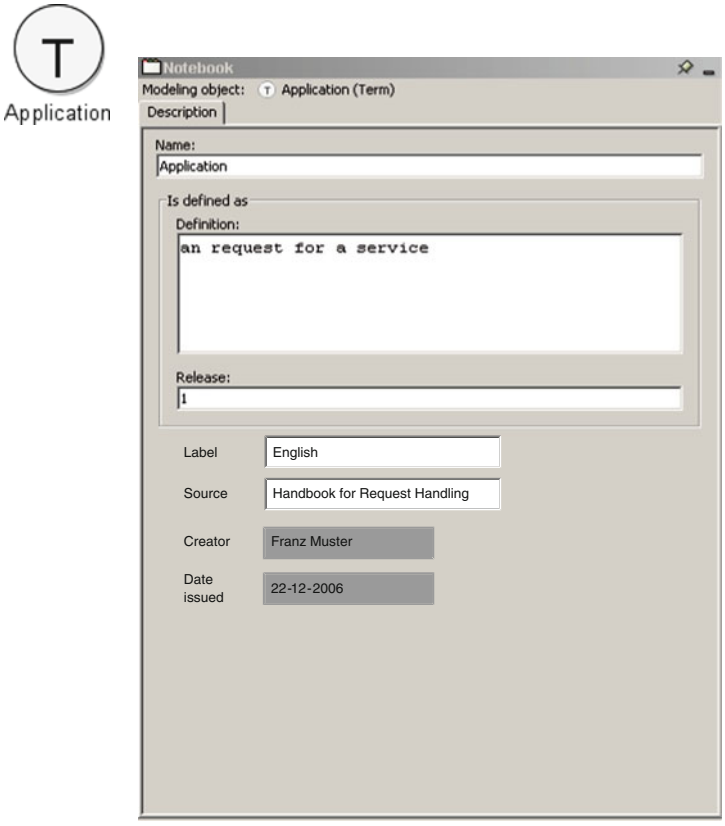


Fig. 19.3 A concept and its properties

[15]The interface is adapted from the FIT Buildtime for Adaptive Processes developed by BOC Asset Management (http://www.boc-eu.com) within the FIT project.

its attributes are captured in a so-called "notebook". Several user workshops (performed within the FIT project[16]) have shown that this simple formalization is much easier to understand than using an ontology modelling tool like Protégé from the very beginning. As Macias and Castells (2004) pointed out: "Generally speaking, emerging web-based technologies are mostly intended for professional developers. They pay poor attention to users who have no programming abilities but need to customize software applications".

After capturing the terms (along with their attributes) they can be grouped, initially by considering relationships between terms. Figure 19.4 shows a grouping of terms based on their graphical representation. Business people group terms along business aspects: life cycle events (leading to the group of "Environment&Construction"), project management issues (leading to the group of "ProjectAspects"), elements of business tasks (leading to the group of "Application Handling") or stakeholder (leading to the group of "People"). Even though business people are not familiar with relating concepts, the two different kinds of groups in the example are easy to understand:

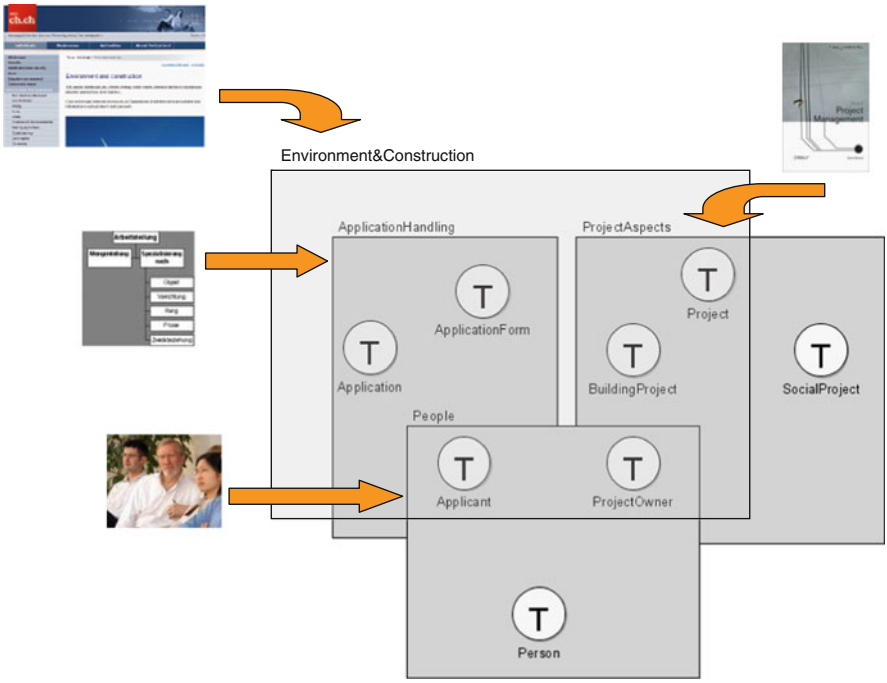


Fig. 19.4 Visual grouping of concepts

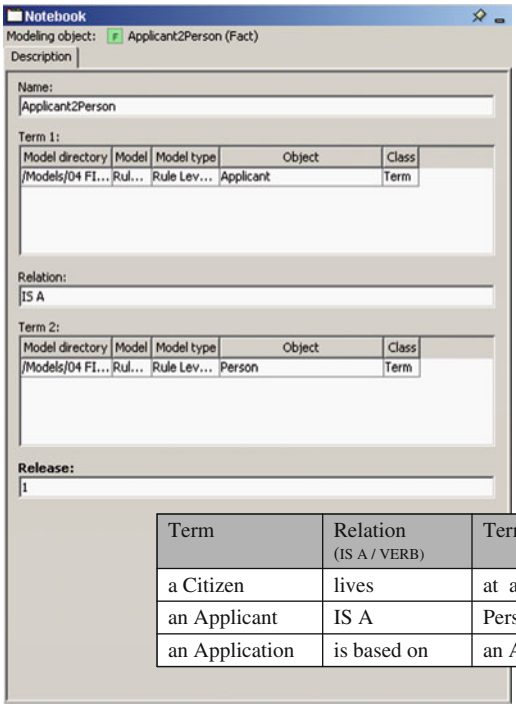
[16]FIT (Fostering self-adaptive e-government service improvement using semantic technologies) is a project funded in the IST programme by the European Union (IST-2004-27090). For further information consult http://www.fit-project.org

(a) terms and specialisations of terms as in the *ProjectAspects group* (where the terms "BuildingProject" and "SocialProject" are specialisations of the term "Project") and
(b) terms related because of a specific business topic as the life event "Environment&Construction" (where the terms "Application", "Application Form", "Applicant", "ProjectOwner", "Project" and "BuildingProject" belong together).

Having grouped the terms, the next step is to make the relations explicit. Again we recommend a semi-formal way of building *facts* (Fig.19.5) that is, relating terms either in an hierarchical way (a "BuildingProject" *IS A* "Project") or as a network (a "Application" *is based on* an "ApplicationForm", a "ApplicationForm" *is the same as* a "RequestForm" etc.) The result of this step is a semi-formal *term model.*

Modelling facts is the last step in ontology building that can be done in such a convenient way for domain experts. Adding more properties, e.g. to express constraints for domain or range restrictions, would quickly make the forms complex and would lead to a kind of "ontology modelling simulation".

