Building a Sustainable Information System in the Domain of Chemical and Biological Safety

Matthias Menger, Patrick Ackermann,1
Thomas Bandholtz, Rainer Jaspert, Antoine Logean2

Abstract
This article gives a big picture of an innovative strategy to handle change in an application which supports assessment processes related to legal obligations of the Federal Environment Agency, Division IV Chemical Safety. Changes affect the data structures that describe the environmental impact of chemical. The solution is based on a data type construction kit, a Domain Specific Language (DSL) on top of XML schema, and dynamic XML processing. The solution enables modifications of the data structures at runtime, even without restarting the application. We do not claim a multiple-purpose solution, but one which is tailored to the common patterns of daily work in Division IV Chemical Safety. This is exactly why it may serve as a blueprint for several similar application challenges in a different domain context.

1. Introduction
Division IV3 of the Federal Environment Agency (Umweltbundesamt, UBA) is concerned with chemical safety and so with the protection of human health and environment against hazardous substances. Division IV investigates and assesses environmental risks of substances and preparations. If risks exist, they give recommendations and develop measures to reduce these risks. This can impose a ban on manufacture or use of chemicals. Regulations on chemical safety are largely prescribed by European Union legislation. Assessment methods are widely harmonized throughout Europe. The assessment results and management decisions have to be discussed, implemented and enforced at European level. The most important examples of these regulations are:

- Chemicals and Biocides Acts,
- Plant Protection Act,
- Washing and Cleansing Agents Act / EC Detergents Regulation,
- Medicinal Products Act,
- EC Registration, Evaluation, Authorisation and Restriction of Chemicals (REACH) and Regulation on Classification, Labelling and Packaging of Substances and Mixtures (CLP),
- Protection against Infection Act,
- Federal Water Management Act (§19g).

The European Chemical Agency (ECHA), located in Helsinki, Finland, has to manage the registration, evaluation, authorisation and restriction processes for chemical substances under REACH to ensure consistency across the European Union. Information Technology tools have been developed to store and exchange information and data on chemicals (Knetsch 2008):

1 Umweltbundesamt, Wörlitzer Platz 1, 06813 Dessau-Roßlau, Germany {patrick.ackermann|matthias.menger}@uba.de
2 innoQ Deutschland GmbH, Halskestr. 17, 40880 Ratingen, Germany {thomas.bandholtz|rainer.jaspert|antoine.logean}@innoq.com
3 http://www.umweltbundesamt.de/uba-info-e/c-fach4.htm
• REACh-IT\(^4\), an online platform to submit data and dossiers (pre-registration, registration, C&L notification, ...) on chemicals. It also allows the Agency and Member States authorities to review the dossiers. The Agency uses REACh-IT to make non-confidential information on chemicals accessible to public on its website.

• The International Uniform Chemical Information Database (IUCLID)\(^5\), essential tool to capture & store, submit, and exchange data on chemical substances stored according to the OECD Harmonised Templates (OHT).\(^6\)

The OHT provide “a guide for structuring data entry/database management systems for reporting a summary of the results of a test on a chemical to determine its properties or effects on human health and the environment (e.g., hydrolysis, skin irritation, repeat dose toxicity, etc.).” IUCLID implements this data model as an XML exchange format which can also be accessed by Web services.

In Germany, the UBA cooperates with other federal authorities, in particular the Federal Institute for Occupational Safety and Health (BAuA)\(^7\), the Federal Institute for Risk Assessment (BfR)\(^8\), the Federal Office of Consumer Protection and Food Safety (BVL)\(^9\), and the Federal Institute for Drugs and Medical Devices (BfArM)\(^10\).

Supporting these activities, there are three well known information systems which include a public version of substance data and information:

• Joint Substance Data Pool Federation/Länder (GSBL)\(^11\) (Menger 2007),
• Dioxin database of the Federal Administration and Länder governments\(^12\) (Knetsch 2007)
• Information System Ecotoxicology and Environmental Quality Targets (ETOX)\(^13\) (Schudoma 2008).

Not so well known – due to exclusive internal use -, since more than 15 years the agency maintains all data about substances and their environmental impact in order to support a reliable substance assessment and evaluation. This (mostly confidential) data is gathered from all kinds of regulation processes and covers all varieties of products and substance compositions, so information can be re-used and enhanced day by day.

