Semantics-driven Development for Electronic Government Applications

Charles Crichton, Jim Davies, Jeremy Gibbons, Steve Harris, Aadya Shukla, and Andrew Tsui
Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD UK
{*crc,jdavies,jg,sharris,aadya,pwtsui*}*@comlab.ox.ac.uk*

Abstract

It has been argued that the challenges in electronic government are purely social: that existing methods and tools are perfectly adequate, and that electronic government projects fail because of people, not technology. While acknowledging that there are organisational and political challenges, this paper argues that there is also a significant, technical challenge, and a corresponding technology gap. It argues also that this challenge may be addressed through a combination of model-driven development and semantic technologies, and reports briefly upon a successful, prototypical application.

1. Introduction

A three-year investigation sponsored by the European Commission, completed in 2007, identified seven major barriers to electronic government: leadership failures, financial inhibitors, digital divides, poor coordination, organisational inflexibility, lack of trust, and poor technical design. The last point refers not to software architecture, but to user interface design, where eGovernment "often lags behind business and societal innovation on the Internet" [21].

While there are major barriers in each of these areas, there are also fundamental, technological challenges. The requirements upon software in electronic government extend beyond those presently addressed in enterprise computing; existing methods, tools, and platforms are not enough; and software development projects in the public sector appear particularly prone to failure [14]. In the UK,

- the Child Support Agency was established to ensure that separated parents continue to meet their financial responsibilities. A new information system was introduced, in support of significant organisational reform, in 2003. By 2006, a backlog of 267,000 cases had built up, with an estimated 36,000 "stuck due to IT failures". Despite charging £456m for its services, the supplier had "struggled to deliver a system that was fit for purpose"—the 500 remaining faults, and 14 critical defects, prevented the envisaged deployment of the software. [27]
- the National Programme for IT in the National Health Service (NHS) was intended to improve services and the quality of patient care. Launched in 2002, the programme is now at least 5 years behind schedule, and the claimed successes correspond to developments that would have been completed in any case. The promised, national integration of data and services has yet to appear, despite an estimated cost of £12.7 billion. [28]

In these cases, there are often problems in, for example, procurement practices, contract management, change management, communications, and decision-making. However, it is difficult to solve these problems in the absence of adequate software technology: when existing solutions fail to match requirements; when new software is too expensive to produce; and when the process of software development remains completely opaque to sponsors and stakeholders.

In this paper, we explain why the software requirements for electronic government exceed those of enterprise computing. We identify a number of specific domain challenges, and argue that these are difficult, if not impossible, to meet using existing technology. We then show how this technology gap may be addressed through a combination of model-driven development and semantic technologies, and report briefly upon a successful, prototypical application. The paper ends with a discussion of related work and future research.

2. Electronic government

The term *electronic government* means more than a literal translation of existing government services and processes into electronic form: it carries expectations of transformation, often in connection with hopes for a better society. According to the World Bank, the European Commission, and the Pacific Council, electronic government is the use of information technology...

- "...to improve outreach, access of information, reduced corruption, increased transparency, greater convenience, revenue growth and cost reductions" [2];
- "...combined with organisational change and new skills in order to improve public services and democratic processes..." [20];
- "...to promote more efficient and effective government, facilitate more accessible government services, encourage interaction between users of public services and providers..." [22].

The explicit mention of transformation, and the use of words such as *access*, *transparency*, *democracy*, and *interaction*, suggests that there may be particular challenges in software development for electronic government. Indeed, Scholl [24, Proposition 5] argues that "Electronic Government poses a new challenge regarding government records (in terms of creation, maintenance, preservation, security, integrity, and accessibility)" over conventional business computing.

2.1. Stakeholder engagement

In electronic government, the stakeholders, including the end users, have a particular relationship to the processes of development and operation: this system is being procured, designed, developed, and operated on their behalf, and at their expense. "Priority of citizens trumps cost considerations" in electronic government, whereas in electronic commerce, "reducing cost was the main driver" [3]. We might consider there to be an implicit contract, reflected in the system requirements, similar to that which exists between government representatives and the people they represent.

This means that the extent to which requirements are "owned by the users" is far greater, and thus the system must be a better fit for the social processes that it is intended to support than is often the case in ordinary enterprise computing. Greater stakeholder involvement is one of the key differentiators of electronic government over electronic commerce systems identified in the recent comparison study reported in [3]. Furthermore, stakeholders may require more in the way of evidence that the system is in fact doing what is expected—the implicit contract applies in operation as well as in development.

There is also a general lack of understanding and expertise regarding information technology on the part of stakeholders. When these stakeholders are driving policy and decision making, then requirements elicitation and iterative design may be extremely difficult: although developers may be able to hypothesise and identify key issues and functionality, there may be insufficient intellectual bandwidth to communicate these to stakeholders in a timely fashion.