It turned out that collecting terms and modelling facts is an iterative process. A good possibility is to start with terms extracted from standards (for example the



Term	Relation (IS A / VERB)	Term
a Citizen	lives	at a Place of Residence
an Applicant	IS A	Person
an Application	is based on	an Application Form

Fig. 19.5 Relating terms to build facts

documents published by eCH for Switzerland). Having consensus on that specific terms invented by communities can be considered and incorporated.

19.3.2 Transforming Terms and Facts to Concepts and Properties

To transform terms and facts to concepts and properties we suggest that IT staff or knowledge engineers, who are familiar with ontology modelling, take the semi-formal models of terms and facts as input for formal ontology modelling, e.g. using a specific ontology modelling tool like Protégé. While the transition from terms to concepts (attributes to data-properties) and from facts to relations is currently performed manually, automation is planned.

During ontology modelling concepts may be added (e.g. to build a super-concept), data-properties will be completed (e.g. for cardinality) and properties will be extended or newly set up (e.g. relations between terms reflecting business aspects, domain and range constraints or simply stating synonymy).

Therefore the ontology model will probably differ from the term model. To ensure consistency between the ontology model and the term model "reverse engineering" is recommended. Transforming the ontology model "back" to the term model allows business people to continue with modelling business from their point of view. Using OWL as an interchange format,[17] BOC's modelling tool, for example, allows the import of ontologies and its representation as a term model.

As in the majority of cases public administration officers will not model the ontology on their own, a graphical presentation will make the cooperative development by business analysts or IT staff and domain experts less difficult (Fig. 19.6).[18]

19.3.3 Negotiating Reuse

It cannot be expected that one e-government ontology can be built for the entire public administration of a state let alone across borders. More likely, several ontologies will be available and could be joined, if reasonable, in two ways: fusion/merging or composition/integration. A lot of research has been done in the field of ontology reuse (cp. amongst others Pinto and Martins, 2002; Kalfoglou and Schorlemmer, 2005) and several tools have been developed to support the process.[19]

It is beyond the scope of this document to go deeper into that topic. However, as a rule of thumb it can be said, that merging ontologies is preferred if there

[17] An interchange format is a format that allows transformationen from one model to another without loss based on agreed standards.

[18] For ontology modeling Protégé is taken; for graphical presentation the Ontoviz tab is activated

[19] A comprehensive overview is given by the IOWA State University, Departement of Computer Science: http://www.cs.iastate.edu/~baojie/acad/reference/2003-07-09_dataint.htm (Date 20-12-06)

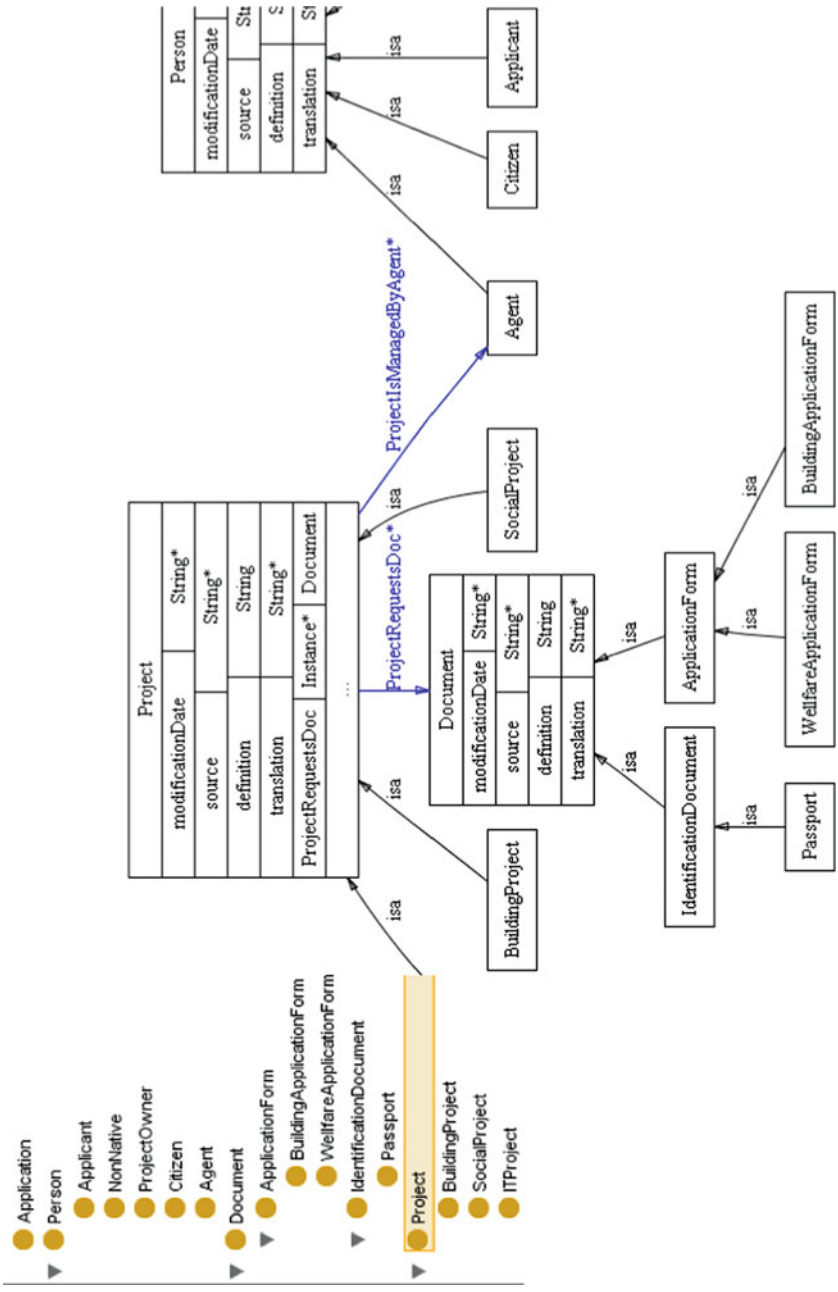


Fig. 19.6 Graphical presentation of concepts and properties

is a significant overlapping of concepts whereas integration is favoured if the overlapping is restricted.

Regarding the e-government domain, reuse of ontologies seems to be a promising approach. For example the "life event concept" is implemented on most governmental web sites worldwide. As already mentioned in Section 19.2, in the OntoGov project an ontology has been built based on the "life event concept" of the national web site of Switzerland (http://www.ch.ch). It was assumed that the ontology could be reused in any municipality in Switzerland or even in other countries. When verifying this intention, however, significant differences have been detected with respect to

- point of view (what has been defined as a life event concept in the ontology has not been regarded as one in an municipality and vice versa)
- structure (what has been modelled as super-concept in the ontology has been regarded as sub-topic in an municipality and vice versa)
- granularity (in some municipalities the life event structure is very detailed where as in others it is not).

Reusing/joining ontologies is in any case not only a technical demand but requires expertise (by domain not IT experts) to negotiate the goal, scope and content of the new ontology. Public administrations move in the direction of knowledge sharing (and reuse) but federalism is still pronounced hindering the possibilities the technique already provides.

From this point of view integration seems to be the better approach for reuse. However the mentioned problems cannot be overcome simply and further research is needed for an applicable solution.

19.4 Ontologies for Modelling Semantically Enriched Processes

In Section 19.3 of this chapter we introduced a method for getting to (main) concepts and relations of a domain ontology. In the following section we will focus on metadata related to the business processes.

Although municipalities provide virtually identical services, implementation of these services takes place individually and is continually repeated.[20] For example: nearly every municipality in Switzerland (about 2,700) performs the service of registration (for someone moving in) and deregistration (for someone moving away). Even though there are efforts to standardize the processes, the real implementation differs significantly, due to a range of factors including the different IT infrastructure of the municipalities. The same is true for services of federal states and administrations like courts.