As this system has come to age, the agency started conceiving of a completely new version of ICS in 2008. One of the basic questions was: why not use IUCLID instead? IUCLID had not been existent when UBA first developed ICS, but in 2008 it was available and comparatively mature. However, IUCLID did not cover all the regulation procedures to be supported, and there was too little impact on the directions of further development. In 2009, the agency started a development project with innoQ Deutschland GmbH as the contractor. The first question was: can we develop the ICSnew based on IUCLID? A thorough research resulted in two findings:

• from a technical point of view, ICSnew might be developed as an extension of IUCLID, either directly based on the IUCLID sources or by a set of plug-ins.
• from a legal point of view, the IUCLID License Agreement does neither allow distributing the source code to any party, nor any party developing plug-ins on its own.

---

\(^4\) https://reach-it.echa.europa.eu/reach/public/welcome.faces
\(^5\) http://iuclid.echa.europa.eu/
\(^6\) http://www.oecd.org/document/13/0,3343,en_2649_34365_36206733_1_1_1_1,00.html
\(^7\) http://www.baua.de/
\(^8\) http://www.bfr.bund.de/
\(^9\) http://www.bvl.bund.de/
\(^10\) http://www.bfarm.de/
\(^11\) http://www.gsbl.de/
\(^12\) http://www.pop-dioxindb.de/
\(^13\) http://webetox.uba.de/webETOX/index.do?language=en
While the technical approach appeared to be feasible with some functional and ergonomic drawbacks, the legal restriction caused a final decision of building the ICSnew from scratch, grounded on the IUCLID (OHT) data model with extensions.

Among the multiple challenges of such a system are:

- complex substance identity including chemicals, compositions, and organisms,
- integration with document and workflow management systems,
- generation of legally valid evaluation reports,
- data exchange with IUCLID and several further involved information systems,
- very high complexity of the impact information taken from studies,
- frequent modifications and extensions of data structures due to new scientific findings and new legislation or legal amendments.

Among these, we will focus on the dynamics of data structures in the following.

2. **Structural Dynamics of Impact Data**

Information on the impact of substances is extracted from studies and expressed as structured data. This has not always been the way to go: in the early days of assessment this information used to be attached in the format of the study itself, maybe even on paper. Even today, the OHT (which are not yet fully available for different details of data requested in the different regulation procedures) provide only structural templates as a recommendation for the input into free text fields for many complex properties. Modelling this information as structured data is a challenging process, and one may get into detail more and more over time.

This process has not come to an end, and maybe will never be finally settled. Apart from this, test scenarios and methods are evolving over time which results in new content types to be structured. Consequently the OHT and the IUCLID exchange format are continuously versioned\(^\text{14}\). The same applies to the data coming from different regulation procedures outside of IUCLID. Furthermore, the regulation procedures themselves and their structural reporting obligations are subject of change as well.

For such reasons, any information system which is based on a fixed data model would need to be continuously versioned as well. Rapid adaption of versioned data structures would suffer from time-consuming software update cycles and from contractor dependencies. In other words: versioning of data structures should be possible without any versioning of the application.

As far as procedures are handled by workflow features of the integrated Sharepoint 2010 we make use of the configuration options of this product (Stolle et al. 2011). This however covers a minor part of the challenge. Procedures also leave there footprint in the data model and user interface of the information system itself.

In the following we focus on a solution which is based on a data type construction kit, a Domain Specific Language (DSL) on top of XML schema, and dynamic XML processing. All this together enables modifications of the data structures at runtime, even without restarting the application.

2.1 **Necessary Limitations of the Data Construction Kit**

From a technical point of view, this cannot be implemented with a totally unbounded flexibility. The application software needs to provide methods with a certain degree of specialisation, so programmers need a minimum of assumptions about the basic data types and composition patterns to appear. ICSnew needs methods such as:

• display content of the data structure,
• edit content of the data structure with validation,
• provide pick-lists for values,
• import/export from/to various data formats,
• generate a data representation for the final report document.

In order to support such methods in a modular way, we need to agree on a set of basic data types. “Basic” does not mean simple, so these types may be, technically speaking, “complex” types such as a numerical value including unit of measurement, decimal places, maximum and minimum. Other examples are a value range or time span, text value with optional pick list, and so on. These types are hard coded in the application, including all the mentioned methods.

On top of these basic types we support domain specific applications of these types (domain basic types) with a rather universal meaning, such as “Temperature” or “Density”. Such application types can be created on-the-fly as they will inherit the methods of their respective basic types.

Basic types and domain basic types can be composed to build property types, which finally describe the data structure of real impact data (property definition).