This is a more pronounced problem in government than in ordinary business [8], as business stakeholders usually (but not always) have a firmer grasp of the technical aspects of information modeling in their business, whether their role is as customer or manager. This is true also in interdisciplinary research, where a successful development requires engagement with scientists who may have little or no intuition regarding information modeling and systems development.

A related concern is the under-resourcing of engagement activities, most notably in the specification and oversight of partnerships with industry [17, 1, 4, 13]. This is complicated by the general lack of technical engagement on the part of policy makers and public sector management [11]—and the difficulty of addressing this lack in the context of outsourcing pressure and 'lean government' culture.

The impact of poor technical engagement would be reduced if procurement and implementation questions could be resolved at the strategic or business level. Unfortunately, the state of the art in software technology is such that this is not yet the case: tried and tested, off-the-shelf solutions do not yet exist in this domain. There is no clear indication of the relative performance of suppliers on similar projects, it is difficult to evaluate candidate solutions at an appropriate level of detail, and there is a general lack of competition [25].

2.2. The context of development

The requirements of electronic government systems are more complex than those of their commercial counterparts; they are also more subject to change. Policy reforms or shifts in public opinion may require substantial changes to the design of a system, changes that may be expensive to make once development is underway. In a commercial context, it is quite common to find that information system design is shaping business processes; in electronic government, this is less likely to be acceptable.

It is also more important that these requirements are correctly reflected in the behaviour of the system. In electronic government, computing systems will do more than facilitate policy—they will serve as its principal, and perhaps its only, implementation. This has significant implications not only for the criticality of development processes, but also for the design of the systems themselves.

In a commercial system, the information pertaining to an individual may define and constrain that individual as a customer; in a government system, it may define and constrain that individual *as a citizen*. The data may be driving the processes of government as they act upon the individual: there is a greater responsibility [18] to maintain its correctness and availability over time.

There is a strong requirement for transparency and *accountability*. The users should be able to assure themselves that the data remains correct, and to see how it is being used and maintained. This is essential if they are to have the desired level of *trust* in governments and the systems that represent them. The scope for regarding data and processes as "internal" is far less—any key mechanism, anything that decides the published values, must be open to scrutiny.

There is also a greater requirement for compliance with external standards and interoperation with other systems: in the distribution of decision-making to regional governments, the increasing role of the private sector in service delivery, and the emergence of challenges requiring coordinated action across national boundaries. Stronger emphasis on legal considerations is another differentiator of electronic government systems identified by Barzilai-Nahon and Scholl [3].

Finally, problems of scalability are exacerbated by the need to deal with the whole of a community or society [7]. Although the number of individuals involved may be less than, for example, the number of users of a global social networking or auction site, the expectation of universal coverage makes it necessary to support a far greater number of different perspectives. Even when it is acceptable to provide services on a selective basis, excluding certain groups within the population, the decision to do so may have political as well as economic consequences; stronger emphasis on ethical considerations and on inclusivity are further differentiators of electronic governance [3].

3. Technological challenges

The domain challenges described in the previous section give rise to specific technological challenges, where expectations extend beyond the capacity of technology already developed for other domains.

3.1. Data sharing

The need for 'one-stop government' [30] has put extra emphasis on information sharing across departments and agencies: for example, Janssen and Cresswell [15] report that the Dutch government has been aiming to allow citizens to provide any piece of data only once, and to require government agencies to share and reuse it. One-stop government represents the highest level of interoperability—and of technological complexity—in Layne and Lee's representation of the stages of development of electronic government [19]: from simple *cataloguing* (downloadable information) and *transactions* (uploadable forms) to *vertical systems* (linking common functions at different levels of government—local, state, and federal) and full *horizontal integration* (linking across different functions, within the same level of government).

The extent of automation in information sharing is a key issue—and if data cannot be shared automatically, then it usually will not be shared in a timely, secure, or appropriate fashion.

- The UK Crown Prosecution Service (CPS) was sent DNA profiles by Dutch police in 2007, but was unable to check these against the national DNA database in a timely fashion. When checks were eventually run, it was found that 11 of the suspects had since committed crimes—crimes that might have been prevented had the data been integrated earlier.

- The UK Revenue (HMRC) misplaced two discs containing records on 25 million individuals, discs that need never have been produced in the first place. A small subset of the data in question was required (by another government agency), but could not be automatically produced and transmitted. Instead, a crude, manual procedure was adopted, which resulted in the loss of a significant amount of sensitive personal data.