[20]For example: the public service "Anmelden/Abmelden" is performed by ne

The exchange of experience and sample process models between similar institutions could drastically reduce the effort of service implementation. For example, in Germany the "Virtual Community Geschäftsprozess-Management[21]" (virtual community for business process management) is based on a repository of process models. Members can provide experience in the form of process models and profit from the experience of other members.

A similar approach is the propagation of reference models for e-government services and organisational structures that can be used and customized directly by municipalities and federal states. Reference models are abstract models of a domain of interest that represent best practices. As administrative organisations – as opposed to companies – do not have to protect a competitive advantage and their processes are based on legal foundations, e-government services are well suited for reference models.

Nowadays, repositories of (reference) models mainly use textual descriptions; thus, they are hard to quest and only indirectly reusable. Adding semantic metadata to process models would be much more helpful in finding adequate services and reusing the *models* (instead of their description) – an approach that is widely accepted for semantic web services. This, however, is still an open research issue, in particular as no standards for e-government ontologies exist.

Even without reference models, business process management in general must cope with the problem of life-cycle management, i.e. how to ensure compliance of models with regulations and how to customize process models. Adding not only metadata to a process but also information about design decisions is an approach for dealing with the problem.

Virtually all public administration processes are based on legal foundations or decrees that are derived from them. Therefore, such regulations also determine the design of a process. If a law is modified, all processes that are based on this law have to be checked for compliance. Besides legal aspects, organizational or technical reasons can also determine the design of a process. If one administration wants to reuse and customize a process from another administration it can be important to know what is specific to the administration and thus can be modified and what is vital for the process to work.

In Hinkelmann et al. (2006a) we presented an approach for life-cycle management of e-government services. To ensure consistency of services, all the decisions are documented. For each design decision at least one reason must be given. A decision includes a description in natural language and a formal reference to the elements forming the basis of the decision (e.g. the respective law). Analogous to IBIS (Issue Based Information System, (Kunz and Rittel, 1970), a design decision is regarded as a topic which is based on a reason. As a design decision can possibly lead to other design decisions, a line of argumentation results. Figure 19.7 shows the structure of a line of argumentation according to the graphic notation gIBIS (Conklin & and Begeman, 1998, pp. 140–152).

[21] http://www.fhvr-berlin.de/vc-gpm/ (information in German only).

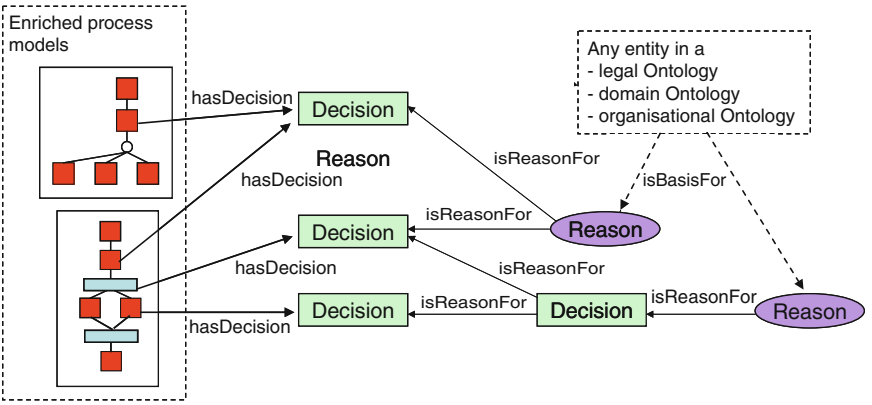


Fig. 19.7 Extending process models with life-cycle information

Apart from the natural language documentation, it is vital for knowledge-based support to also formally describe the design decisions and reasons. This is done by linking design decisions and reasons using an ontology in which the reasons for the design decisions are explicitly modelled and refer to the underlying law. This ontology is called "lifecycle ontology" as all activities performed on process models are

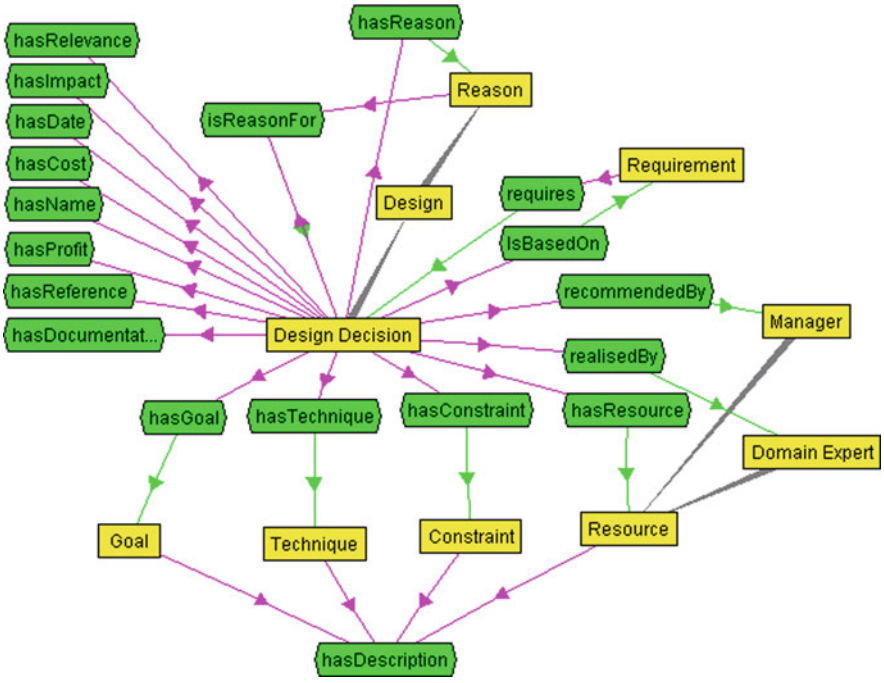


Fig. 19.8 Part of the OntoGov lifecycle ontology (developed using KAON)

captured, e.g. when adding a new activity to a process the reason must be expressed and the respective decision is saved along with further informations like user or creation date. Figure 19.8 shows the relevant part of the lifecycle ontology. The rectangles represent concepts and the hexagons represent relations. It can be seen that DesignDecision is a subconcept of Decision and that Decision is a subconcept of Reason, because every decision can have consequences, and so can be a reason for another decision. In this way we can model the decision-reason chains shown in Fig. 19.7.

In addition, extended process models are formally linked with the design decisions of the lifecycle ontology (see Fig. 19.7). In this way, design decisions and reasons that define process (or reference) models become transparent and traceable.

Details on how the design decision can be used for change propagation can be found in Hinkelmann et al. (2006a).

19.5 Ontologies for Modelling Business Rules

As already shown, public services are based on legal rules and regulations binding for all municipalities but at the same time municipalities may maintain distinctive features, so the business processes are quite *similar* but not *identical.*[22] Additionally, a lag time can occur between adoption and implementation of rules, so delays and human errors can arise.

Second, e-government services are knowledge-intensive processes resulting in manually performed process execution. Even though ICT supports most of the *functions* (e.g. tax system) very few *tasks* are automated. In addition, it is not only the knowledge about how to perform *the tasks* within a process (the so called functional knowledge) but also the knowledge about *the process flow* (the so called process knowledge) that isn't explicitly documented.

So the next step in enhancing business processes is adding business rules. Business rules can be regarded as an appropriate approach not only to make that knowledge explicit, to have greater control and oversight, but also to ensure consistent implementation of policies in an automated way, because changes to the rules can be immediately reflected in all services related to them.