Figure 1
Type hierarchy of the data construction kit
2.2 A Domain Specific Language (DSL) for Data Configuration

The original plans from 2008 made provisions for implementing those data structures in XML schema. While all these data types and properties certainly can be expressed and processed in XML, using a common XML schema editor for the configuration raises a serious problem: How can domain administrators be guided and controlled within the described boundaries of the ICSnew data construction kit? XML schema is a general purpose modelling language, and what we would need here is a subset with a fixed set of basic types and extension patterns. Consequently we would have to develop a dedicated validating parser to make sure the schema conforms to the ICS patterns. Thinking this over more closely, we found it more convenient to develop a Domain Specific Language (DSL) (Fowler 2010) for the purpose of data configuration. Using Eclipse’s Xtext and the integrated Extended Backus-Naur-Formalism (EBNF) it was comparatively easy to develop a tailor-made configuration language including a validating parser and a full featured Eclipse text editor (Figure 2).

3. XML Schema Evolution at Application Runtime

When it comes to application runtime, processing the DSL directly is not the best choice. Here XML has the clear advantage of a wide spread tool support. Due to the close integration of Xtext with the Eclipse

http://www.eclipse.org/Xtext/
Modelling Framework (EMF) this is not a problem, as the DSL can be easily converted into an XML schema. What we have so far is a tailored subset of XML schema without the drawbacks and risks of having this subset edited by domain administrators which are guided by best practise only. But how can the data be processed at runtime, and how can data structures change at runtime?

3.1 Hybrid Database Architecture

The first element of the solution is the hybrid database architecture. We have static (relational) data structures describing the general model of substances, evaluation procedures, users, and so on. When it comes to the impact data in this context, there is just a single XML BLOB column in the table for the complex property. So whenever the structure of the impact data changes, the relational table needs not to reflect this, every change happens within the apparently static XML BLOB cell.

This architecture is not new, but over many years there had been serious drawbacks in querying, indexing, and processing in general, so this solution was rarely implemented in heavy duty production systems. Revisiting the current state of implementation, we found that Oracle 11g is doing a good job today:
- the new Binary XML data type provides considerably improved performance compared to the traditional XMLType/CLOB;
- the new XMLIndex outclasses the old text index (CTXXPath);
- XML Streaming accelerates content processing of large XML documents.

3.2 Life Processing of XSD and XML Documents

How can an application process data when the data structures have not been known at design and build time? The solution is based on interpreting XML schema definitions at runtime. As mentioned earlier, we do not claim a general purpose solution – the basic types discussed in section 2.1 are hard coded by the application. This means: whenever they need to be changed or extended, the application needs to be changed or extended as well. However, based on 15 years of domain experience and two more years of project development (production scheduled for summer 2011), we are highly confident that the supported basic type set will serve the application for many years in the future. Data structure dynamics mainly is an issue of composition while a fix set of core elements is re-used again and again in all these compositions. Identifying these core elements certainly remains a challenge and needs some time. Having said this, let’s have a closer look to some processing examples.
ICSnew is a Web application, so the client interface is presented in XHTML. When we need to display a dialog for impact data input, we process the corresponding XML schema. Wherever the application detects one of the well-known basic types, it can simply unpack some well-prepared corresponding XHTML snippet and adjust the supported parameters as found in the schema. Some snippets include validation methods in Java Script. Dynamic pick-lists can be accessed directly from the client using AJAX requests to the server. When the user submits and the validation shows no errors, the content of the dialog is transformed into an XML document and stored into the binary XML column.

When the same content needs to be displayed – either in read-only or in editing mode, the XML schema information needs to be merged with the corresponding XML document instance while the XHTML page gets constructed. Implementing this is certainly more challenging than conventional programming based on fix data structures, but programmers will agree that it does not need witchcraft and wizardry. The most serious challenge is implementing this in a performant way. In this context we must be aware that ICS will have to serve some 50 or 100 users at the same time – not thousands or millions. Nevertheless, further scalability is an architectural issue, and we might report on this sometime later.

3.3 The Configuration Cycle

Having described the main aspects of the architecture, we like to recapitulate the complete picture briefly by describing the full configuration cycle with a simple diagram (Fehler! Verweisquelle konnte nicht gefunden werden.).

![ICS Data configuration cycle diagram](image)

Figure 4
ICS Data configuration cycle
The Domain Administrator receives information that some data structure needs to be modified or added (see section 3.4 for a closer description of patterns of change).

He starts the DSL editor (section 2.2) from the administrator’s workbench and does the necessary editing.