The most obvious, technological barrier to effective data sharing is a lack of functionality, at an appropriate level of automation, accessible to the end user. Both of the agencies above had both staff and data available, and the sharing requirement was clearly understood, but the requisite data integration and transmission could not be performed—the functionality had not been implemented, or had not been made accessible to the users.

We might hope that this barrier could be breached by additional training, by more careful management of requirements, or through better management of the relationship with the supplier. However, in many cases, it is simply not economic for the supplier to modify the design to address a particular requirement. It may be that the data itself simply does not admit to effective integration or sharing, as the precise meanings are incompatible and/or inaccessible.

3.2. Semantics

When data refers to the result of a measurement, observation, or action in the real world, then a considerable amount of contextual information may be required to make sense of it for a particular purpose. A spreadsheet of numbers with columns titled *A*, *B*, and *C* would have little value without some indication of what *A*, *B*, and *C* represent. A short, informal explanation of meaning, however, is rarely enough: we should draw our comparisons with the verbosity required in legal documents, and the effort that goes into establishing and maintaining agreed interpretations.

The coverage and scale of electronic government initiatives means that we are required to share information across complex networks of organisations, people, languages, information systems, information structures, rules, processes, and practices [23]. This makes the establishment and maintenance of common meanings a significant challenge: these meanings have to be open and accessible to a wide range of people.

Wimmer *et al.* [31] identify 'semantic and cultural interoperability' (that is, cross-organisational collaboration specifically between stakeholders from different cultural backgrounds) as an important theme of future electronic government research, listing "how can information systems be modeled and designed, which embody semantic and cultural interoperability?" and "how can a shared understanding and seamless interoperability of public service design be created among different cultures and communities?" among the key research questions in the domain.

For example, Dawes *et al.* [9] discuss the example of land parcel data, used for taxation, conveyancing, emergency response, transportation routing, facility siting, town planning, infrastructure management, and notification. Such versatile data is clearly extremely valuable, but that value can only be extracted if the data semantics is properly recorded; Dawes *et al.* cite issues with data inconsistencies reducing the value of the data as soon as it is collected.

A natural tool in the communication of common meanings, or at least representations, is XML (eXtensible Mark-up Language): this standard notation allows the communication of additional, contextual information—metadata—along with data content, and provides the syntactic framework for the transport of meanings; and XML schema can be used to suggest meaning. However, XML does not by itself provide any standard means to capture and communicate a semantic definition.

The same is true of the other technologies of the *semantic web*: they allow us to associate metadata with data, and this metadata may correspond to semantic information, but they do not, as yet, provide any standard means of associating metadata with informal meanings and usages. For example, OWL Full includes `owl:DeprecatedClass` and `owl:DeprecatedProperty`, but the user is left to decide exactly what the interpretation of these tags might be, and how they are to be used in a particular context.

3.3. Metadata elements

What is required is an agreed standard for structured, semantic metadata, a specification of how we may represent and communicate meaning through the use of XML and other technologies. In practice, this entails more than the use of a basic data dictionary, or the definition of a single schema or metamodel. Consider for example, the British Standard address specification [26], included as part of the electronic Government Interoperability Framework [6]. This specification can be described in terms of 60 classes or entities, and this description can be exported as an XML schema: a fragment of this schema is shown in Figure 1.

```
<xsd:complexType name="BSaddressStructure">
  <xsd:sequence>
    <xsd:element
      name="SAON" type="AONstructure" ...
    <xsd:element
      name="PAON" type="AONstructure" ...
    <xsd:sequence>
      <xsd:element
        name="StreetDescription" ...
      <xsd:element
        name="UniqueStreetReferenceNumber"
        type="USRNtype" minOccurs="0" ...
    </xsd:sequence>
    <xsd:choice>
    ...
      <xsd:sequence>
        <xsd:element ref="Town"/>
        <xsd:element
          ref="AdministrativeArea" ...
      </xsd:sequence>
      <xsd:element ref="AdministrativeArea"/>
    </xsd:choice>
    <xsd:element name="PostTown" ...
    <xsd:element name="PostCode" ...
  ...
```

Figure 1. Part of an address schema

Although this describes a representation, and is extremely useful in that regard, it may not be an adequate specification of meaning: postal addresses may be collected for different purposes within a particular application, or even within a single document.

For example, we might combine postal addresses of industrial corporations with records of the amount of industrial waste produced, in order to present a map of areas of particularly high pollution within the United States. But the results would be meaningless if the postal addresses were those of the corporations' headquarters, and not those of the actual polluting facilities. Similarly, a map of locations where crimes are reported might prove to be a map of police stations, as opposed to a map of places in which crimes actually occur.