19.5.1 Business Rules Classification

The Business Rules Group defines a "Business Rule" as:

> ...a statement that defines or constrains some aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business" (Business Rules Group, 2006).

[22] The terms *service* and *process* differ in their coverage: whereas *service* comprises all aspects of e-government service provision a Public Administration has to offer, *process* is about the (IT-supported) tasks performed within a *service.*

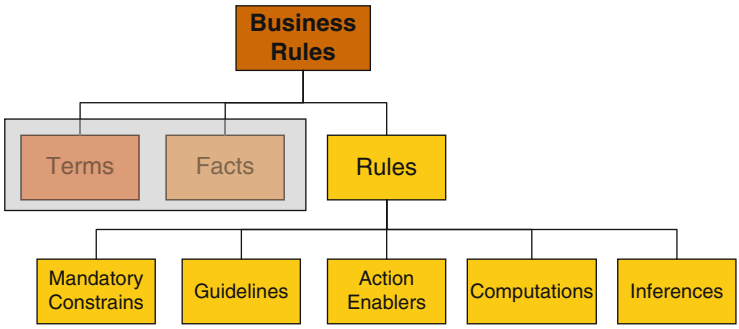


Fig. 19.9 Classification scheme after Barbara von Halle (2002)

There are several classification schemata for Business Rules formalization. Since it is well understood and comprehensive, we follow the classification scheme given by von Halle (2002, p. 27 ff.) in order to explain the different kinds of business rules.

Business rules consist of terms, facts and rules. "The terms and facts are the semantics behind the rules. They will also become the foundation for a logical data model and physical database [. . .]" (von Halle 2002, p. 32). In Section 19.3 we already explained how terms and facts can be identified and and transformed to concepts and properties (Fig. 19.9). Rules are split up into five sub-classes:

– mandatory constraints
– guidelines
– action enablers
– computation rules
– inference rules

In the following discussion we explain these rule types in more detail. Similar to the approach for determining terms and facts we follow a two-step approach starting with a semi-formal representation that can easily be understood by people in public administrations and that can later be formalized in cooperation with IT people and knowledge engineers.

19.5.2 Semi-Formal Rule Respresentation

Semi-formal representation is the starting point for rule development. Each rule type has a specific structure or uses predefined relations. The rules should be phrased using previously defined terms and facts (see Section 19.3). It can also be the case, however, that new terms and facts are identified while formulating the rules thus leading to an extension of the ontology.

Mandatory Constraints

Mandatory Constraints are statements which must always be applied. To express this kind of rule the auxiliary verb "must" is used as a predicate. The phrase "must not" is used to express negative constraints.

Subject	Predicate	Object
a resident	MUST provide	a lease contract or contract of purchase for her place of residence
a decision	MUST BE in list	application decision ('applicaiton approved', 'application denied', 'decision postponed')

Guidelines

Guidelines are rules which could be followed but are not mandatory. Their form is similar to mandatory constraints but using the auxiliary verb "should" or "should not" to express the negation

Subject	Predicate	Object
a confirmation	SHOULD BE sent	within 5 days
a decision	SHOULD include	an explanation

Action Enablers

Action Enablers initiate a process (step), if the condition holds.

	Term	Condition		Action
IF	a building	is closer than 50 meters to a natural water	DO	approve environmental compatibility
	a building	is under protection of historical heritage		historical preservation agency approves application

Computations

Computations are rules to perform calculations, like sum and difference.

Subject		Computation
fee	IS COMPUTED AS	days of delay x 10 CHF

Inferences

Inferences are statements which establish the truth of a new fact, if the condition holds.

	Term	Condition		Consequence
IF	an application a building	is delayed is older than 100 years	THEN	a fee is payable it is an historical building

19.5.3 Formalization

To automatically apply rules, a formal, executable rule language is required for all kinds of business rules. There is a broad spectrum of formalisms with different levels of expressiveness. Terms and facts, for instance, can be represented as vocabulary, database schema or ontologies. In Chapter 3 we introduced modelling terms and facts as concepts and properties. To formalize rules it makes sense to use the ontology as knowledge base. In this section we describe how to express business rules using ontologies.

19.5.3.1 Property Restriction

Some mandatory constraints can be expressed by constraining the range of a property (value restrictions) or the number of values a property can take (cardinality restrictions).

Subject	Predicate	Object
a decision	MUST BE in list	application decision (‘applicaiton approved’, ‘application denied’, ‘decision postponed’)

This sample rule can be expressed by using a value restriction. The concept “decision” provides the property “hasApplicationDecision” which contains a list of possible values (Fig. 19.10).

The following document fragment expresses the same restriction in OWL:

```
<owl:Class rdf:ID="decision">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:allValuesFrom>
          <owl:Class>
            <owl:oneOf rdf:parseType="Collection">
```

```
            <applicationDecision rdf:ID="Application_denied"/>
            <applicationDecision rdf:ID="Application_approved"/>
            <applicationDecision rdf:ID="Decision_postponed"/>
          </owl:oneOf>
        </owl:Class>
      </owl:allValuesFrom>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="hasApplicationDecision"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
    <!-abbreviated -->
</owl:Class>
```

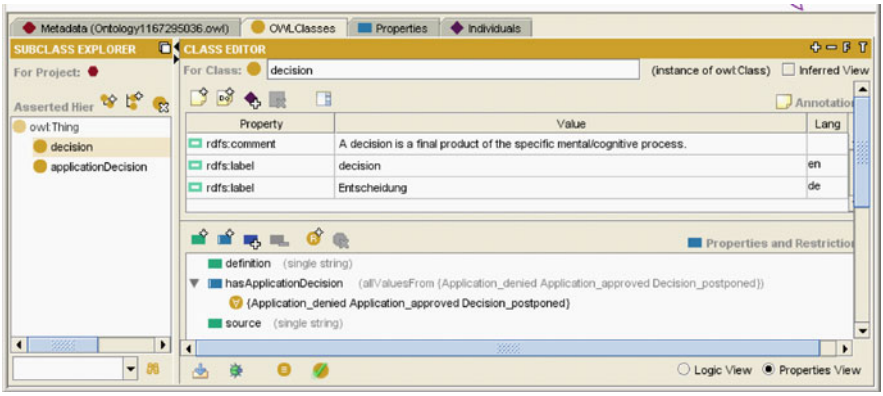


Fig. 19.10 Value restriction modelled in Protègè

19.5.3.2 Semantic Web Rule Language

Restrictions cannot be used to express every kind of constraint; in particular they cannot be used to express inferences, computations and guidelines. Therefore we use the Semantic Web Rule Language (SWRL) which has been published as a Member Submission by W3C. The aim of SWRL is to combine the Ontology Web Language OWL DL and OWL Lite with the Datalog RuleML sublanguage (Horrocks et al., 2004). It thus extends the set of OWL axioms to include Horn-like rules and enables Horn-like rules to be combined with an OWL knowledge base.

The proposed rules are of the form of an implication between an antecedent (body) and consequent (head). Whenever the conditions specified in the antecedent hold, the consequent must be true. Antecedent and consequent consist of zero or more atoms. Atoms in these rules can be of the form C(x), P(x,y), sameAs(x,y) or

differentFrom(x,y), where C is an OWL concept, P is an OWL property, and x,y are either variables, OWL individuals or OWL data values (Horrocks et al., 2004).

Inferences

Inferences can be expressed using SWRL. If a condition holds, then the fact in the consequence must hold, too. In ontologies e.g. a value of a property is derived.