He saves the DSL instance (new or modified), and the administrator’s workbench converts it into an XML schema document.

He opens the corresponding application HTML-dialog, generated from the new XSD, in preview mode and does some necessary testing (insert new data, save, display again, modify, export, etc.).

When everything works fine, he triggers the production release.

Start again with (1).

All this happens at application runtime without any interruption of operation. The application does not even need to be restarted.

3.4 Patterns of Change to be Supported

So far we omit one crucial question: what happens to the existing data when its data structure gets changed? Does it need to be converted so it conforms to the new schema? We investigated the common patterns of change and selected a downwards compatible versioning strategy. This means: existing data remains valid even when the schema gets versioned. We briefly describe the supported cases and the solution strategy.

Case 1: Names of properties change. Visual names are not used as element or attribute names in the schema. They are stored in annotations. This is why the XML document is unconcerned of such changes. The visual names are solely stored in the schema.

Case 2: Sequence of properties changes. Same solution as in case 1: the visual solution is solely stored in XML schema annotations. The xsd:sequence as it appears in the schema document is ignored by the application whenever property values are displayed for human reading.

Case 3: Additional property added. If this property is not mandatory, there is no problem. If it is mandatory, the scheme needs to provide a default value. In ICSnew most properties are not mandatory.

Case 4: A property is expired. This wording implies that (for legal documentation purposes) existing property values are not deleted. Such a property is annotated with a timestamp in the schema. The application still displays such properties, but does not allow editing any more.

Case 5: Existing generic properties are refined and split into multiple specialised properties. This is the most complex case. The solution combines case 4 and case 3: the generic property is declared to be expired, and the specific properties are added. If the existing generic property data needs to be converted to a corresponding specialised data type, we found that there is no general way to do this automatically. Each case follows individual rules, and mostly some new data elements need to be specified by domain experts. Such a specialising conversion needs to be handled outside the ICSnew application, but possibly based on the XML representation of the legacy data, finally importing the modified XML documents into the new properties.

4. Conclusions and outlook

What has been described here gives a big picture of an innovative strategy to handle change in an application which supports assessment processes related to legal obligations of the Federal Environment Agency.
We restricted ourselves to changes affecting the data structures that describe the environmental impact of chemical substances. We did not talk about change of the evaluation procedures or the organizational structure of the agency. These are different issues which may be discussed in further contributions.

The solution is based on a data type construction kit, a Domain Specific Language (DSL) on top of XML schema, and dynamic XML processing.

The solution enables modifications of the data structures at runtime, even without restarting the application.

We do not claim a multiple-purpose solution, but one which is tailored to the common patterns of daily work in Division IV Chemical Safety. This is exactly why it may serve as a blue print for several similar application challenges in a different domain context. Real-world dynamics of a specific domain context can be handled, general-purpose dynamics lead to shallow solutions which will fail in concrete scenarios due to abstraction.

The solution has been successfully implemented in several iterations so far, having reached the first proof-of-concept in June 2010. Several issues had to be solved before ICSnew gained production maturity (scheduled for autumn 2011):

- basic performance is promising, but needs optimization (which is not a surprise); scalability is not crucial in this case, but an architectonical issue to be refined (once an application has proved successful, it will have to scale …);
- we have started with medium complexity of data structures, solved the general problems of the approach in general, and proceeded to high complexity and more details over the next months;
- user experience concerning the DSL was in an early stage. In the original plans of the UBA, domain administrators have been assigned to XML schema editing, which certainly is more tedious than editing the DSL. Initially we had planned a fallback of building a graphic configuration wizard on top of the DSL. This would not affect the DSL itself; such a wizard would just be another editor, finally writing the same DSL text format to disc. But finally, the administrators got used to the text-based editor and regarded the wizard a waste of restricted project resources.

Bibliography

http://my.safaribooksonline.com/9780132107549

http://iai-uierv1.iai.fzk.de/ictensure/site?mod=litdb&subject=art&pid=L809E06E9&action=detail

http://iai-uierv1.iai.fzk.de/ictensure/site?mod=litdb&subject=art&pid=L473353D5&action=detail

http://iai-uierv1.iai.fzk.de/ictensure/site?mod=litdb&subject=art&pid=LB1CB0828&action=detail

http://iai-uierv1.iai.fzk.de/ictensure/site?mod=litdb&subject=art&pid=L36114871&action=detail

schutz“, Dessau, 2008.

http://iai-uiserv1.iai.fzk.de/ictensure/site?mod=litdb&subject=art&pid=L49E8DBE7&action=detail