Furthermore, the development of XML schema without a coherent semantic or modelling framework can lead to confusion and inconsistency. For example, the UK Government Data Standards Catalogue includes the schema `PersonDescriptiveTypes`, part of which is shown in Figure 2. Each of the descriptive types shown shares a verification level element with the same representation: `0`, `1`, `2`, or `3`. However, the meaning of the verification level—what has to be done to achieve it—is different in each case.

Inspection of accompanying documentation suggests that the following interpretations should apply to the verification level values:

- birth date: `1`, secondary cerificate; `2`, primary certificate; `3`, undefined.
- death date: `1`, declaration by resposible person (hospital/police); `2`, declaration by court; `3`, primary certification.
- marital status:
  - married: `1` secondary certificate; `2` confirmation from unit; `3` primary certificate;
  - divorced: `1` confirmation from unit; `2` primary certificate; `3` undefined;
  - widowed: `1` declaration by responsible person; `2` declaration by court; `3` primary certificate;
  - separated: `1` telephone call; `2` decree nisi; `3` judicial separation.

The schema simply doesn't contain enough information for users or developers to consistently assign verification values. Not only is more informal explanation required, we should be able to determine automatically that the verification level value means something different in each case.

```
<xsd:simpleType name="VerificationLevelType">
  <xsd:restriction>
    <xsd:enumeration value="Level 0"/>
    <xsd:enumeration value="Level 1"/>
    <xsd:enumeration value="Level 2"/>
    <xsd:enumeration value="Level 3"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="PersonBirthDateStructure">
  <xsd:sequence>
    <xsd:element name="PersonBirthDate"
 type="core:DateType"/>
    <xsd:element name="VerificationLevel"
 type="VerificationLevelType"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="PersonDeathDateStructure">
  <xsd:sequence>
    <xsd:element name="PersonDeathDate"
 type="core:DateType"/>
    <xsd:element name="VerificationLevel"
 type="VerificationLevelType"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType
  name="PersonMaritalStatusStructure">
  <xsd:sequence>
    <xsd:element name="MaritalStatus"
                 type="core:MaritalStatusType"/>
    <xsd:element name="VerificationLevel"
                 type="VerificationLevelType"/>
  </xsd:sequence>
</xsd:complexType>
```

Figure 2. a part of `PersonDescriptiveTypes`

We need more information, and we need it available automatically, without the need for costly manual intervention, to:

- choose an appropriate meaning and value domain for data that we are to collect (or have already collected);
- look up the meaning of data already collected, and determine whether it is compatible with data from another source;
- if possible, discover how data with this meaning can be transformed for use in other situations.

Without this access, we will find that systems cannot be integrated, and data cannot be combined with any guarantee of consistency.

Moreover, it is unrealistic to insist on a single, unified collection of metadata items, on which all participants must agree. This would require forethought for interoperability: only systems designed to employ this distinguished metadata may work together. Rather, scope must be provided for federated, versioned metadata repositories, with translations between different views—not so much 'Esperanto' as a 'BabelFish'—allowing both unpreplanned integration of external systems, accommodation of different points of view, and revision of metadata in the light of experience and changing requirements.

4. Model-driven development

One way to represent the semantic information required, and to facilitate programmatic access, is to represent the various aspects of semantics using models of usage. We can identify three particularly useful kinds of model:

- *ontologies*, models which explain the meaning of a metadata item in terms of named relationships to other elements;

- *applications*, models in which the item appears in context: for example, in the context of a design document, or a form template;

- *transformations*, models which explain how data collected against one set of elements can be transformed to fit another

Although only the first of these is usually seen as defining or recording meaning, the others also have semantic import: meanings are sometimes best expressed, and will evolve, through usage.

The use of a registry of models, linked by references to common metadata elements, amounts to an extended form of Model Driven Development (MDD), with an obvious advantage: if we can develop systems that are faithful to models in which every attribute is a registered metadata element, then we can be sure of the meaning of the data these systems collect.

If we can develop and configure systems through automatic transformations, then we have a means of automatically translating agreements on intended meaning—developed iteratively, in collaboration with stakeholders—into the behaviour of the working system. This would work to increase trust, to promote standards adoption, and—by reducing the cost of developing alternative or additional functionality—to facilitate the development of systems that meet the varying needs of a wide range of stakeholders.

The extent to which automation is possible depends upon the kinds of properties that we wish to express in our models, and the kinds of system that we need to build. As such, it can be determined only through engagement with stakeholders, through practical experience and successful application. We have successfully demonstrated the integ

Our characterisation of the technology gap in electronic government, and our proposed solution, is based upon experience in a related domain: that of healthcare informatics.