	Term	Condition		Consequence
IF	a building	is older than 100 years	THEN	it is an historical building

To express the inference written above the term "a building" can be presented as a concept, which contains the datatype property "age". To present the consequence, a concept called "ancientBuilding" has to be added, which can be modeled as a sub-concept of "building".

```
building(?x)  ∧
age(?x, ?y) ∧
swrlb:greaterThan(?y, 100)
  →  historicalBuilding(?x)
```

If the condition "an instance of building exists, whose age is greater than 100" holds, the found instance is also an instance of "ancientBuilding".

Mandatory Constraints

Mandatory constraints which cannot be represented as property-restrictions can be expressed by using SWRL. Consider for example the mandatory constraint below.

Subject	Predicate	Object
a person	MUST be	Older than 18 to make an application

Several concepts have to be created to be able to present the rule. First the two terms "person" and "application" have to be modeled as concepts. To present the fact "an application is filled out by a person" the object property "filledOutByPerson" is defined with the domain "application" and the range "person".

The constraint can be expressed by a rule using negation stating that there is an inconsistency if the age of an application is not 18 or older. Thus, we use

the concepts “exception”, “ontology” and “ontologyStatus”. The “ontologyStatus” provides the datatype property “inconsistent”. To have a hint about the reason for the inconsistency an object property “hasException” links from “ontology” to “exception”.

```
application(?x)  ∧
filledOutByPerson(?x, ?y) ∧
age(?y, ?z)  ∧
swrlb:lessThan(?z, 18) ∧
ontology(?o)
  →  hasException(?o, ExceptionPersonMustBeOlderThan18)
```

The SWRL rule above expresses that the ontology has the exception “ExceptionPersonMustBeOlderThan18”, if an application is filled out by a person who is younger than 18.

The atom “application(?x)” holds, if an instance of the concept “application” exists. The atom “filledOutByPerson” holds, if the instance of “application” is related to an instance of the concept “person” by property “filledOutByPerson”. If the found instance of “person” is of “age” ?z, this atom holds. The built-in relation “swrlb: lessThan” holds, if ?z is less than 18. If an instance of “ontology” is found, the condition of the last atom is fulfilled, so the whole condition is true. The consequence is that the instance “ExceptionPersonMustBeOlderThan18” of the concept “exception” is linked to the ontology by the object property “has Exception”.

If a constraint is violated the status of the ontology is inconsistent. It can be expressed by following rule:

```
ontology(?x)  ∧
hasException(?x, ?y)  ∧
ontologyStatus(?z) →
inconsistent(?z,true)
```

Guidelines

As with the constraints, guidelines can also be represented as logical rules, using the reserved property “warning” instead of inconsistent and the concept “guideline” to have a hint about the reason for the warning.

Subject	Predicate	Object
A confirmation	SHOULD BE sent	within 5 days

To express the guideline below the term “confirmation” should be represented as a concept. It contains the datatype-property “sent”.

```
confirmation(?x)  ∧
sent(?x, ?y)  ∧
swrlb:greaterThan(?y, 5)  ∧
ontology(?z)
  → violatedGuideline (?z, ConfirmationShouldBeSentWithin5Days ) ∧
     accept(ConfirmationShouldBeSentWithin5Days,false)
```

The guideline has the datatype-property “accept”, so that an officer can accept the violation. To express that the status of the ontology is “warning”, the rule can be expressed as follows:

```
ontologyStatus(?x)  ∧
ontology(?y)  ∧
violatedGuideline(?y, ?z)  ∧
accept(?z, false)
  →  warning(?y, true)
```

Action Enablers

Action enabling rules trigger planning and refinements of a process from predefined activities like the rule below.

	Term	Condition		Action
IF	a building	is closer than 50 meters to a natural water	DO	approve environmental compatibility

Based on the context of the actual service execution, action enabling rules associated to particular process steps are invoked at runtime to dynamically determine

and initiate the appropriate actions. Therefore, every process must be modeled as an OWL-S process. To express the sample action enabling rule above, an atomic process[23] called "ApproveEnvironmentalCompatibility" must be executed.

To present the condition of the rule an instance of the class "SWRL-Condition" of the OWL-S-Ontology must be created and the following condition must be entered at the property "expressionObject".

```
building (?x)    ∧
distanceToNaturalWater (?x, ?y)  ∧
swrlb:lessThan(?y, 50)
```

To express the consequence, an instance of the class "If-Then-Else", which is subclass of "ControlConstruct" has to be created, which object-property "ifCondition" link to the condition and the property "then" link to the atomic process.

Computations

The statement provides an algorithm to compute the value of a term.

Subject		Computation
Fee	IS COMPUTED AS	days of delay x 10 CHF

To represents the computational rule, a concept "BuildingApplication" is used. This concept contains the two datatype properties "daysOfDelay" and "fee". These two properties are used to compute the product.

```
BuildingApplication(?x) ∧
daysOfDelay(?x, ?y)
-> swrlb:multiply(?fee, ?y, 10) ∧
fee(?x, ?fee)
```

[23] A process model is composed of atomic processes and/or composite processes. An atomic process is defined as a non-decomposable process, e.g. in a process implementation with web-services it can be executed using a single call.

For every instance of “BuildingApplication” which has a delay, the fee is computed as the value of “daysOfDelay” multiplied by 10. Similar to multiplication, built-ins for subtraction, addition and division can be used.

Negation

A problem exists in expressing negations. In the above examples negation is expressed by an inverse predicate “lt” (less than) instead of “NOT greater than”. However, in the current version of SWRL such inverse predicates are not available in every case. For example, the rule “If the construction plan is not available, do request documents.” cannot be represented directly in SWRL. This constraint can be expressed by a rule that stated that there is an inconsistency. This kind of negation is called negation as failure: it is true if no construction plan is mentioned in the knowledge base.

Rule Set

Every rule expressed in SWRL can be combined into rule sets. For example, all rules which are defined for a process to evaluate an application form can be combined in a rule set called “FormEvaluationRules”. Our approach is to use the following three concepts to express rule sets: “Rule”, “RuleSet” and “File” (Fig. 19.11).

The concept “Rule” provides the datatype properties “RuleID”, “RuleName” and “RuleDescription” to store the name, id and description of a rule. The SWRL-file itself will be stored in an external file. To retrieve the file, the instance of ‘File’ contains the datatype property “filePath”. To have a relation between a file and the expressed rule, the ontology provides the object properties “containsRule” and “isStoredIn” respectively. All rules could be combined into rulesets by creating an instance of “RuleSet” and using the object property “comprisesRule” to link to the rules. One benefit of not storing rules directly in a rule set is that a rule can be combined to multipe rule sets.

To have a link between processes and rule set, the concept “process” of the OWL-S ontology has to be extended by the object property “containsRuleSet”, the range of which links to “RuleSet”.

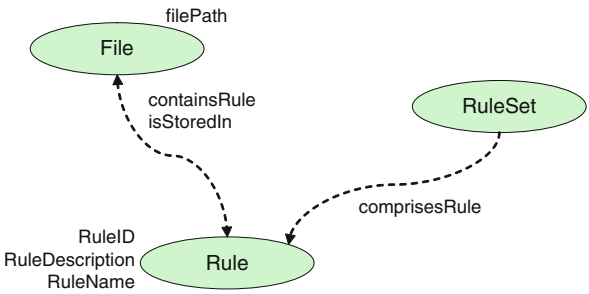


Fig. 19.11 Rule ontology

19.6 Ontologies for Modelling Agile E-Government Processes[24]

Although there are legally binding rules and regulations every administration has to obey, dealing with people's concerns means dealing with different circumstances every time. In this sense, e-government services are often knowledge intensive processes, where the actual process execution and the involved participants and administrations depend on various factors. Consider for example the process of building permission in a European municipality. Under particular circumstances an environmental compatibility check is required or the historical preservation agency has to be involved, in which case complex clarifications have to be made. Modelling all possible variants of a process can lead to complex process models. Sometimes this modelling can even be impossible if the tasks are mutually dependent on one another.