Many of the challenges of electronic government are encountered also in healthcare informatics, and effective data sharing is equally important: without the ability to combine data from different sources and different experiments, it may be impossible to draw any conclusions as to the effectiveness of a particular treatment, or to establish the relationship between genetic cause and clinical effect.

The same basic approach to the registration of metadata elements has been adopted in both domains. The largest metadata registry in use, with over 17,000 registered elements, is maintained by the US National Cancer Institute. The same implementation, however, has been adopted by the US Departments of Justice and Homeland Security for their *National Information Exchange Model*, intended to "develop, disseminate and support enterprise-wide information exchange standards and processes… to effectively share critical information in emergency situations, as well as support the day-to-day operations of agencies throughout the nation" [10].

Both of the above registries conform to ISO 11179, an international standard for metadata registration. At present, this standard lacks any support for models, and supports only the definition of semantics in terms of a single conceptualisation of the domain. These shortcomings are easily remedied, and in our own implementation of the standard, we have constructed a metadata registry [12] that can hold models of application and transformation, as well as multiple ontologies or conceptualisations.

Our extension and interpretation of the standard was driven by our need to apply a model-driven approach to the development of software systems handling clinical and genomic data. From the perspective of systems development, these systems have much in common those required in electronic government: for example, the ethical issues surrounding the acquisition and processing of the data, the degree of accountability, the scale and complexity of the enterprise, involving (in the UK at least) a wide variety of organisations, and—last but not least—the potential importance of the data for the wellbeing of the subject.

4.1. Models

We can use any kind of model to express or extend the semantics of a metadata element, provided that a suitable metamodel exists within our chosen modelling framework. To date, we have used:

- SKOS [29] ontologies to describe basic classifications of metadata, for administrative or organisational purposes, or to further explain their semantics through additional context
- UML models, XML schema, InfoPath templates, and XForms to describe applications
- XSLT documents to describe transformations

Our transformations to date have been very limited in nature—we have concentrated upon facilitating the registration, identification, and re-use of metadata elements in application models.

The transformations used for the generation of software artifacts from the application models, and for the semantics-driven integration of data from different sources, are implemented in Java and C#. Although these could be registered as metadata elements, their availability for re-use would be only of theoretical interest at this point.

4.2. Curation

Metadata elements are added to the registry on the basis of need or design. If there is an existing data set or application that we wish to connect to, then we must add an appropriate collection of semantic metadata elements. Alternatively, we may add elements to describe preferred forms of measurements or observations, even if those forms are not yet in use.

Naturally, the latter are often based upon the former, but it requires engagement with domain experts and stakeholders, and a process of manual curation, before a set of metadata elements can be recommended for use in a particular domain. (That recommendation itself is captured as a model, relating the elements to their preferred definitions, and recording the basis for the recommendation).

The attributes of metadata elements are left unchanged from the point of registration: however, new references can be made to them at any time, adding new classifications or models of usage. In this way, we can identify the precise semantics associated with any piece of data, simply by listing a collection of unique identifiers that correspond to basic elements (combinations of terms and value domains) and models (a complete characterisation of the context used at the time).

Although based entirely upon extension, rather than revision, the curation task requires significant effort if the desired levels of interoperability and re-use are to be achieved. Fortunately, this effort can be distributed: different agencies or users can produce new classifications, put forward new ontologies asserting compatibility or refinement, and register new transformations.

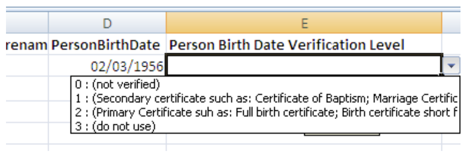
4.3. Access

The metadata elements in the registry are accessed through *plug-ins* for office applications and modelling tools: in particular, Word, Excel, InfoPath, and Enterprise Architect. These are commonly-used applications for the documentation of design intentions and the presentation of information.

Our plug-in implementations share a user interface control that allows for free-text and classification-based search across multiple registries. The difference between them lies in what happens when a metadata element is selected for use.

In Word, the plug-in simply inserts the preferred name for the metadata element, together with a link to its semantics. It can also be used to create new metadata elements and transmit them back to the registry—a user who is unable to find an element with the right semantics can create a new one, register it, and use it. It may later be replaced, via a registered transformation, with a recommended element, or it may itself form part of a later recommendation.

In Excel, the element is used to apply a type to the currently selected column; if this is an enumerated type, then the enumeration definition itself is incorporated as an additional worksheet, as a record of the semantics used. The typing information is used as a control: only input of an appropriate type will be accepted and, in the case of an enumerated type, a drop-down box offers the list of acceptable values. The result is an Excel document with a built-in semantics that is both machine- and human-readable. Again, the plug-in can also be used to create new elements.



	D	E
renam	PersonBirthDate	Person Birth Date Verification Level
	02/03/1956	

0 : (not verified)
1 : (Secondary certificate such as: Certificate of Baptism; Marriage Certific
2 : (Primary Certificate suh as: Full birth certificate; Birth certificate short f
3 : (do not use)

Figure 3. Using a metadata element in Excel

Figure 3 shows the result of incorporating a simple metadata element in a spreadsheet. A “drop down” appears on data entry, constraining the user’s choice to the appropriate value domain, and providing notes on meaning. There is also a clickable link to another sheet in the spreadsheet, automatically generated and maintained, that contains the full definition of each metadata element used—a snapshot of the information in the metadata registry—complete with links back to the original source.

If the metadata element is a basic element (the simple combination of a term and a value domain, with references to one or more models), then the Enterprise Architect (EA) plug-in will give semantics to an attribute, or add a new class to the current UML model. If it is a model, then the user has the option of importing the model components into the current design, or simply including an appropriate reference.

The InfoPath plug-in has a dual role. If it is being used—by a domain expert—to design a new form, then it allows the selection of semantic metadata to describe new fields. If it is being used to instantiate an existing form definition, it allows appropriate data to be inserted—either as a fresh value, in which case it is automatically associated with the appropriate semantic metadata, or by reference, in which case any existing semantic metadata is checked for consistency.

4.4. Meta-data mining

Experience suggests that a surprising amount of re-usable metadata can be created automatically from existing artifacts. In the course of our work in the healthcare informatics domain [5], we created semantic metadata on the basis of existing forms implemented in Microsoft's *InfoPath*: forms used for national clinical studies by US Veterans Health Administration. We constructed a prototype tool to mine an existing set of forms for candidate metadata elements; these candidate elements were then examined, curated, and used as the basis for the generation of a replacement set of forms.

An InfoPath form template comprises an archive of files in a Microsoft Cabinet file: XML schemas to describe the type and structure of the information; XSLTs to describe its presentation—graphics, text, and fields; and a manifest, giving additional information, such as validation rules. The fields are described using a specific markup, using XPath to bind the field to the XML datastructure that the form fills in: the InfoPath application uses the XSLTs to produce HTML with extra tags for binding the fields.

Our mining tool reads the form manifest, and then extracts a collection of candidate metadata elements from the XML schemas *and* the XSLTs. The XSLTs present a loose association between a piece of text describing a field to the user and a field binding, and this allows us to suggest a candidate semantics based upon the text of a question, rather than just an XML tag name. The usage of tables may mean that the precise association between text and field is not always determined: if it cannot match the layout against any of a series of

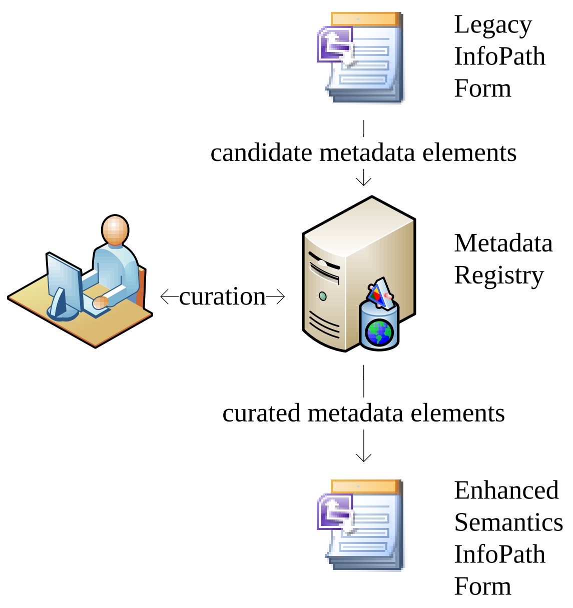

Legacy InfoPath Form
candidate metadata elements
Metadata Registry
←curation→
curated metadata elements
Enhanced Semantics InfoPath Form

Figure 4. metadata mining

common patterns, the tool produces all of the possible candidates, as input into a manual curation process.

An obvious advantage in programmatic mining of existing software artifacts is that we have the opportunity of evaluating semantics on the basis of actual usage, rather than reported intention. Data that is needed for purely administrative or internal purposes, and does not form part of a reporting requirement, may well be overlooked by any semantic review process: determining what is actually used at present, and how, greatly improves the chance of successful integration.

4.5. Re-use

Although existing programs and services would need to be rewritten to take full advantage of structured semantic metadata, we can often re-use existing software by creating a mapping—a translation table—between its data attributes and metadata items with registered semantics. The mapping for a specific component can then be used, on-the-fly if necessary, to (re-)unite the data produced with its semantics.