To deal with these kind of agile processes, the Agile Process Management (APM) approach combines business and knowledge work processes by linking process models and business rules (Hinkelmann et al., 2006b). The business rules extend process execution on three ways:

- *Variable process execution*: Determine activities and processes to be executed thereby accounting for dependencies between activities
- *Intelligent resource allocation at run time*: Selection of employees based on special skills and selection of particular web services adequate for the actual circumstances
- *Intelligent branching and decision making*: deriving workflow-relevant data using inferences and computing values
- *Consistency checking*: Avoid violation of integrity constraints and guidelines

Agile processes management includes all of the previous modelling approaches to provide adaptable, flexible and reusable e-government services: *Ontologies* build the basis for modelling and executing *semantically enriched processes* and *business rules*.

At run time action-enabling rules select the activities that have to be executed depending on the actual context of the process instance. Inference rules allow for resource allocation and support the user in decision making, while integrity constraints and guildelines (in combination with inference rules) ensure consistency checking and compliance.

Figure 19.12 shows the three aspects of ontology modelling and use:

- domain modelling to make business knowledge explicit,
- business rules modelling to make rules and regulations explicit,

[24] A process is considered "agile" when its execution model is created flexible at runtime, based on the results of triggered rules instead of static pre-defined models.

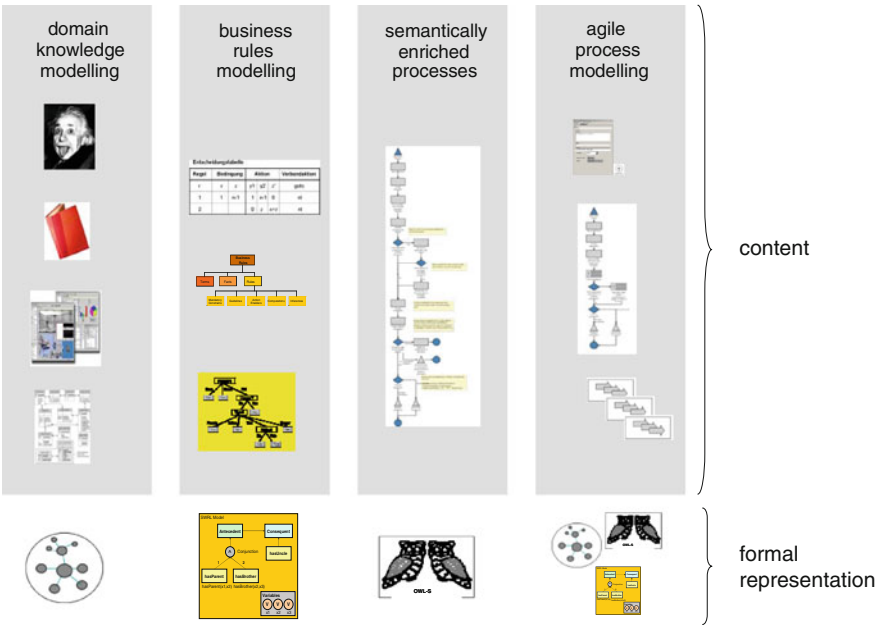


Fig. 19.12 Modelling agile services with ontologies, rules and semantically enriched process models

– process (life-cycle) metadata to make design decisions explicit and allow for change management.

The far right of the figure depicts the agile process modelling, combining ontologies with rules. Let us explain the agile process management framework with an example:

The first phase of designing agile processes is sketching a "process skeleton" that means the main activities (atomic processes) of a (composite) process. Figure 19.13 depicts a skeleton of a process model for building permission where parts are marked as knowledge intensive tasks. As already mentioned, depending on particular circumstances different activities are to be executed during the process, e.g. checking for environmental compatibility or historical preservation. The conditions for these activities are not always given at the beginning but may appear only after additional clarifications are made. Thus, we have an unforeseeable sequence of data collections, clarifications and decision making which cannot be modelled exactly but must be determined at run-time. Decision making, however, may require specific skills, in which case experts with relevant experience should be allocated to perform the task. Since laws and regulations in this area are quite complex, a lot of guidelines and constraints have to be considered.

Thus, the knowledge-intensive parts in the process can than be regarded as agile, containing variable processes, intelligent branching and flexible resource allocation. To cope with this agility, we extended process modelling in two ways:

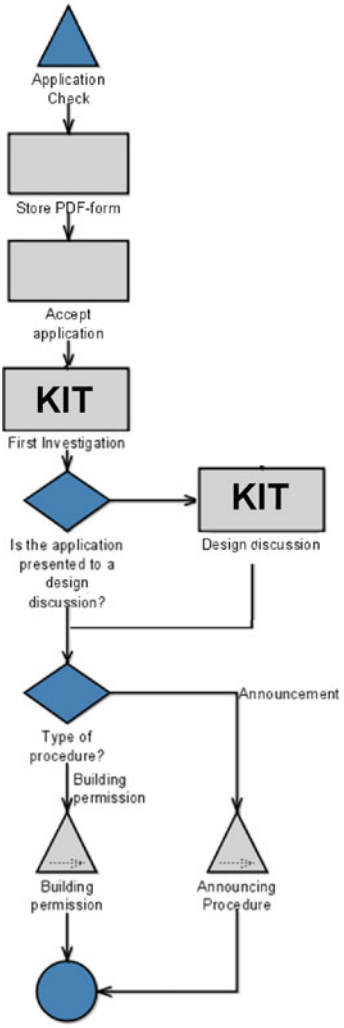


Fig. 19.13 Building permission process skeleton with knowledge-intensive parts

– First, to each activity we can associate sets of rules that are executed at run time and affect the execution by selecting resources, deriving values and checking constraints and guidelines. As shown in Section 19.5, these rules refer to concepts modeled in an ontology.
– Second, we have a new modelling object for variable process parts. A variable process corresponds to a subprocess with the particularity that the activities of the subprocess are determined at run-time instead of modeling time using action-enabling rules (see Section 19.5.1).

In Fig 19.14 the activities with a question mark are variable processes. The first variable process has a reference to various possible activities: formal investigation,

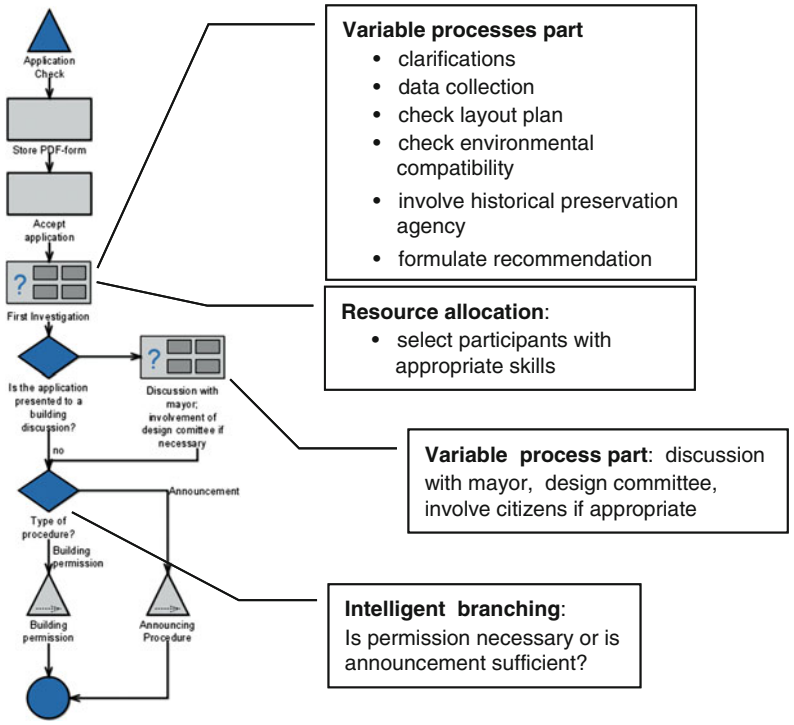


Fig. 19.14 Building permission as agile process

collection of data, check layout plan, check environmental compatibility, check for historical preservation, and formulate recommendation. The second one is executed if a design discussion with the mayor, experts and citizens is necessary.

Intelligent branching depends on inference and computation rules that are executed at run time. In the sample process there are two possibilities depending on the decision in previous phases: One branch starts a subprocess to handle a building permission application. The second branch leads to a subprocess for simpler cases where a building announcement is sufficient and no formal permission is necessary.

Using OWL, OWL-S and SWRL as interchange formats allows the migration of ontologies, process models and rules into executable code that can be Java, BPEL or commercial software to provide the appropriate interfaces. With that approach the agile process model can be executed. Fig. 19.15 shows the overall picture of agile process management. Rule sets are associated to business process models, terms and facts of the business rules are defined in an ontology. For execution a rule engine[25] is linked to a workflow engine,[26] both having access to application data.

[25] A business rule engine or inference engine is a software component that separates the business rules from the application code and allows deriving answers from a knowledge base.

[26] "The workflow enactment software interprets the process description and controls the instantiation of processes and sequencing of activities, adding work items to the user work lists and

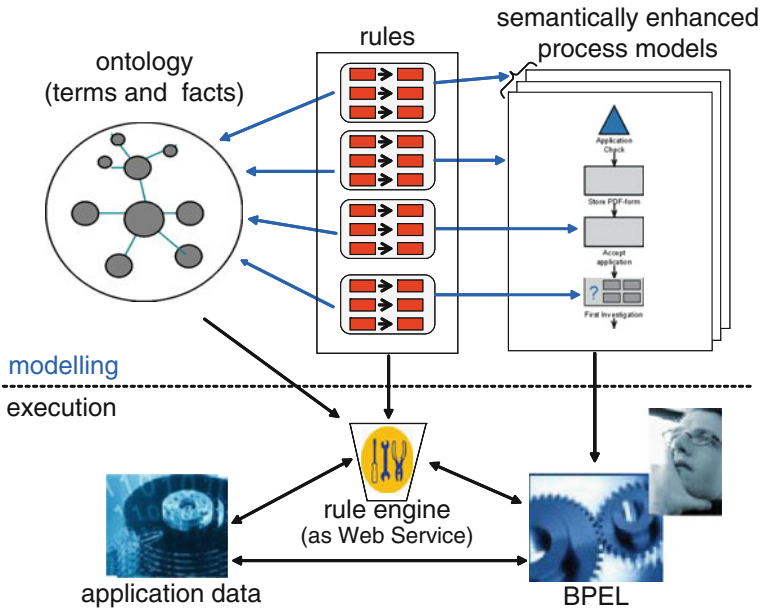


Fig. 19.15 Agile process management framework

19.7 Conclusion

Recent studies have shown that there is a significant advance in e-government in nearly every country but there are still many challenges. In this chapter we have presented some work to reinforce e-government that is based on the use of ontologies.

One of the greatest challenges is the effort required in building and maintaining ontologies. Although the public sector is strongly regulated and the services are nearly identical in many municipalities and public administrations, there is still no standard vocabulary. Even life events, a concept used to organise most of the portals and web sites differ in naming, granularity and structure. In many countries, however, there are projects and initiatives to define a standard vocabulary for describing resources and services. This will have an important influence in finding services on the web and in interoperability of services between public administrations. There are also international consultations to enforce compatibility, in particular in the European Union.

---

invoking application tools asnecessary. This is done through one or more co-operating workflow management engines, which manage(s) the execution of individual instances of the various processes." (TC00-1003 Issue 1.1 Workflow Reference, Workflow Management Coalition Page 14 of 14, URL: http://www.wfmc.org/standards/docs/tc003v11.pdf.

We presented an approach for ontology building that starts with the collection and consolidation of terms which in subsequent steps are related to facts and then can be transformed to ontologies. A major advantage of this approach is that it takes into account the fact that domain experts in the public administrations must be involved in this process.

Building ontologies, however, is only an initial step. We also described how ontologies can be used for semantic process modelling. This allows a public administration to exchange experiences in process design and lifecycle management. In combination with business rules this semantically enhanced process modelling results in what we called agile process management.

References

Abecker, A., A. Bernardi, K. Hinkelmann, O. Kühn, and M. Sintek. 1998. Toward a well-founded technology for organizational memories. *IEEE Intelligent Systems and their Applications* 13(3) S40–48, May/June 1998. Reprint in *The Knowledge Management Yearbook 1999–2000*, eds. J.W. Cortada, and J.A. Woods, 185–199. Boston: Butterworth-Heinemann.

Borst, P. 1997. Construction of engineering ontologies for knowledge sharing and reuse, Phd thesis, Enschede, The Netherlands: University of Twente. http://purl.org/utwente/fid/1392. Retrieved, 15 Dec 2006.

Business Rule Group. 2006. What is a business rule? http://www.businessrulesgroup.org/defnbrg.shtml. Retrieved, 05 Dec 2006.

Capgemini Consulting. 2006. Online availability of public services: How is europe progressing? Web Based Survey on Electronic Public Services, Report of the 6th Measurement, June 2006. http://www.capgemini.com/resources/thought_leadership/2006_online_availability_of_public_services/. Retrieved, 22 Dec 2006.

Chappel, O. 2006. Term–fact modeling, the key to successful rule-based systems. http://www.brcommunity.com/b250.php. Retrieved, 22 Nov 2006.

Conklin, J., and M. Begeman. 1998. gIBIS – A hypertext tool for exploratory policy discussion. In Proceedings of CSCW98, 140–152. Seattle, WA.

Daconta, M. C., L.J. Oberst, and K.T. Smith. 2003. *The semantic web*. New York, NY: Wiley.

Dublin Core Metadata Initiative. 2006. DCMI metadata terms. http://dublincore.org/documents/dcmi-terms/. Retrieved, 23 Dec 2006.

eCH. 2006. Best practice struktur prozessinventarliste. Standard Recommendation No. eCH-0015 of eCH eGovernment Standards. http://www.ech.ch/index.php?option=com_docman&task=doc_download&gid=326. Retrieved, 5 Jan 2007.

eCH. 2008. eCH-0049 Vernehmlassungseingaben Themenkatalog Privatpersonen. Standard Recommendation No. eCH-0049 plus addenda of eCH eGovernment Standards. http://ech.ch/index.php?option=com_docman&task=cat_view&gid=118&Itemid=181&lang=de. Retrieved, 01 Sep 2008.

Fernández-López, M., and A. Gómez-Pérez. 2003. Overview and analysis of methodologies for building ontologies. *Cambridge Journals* 17:129–156.

Fernandez, M., A. Gomez-Perez, and N. Juristo. 1997. METHONTOLOGY: From ontological arts towards ontological engineering. In Proceedings of the AAAI-97 Spring Symposium Series on Ontological Engineering, 33–40. Stanford, CA.