This approach is useful also in the development of new software artifacts. It is generally impossible to incorporate every aspect of a system design within a metamodel, and thus to produce models that completely determine the behaviour of a component. In model-driven development, any software artifacts that are automatically generated may require manual customisation before final deployment—this is particularly true of user interface components.

We would not wish to repeat the process of customisation whenever the model was updated, and yet we need to know that the semantics of the data collected is that given in the latest version of the model. We can achieve this by creating mappings between the data attributes in the generated software and local, fixed attributes. These may themselves be registered, in the context of a model of the specific application.

4.6. Generation

Much of our generation activity has been focussed upon forms: we create models of forms in which every field is associated with registered semantic metadata, and we generate working forms from these models. The form model is stored as an XML document, easily transformed to an XML schema in which each of the sections is a `complexType`.

This schema is then automatically marked-up using the Semantic Annotation for Web Services Description Language (SAWSDL) [16], with references that associate each part of the XML schema with the appropriate metadata, making the underlying semantics accessible to any SAWSDL-aware tool. The intent of the form is thus captured in a platform-independent fashion: the same model can be used to generate forms in XForms, InfoPath, and in the Windows Presentation Foundation.

Once an artifact has been created—a new model, or a new form—then this can be registered as a model, adding to the usage semantics of the metadata elements involved. Naturally, the addition of a model does not imply any recommendation for re-use: it simply provides a record of usage, together with all of the provenance data, and precise semantics, that another user or agency would need to decide upon its suitability.

In our work on clinical informatics, we have produced a basic, but complete, semantics-driven framework for data capture applications, shown in Figure 5. Developers can use the Enterprise Architect (EA) tool to create class diagrams in which attributes are linked to data elements in a metadata registry. A UML profile is used to generate XML stylesheet transformations from the EA project (eap) file, for components such as dates, text boxes, and lookups.

Another transformation generates the XML schema with SAWSDL annotations, which is used as the basis for the automatic generation of a raw XForm *and* as the basis for automatic processing and integration—using RDF and SPARQL—of any subsequent, submitted data (not shown in the figure for reasons of space). This XForm is then customised by the generated XSLT to produce a form ready for deployment.

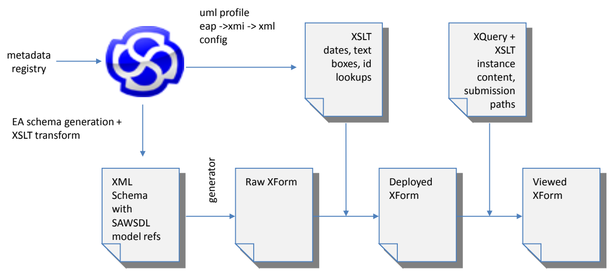

metadata registry
uml profile eap ->xmi -> xml config
XSLT dates, text boxes, id lookups
XQuery + XSLT instance content, submission paths
EA schema generation + XSLT transform
generator
XML Schema with SAWSDL model refs
Raw XForm
Deployed XForm
Viewed XForm

Figure 5. semantics-driven development

5. Discussion

In this paper, we have argued that a significant technology gap exists in the domain of electronic government. We have presented a brief, top-down account of problems and necessity, and a bottom-up account of practical (in)sufficiency. We discussed prototypical technology that appears to be at least a partial solution to these problems, but which is not presently in use in the electronic government domain.

Having discussed both challenges and an existing, partial solution, we are in a position to set out what may prove a significant research challenge in software engineering: to make semantics-driven management of data standard, accepted practice across the whole spectrum of enterprise computing, and for electronic government in particular.

This requires an advance in the state of the art. We need methods and tools for the creation, maintenance, and deployment of models of information, studies, and processes. The development of these methods and tools will require the incorporation of techniques for effective collaboration with users and domain experts, as well as advances in software technology.

A key point is that a continuously-evolving context in which meaning and interoperability are critical concerns forces a raising in the level of abstraction. The current state of the art in model-driven development entails automatic generation of system artifacts, but manual construction of the transformations that generate them. In the context of electronic government—and elsewhere—these transformations, too, will need to be automatically generated. As our semantics evolve, as our registries are updated, as our requirements change, the transformations that link our semantics to data need to be re-generated to match.

While acknowledging that this research challenge is only one of those faced in electronic government—problems with software tend to occupy only a small proportion of the reports of enquiry into initiatives such as the CSA computerisation [27]—we would submit that these other challenges will be easier to resolve if the software systems were guaranteed fit for purpose, and driven by the semantics of the data they hold.

References

[1] D. Aldrich, J.C. Berot, and C. McLure. E-government : Initiatives, developments and issues. *Government Information Quaterly*, 19, 2002.