Fraser, J., N. Adams, A. Macintosh, A. McKay-Hubbard, T.P. Lobo, P.F. Pardo, R.C. Martínez, and J.S. Vallecillo. 2003. Knowledge management applied to egovernment services: The use of an ontology. Knowledge Management in Electronic Government: 4th IFIP International Working Conference, KMGov 2003, Rhodes, Greece, May 26–28, Procedings. Berlin/Heidelberg: Springer.

Gandits, F., and R. Schaffer. 2006. Standardattribute für Leistungen und Leistungsgruppen. Recommendation of the Austrian e-Government Bund-Länder-Gemeinden. http://reference.e-government.gv.at/uploads/media/st-att-1-0-0-2004-0102.pdf. Retrieved, 05 Jan 2007.

Gómez-Pérez, A. 2001. Evaluating ontologies. http://www3.interscience.wiley.com/cgi-bin/fulltext/77002631/PDFSTART. Retrieved, 15 Dec 2006.

Gruber, T.R. 1993a. A translation approach to portable ontology specifications. *Knowledge Acquisition* 5(2):199–220.

Gruber, T.R. 1993b, Toward principles for the design of ontologies used for knowledge sharing. In *Formal ontology in conceptual analysis and knowledge representation*, eds. N. Guarino, and R. Poli, The Netherlands: Kluwer Academic Publishers. For download at, http://www2.umassd.edu/SWAgents/agentdocs/stanford/ontodesign.pdf. Retrieved, 05 Jan 2007.

Grüninger, M., and M.S. Fox. 1995. Methodology for the design and evaluation of ontologies. http://www.eil.utoronto.ca/EIL/public/method.ps. Retrieved, 15 Dec 2006.

Gugliotta, A., V. Tanasescu, J. Domingue, R. Davies, L. Gutiérrez-Villarías, M. Rowlatt, M. Richardson, and S. Stinčić. 2006. Benefits and challenges of applying semantic web services in the e-Government domain. http://kmi.open.ac.uk/projects/dip/resources/Semantics2006/Semantics2006_DIP_Camera_Ready.pdf. Retrieved, 15 Dec 2006.

Gugliotta, A. 2006. Knowledge modelling for service-oriented applications in the e-Government domain. http://www.dimi.uniud.it/gugliott/thesisgugliotta.pdf. Retrieved, 15 Dec 2006.

Swiss Federal Strategy Unit for Information Technology FSUIT. 2004. HERMES – Management and execution of projects in information and communication technologies (ICT). http://www.hermes.admin.ch/ict_project_management/manualsutilities/manuals/hermes-foundations/at_download/file. Retrieved, 06 Jan 2007.

Hinkelmann, K., F. Probst, and B. Thönssen. 2006a. Reference modeling and lifecycle management for e-Government services. In eds. A. Abecker, A. Sheth, G. Mentzas, and L. Stojanovic, Semantic Web Meets e-Government, Papers from the 2006 AAAI Spring Symposium, AAAI Technical Report SS-06-06.

Hinkelmann, K., F. Probst, and B. Thönssen. 2006b. Agile process management framework and methodology. In eds. A. Abecker, A. Sheth, G. Mentzas, and L. Stojanovic, Semantic Web Meets e-Government, Papers from the 2006 AAAI Spring Symposium, AAAI Technical Report SS-06-06.

Horrocks, I., P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. 2004. SWRL: A semantic web rule language, combining OWL and RuleML. http://www.w3.org/Submission/SWRL/. Retrieved, 08 Dec 2006.

Chevallerau, F.-X. 2005. eGovernment in the member states of the European Union, Brussels. http://europa.eu.int/idabc/egovo. Retrieved, 05 Jan 2007.

Kalfoglou, Y., and M. Schorlemmer. 2005. Ontology mapping: The state of the art. http://drops.dagstuhl.de/opus/volltexte/2005/40/pdf/04391.KalfoglouYannis.Paper.40.pdf. Retrieved, 25 Dec 2006.

Kunz, W., and H.W.J. Rittel. 1970. Issues as elements of information systems. WP-131, University of California. http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf. Retrieved, 5 Dec 2005.

Macias, J.A., and P. Castells. 2004. Providing end-user facilities to simplify ontology-driven web application authoring. *Interacting with Computers* 19(4):563–585, July 2007.

Maedche, A., and S. Staab. 2004. Ontology learning. In *Handbook on ontologies*, eds. S. Staab, and R. Studer, 173–189. New York, NY: Springer.

McGuinness, D.L., and F. van Harmelen, eds. 2004. OWL web ontology language overview: W3C Recommendation 10 Feb 2004. http://www.w3.org/TR/owl-features/. Retrieved, 15 Dec 2006.

Noy, N.F., and D.L. McGuinness. 2002. Ontology development 101: A guide to creating your first ontology. http://protege.stanford.edu/publications/ontology_development/ontology101.pdf. Retrieved, 15 Dec 2006.

Martin, D., ed. 2004. OWL-S: Semantic markup for web services. W3C Member Submission 22 Nov 2004. http://www.w3.org/Submission/OWL-S/. Retrieved, 06 Jan 2007.

Peristeras, V., and K. Tarabanis. 2006. Reengineering public administration through semantic technologies and the GEA domain ontology. AAAI Spring Symposium on Semantic Web Meets e-Government, Stanford University, March 2006. http://www.semantic-gov.org/index.php?name=UpDownload&req=getit&lid=1. Retrieved, 15 Dec 2006.

Pinto, H.S., and J.P. Martins. 2002. Evolving ontologies in distributed and dynamic settings. In *Proceedings of the 8th International Conference on Principles of Knowledge Representation and Reasoning (KR2002)*, eds. D. Fensel, F. Giunchiglia, D.L. McGuiness, M.A. Williams, San Francisco, CA: Morgan Kaufmann.

Pinto, H.S., and J.P. Martins. 2004. Ontologies: How can they be built? *Knowledge and Information Systems, Computer Science*, 6(4). http://springerlink.metapress.com/content/0p5yqrdh5t5dvd06/fulltext.pdf. Retrieved, 16 Dec 2006.

Schreiber, G., H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van de Velde, and B. Wielinga. 1999. *Knowledge engineering and management: The CommonKADS methodology*. Cambridge, MA: MITPress.

Stojanovic, L., A. Abecker, D. Apostolou, G. Mentzas, and R. Studer. 2006. The role of semantics in eGov service model verification and evolution. AAAI Spring Symposium on Semantic Web Meets e-Government, Stanford University, March 2006.

Sure, Y., M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. 2002. OntoEdit: Collaborative ontolgy development for the semantic web. In *International Semantic Web Conference ISWC 2002*, LNCS 2342, eds. I. Horrocks, and J. Hendler, 221–235. Heidelberg: Springer. http://www.aifb.unikarlsruhe.de/WBS/ysu/publications/2002_iswc_ontoedit.pdf. Retrieved, 15 Dec 2006.

Uschold, M., M. King, S. Moralee, and Y. Zorgios. 1998. The enterprise ontology. *Cambridge Journals* 13:31–98.

Uschold, M. 1996. Building ontologies: towards a unified methodology. Proceedings of Expert Systems 96, Cambridge, Dec 16–18 1996.

Von Halle, B. 2002. *Business rules applied, building better systems using the business rules approach*, New York, NY: Wiley.

Wimmer, M., and E. Tambouris. 2002. Online one-stop government: A working framework and requirements. In Proceedings of the IFIP World Computer Congress, Montreal, Aug 26–30 2002.

A. Wroe. 2006. TERREGOV white paper: Semantic based support to civil servants. http://www.terregov.eupm.net/my_spip/index.php?param=12. Retrieved, 15 Dec 2006.