[2] World Bank. egovernment practice group. `http://go.worldbank.org/C81XI6R6F0`.

[3] Karine Barzilai-Nahon and Hans J. (Jochen) Scholl. Similarities and differences of e-commerce and e-government: Insights from a pilot study. In *Proceedings of the 40th Hawaii International Conference on System Sciences*. IEEE, 2007.

[4] T. Bovaird. Public-private partnership : From contested concepts to practice. *Journal of Public Administration*, 70, 2004.

[5] Radu Calinescu, Steve Harris, Jeremy Gibbons, Jim Davies, Igor Toujilov, and Sylvia B. Nagl. Model-driven architecture for cancer research. In *The 5th IEEE International Conference on Software Engineering and Formal Methods (SEFM 2007)*, pages 59–68, 2007.

[6] S.D. Clarke. e-GIF, e-GMS and IPSV: What's In It For Us? *Legal Information Management*, 7(04):275–277, 2007.

[7] Commission of the European Communities. The i2010 eGovernment action plan. Technical report, European Union, 2007. `http://ec.europa.eu/idabc/servlets/Doc?id=25286`.

[8] Wm. Arthur Conklin. Barriers to adoption of e-Government. In *Proceedings of the 40th Hawaii International Conference on System Sciences*, 2007.

[9] Sharon S. Dawes, Meghan E. Cook, and Natalie Helbig. Challenges of treating information as a public resource: The case of parcel data. In *Proceedings of the 39th Hawaii International Conference on System Sciences*, 2006.

[10] Department of Justice and Department of Homeland Security. National Information Exchange Model. `http://www.niem.gov`.

[11] S. Glaister. Past abuses and future uses of private finance and public private partnerships in transport. *Public Money and Management*, 19, 1999.

[12] Steve Harris, Jim Davies, Charles Crichton, Aadya Shukla, and Jeremy Gibbons. Metadata Standards for Semantic Interoperability in Electronic Government. In *ICEGOV 2008*, 2008. to appear.

[13] S.A. Hazlett and F. Hill. E-government: the realities of using it to transform the public sector. *Managing Service Quality*, 13, 2003.

[14] R. Heeks. Success and failure rates of eGovernment in developing/transitional countries: Overview. `www.egov4dev.org/sfoverview.htm`, 2003.

[15] Marijn Janssen and Anthony Cresswell. Enterprise architecture integration in e-Government. In *Proceedings of the 38th Hawaii International Conference on System Sciences*, 2005.

[16] Jacek Kopecký, Tomas Vitvar, Carine Bournez, and Joel Farrell. SAWSDL: Semantic Annotations for WSDL and XML Schema. *IEEE Internet Computing*, 11(6):60–67, 2007.

[17] J. Langford and J. Roy. E-government and public private partnerships in canada when failure is not an option. *International journal Of Electronic Business*, 4, 2006.

[18] M.S. Laskowaski. The impact of electronic access to government information: What users and document specialists think. *Journal of Government Information*, 27, 2000.

[19] K. Layne and J. Lee. Developing fully functional e-government: A four-stage model. *Government Information Quarterly*, 18:122–136, 2001.

[20] Commission of the European Communities. Role of e-government for europes future. Technical report, European Union, 2003.

[21] Modinis Project Partners. Breaking Barriers to eGovernment. `www.egovbarriers.org/`, 2007.

[22] PCIP. Pacific council working group report. `http://www.pacificcouncil.org`.

[23] A. Salminen. Building Digital Government by XML. In *Proceedings of the 38th International Conference on System Sciences*, page 122b, 2005.

[24] H. Jochen Scholl. E-Government: A special case of ICT-enabled business process change. In *36th Hawaii International Conference on System Sciences (HICSS36)*, Waikoloa, HI, 2003.

[25] H.J. Scholl. Electronic government: Information management capacity, organizational capabilities, and sourcing mix. *Government Information Quarterly*, 23, 2006.

[26] UK Cabinet Office. British Standard Address Specification, BS7666. `http://www.govtalk.gov.uk/`.

[27] UK National Audit Office. Child Support Agency—Implementation of the Child Support Reforms, 2006.

[28] UK National Audit Office. The National Programme for IT in the NHS: Progress since 2006, 2008.

[29] W3C. Simple knowledge organization system. `http://www.w3.org/2004/02/skos/`.

[30] M. Wimmer and E. Tambouris. Online one-stop government. In *Information Systems: The e-Business Challenge*, pages 117–130, 2002.

[31] Maria Wimmer, Cristiano Codagnone, and Marijn Janssen. Future e-Government research: 13 research themes identified in the eGovRTD2020 project. In *Proceedings of the 41st Hawaii International Conference on System Sciences*, 2008.