Designing Universal Chemical Markup (UCM) through the reusable methodology based on analyzing existing related formats

Background: In order to design concepts for a new general-purpose chemical format we analyzed the strengths and weaknesses of current formats for common chemical data. While the new format is discussed more in the next article, here we describe our software tools and two stage analysis procedure that supplied the necessary information for the development. The chemical formats analyzed in both stages were: CDX, CDXML, CML, CTfile and XDfile. In addition the following formats were included in the first stage only: CIF, InChI, NCBI ASN.1, NCBI XML, PDB, PDBx/mmCIF, PDBML, SMILES, SLN and Mol2.

Results: A two stage analysis process devised for both XML (Extensible Markup Language) and non-XML formats enabled us to verify if and how potential advantages of XML are utilized in the widely used general-purpose chemical formats. In the first stage we accumulated information about analyzed formats and selected the formats with the most general-purpose chemical functionality for the second stage. During the second stage our set of software quality requirements was used to assess the benefits and issues of selected formats. Additionally, the detailed analysis of XML formats structure in the second stage helped us to identify concepts in those formats. Using these concepts we came up with the concise structure for a new chemical format, which is designed to provide precise built-in validation capabilities and aims to avoid the potential issues of analyzed formats.

Conclusions: We believe our analysis methodology is potentially highly reusable and could be easily adapted even for domains outside the chemistry area. It is because the methodology and software tools will need only few changes, although analyzed formats and software quality requirements for a format will differ according to the given domain.

PeerJ PrePrints

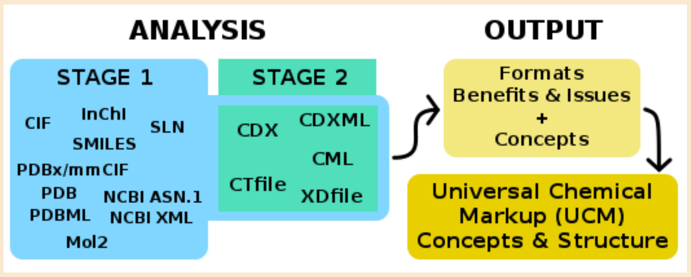
PeerJ PrePrints

Designing Universal Chemical Markup (UCM) through the reusable methodology based on analyzing existing related formats

Jan Mokrý[1] **and Miloslav Nič**[2]

[1]**Department of Inorganic Chemistry, University of Chemistry and Technology Prague, Technická 5, 166 28, Prague 6, Czech Republic**
[2]**Department of Software Engineering, Czech Technical University in Prague, Thákurova 9, 160 00, Prague 6, Czech Republic**

ABSTRACT



Background
In order to design concepts for a new general-purpose chemical format we analyzed the strengths and weaknesses of current formats for common chemical data. While the new format is discussed more in the next article, here we describe our software tools and two stage analysis procedure that supplied the necessary information for the development. The chemical formats analyzed in both stages were: CDX, CDXML, CML, CTfile and XDfile. In addition the following formats were included in the first stage only: CIF, InChI, NCBI ASN.1, NCBI XML, PDB, PDBx/mmCIF, PDBML, SMILES, SLN and Mol2.

Results
A two stage analysis process devised for both XML (Extensible Markup Language) and non-XML formats enabled us to verify if and how potential advantages of XML are utilized in the widely used general-purpose chemical formats. In the first stage we accumulated information about analyzed formats and selected the formats with the most general-purpose chemical functionality for the second stage. During the second stage our set of software quality requirements was used to assess the benefits and issues of selected formats. Additionally, the detailed analysis of XML formats structure in the second stage helped us to identify concepts in those formats. Using these concepts we came up with the concise structure for a new chemical format, which is designed to provide precise built-in validation capabilities and aims to avoid the potential issues of analyzed formats.

Conclusions
We believe our analysis methodology is potentially highly reusable and could be easily adapted even for domains outside the chemistry area. It is because the methodology and software tools will need only few changes, although analyzed formats and software quality requirements for a format will differ according to the given domain.

Keywords: chemical formats analysis, reusable methodology, designing UCM, UCM concepts, utilizing XML benefits

PeerJ PrePrints

BACKGROUND

Nowadays various chemical data formats exist. Some underwent a long development, other emerged more recently, as can be seen by comparing the older list of proposed chemical Multipurpose Internet Mail Extensions[1] with newer online listings.[2,3] Many chemical formats are tailored for specific needs in the given domain of chemistry. Other are for more common chemical data and as such enable the recording of chemical structures, sometimes with reactions, properties and additional data. Our main goal was to design concepts for a new general-purpose chemical format, which would combine the strengths of these formats for common chemical data, while avoiding their weaknesses where possible.

We decided to explore the idea of a new chemical format, because we noticed that currently the widely used general-purpose chemical formats often provide only limited built-in validation capabilities. Details about validation in various chemical formats are described later in the Benefits and issues analysis results (see ISSUES 8, 9 and 10) and in additional file 2. Since we have not found an in-depth comparison of current general-purpose chemical formats in scientific literature, we chose to analyze at least widely compatible formats with sufficient expressive power to capture common chemical data. We usually skipped specialized chemical formats, such as those limited only to a specific chemical drawing or viewing software, or those specific for computational chemistry programs. Instead, our search focused especially on formats that enable effective processing and publication of common chemical data in current multi-platform computing environment glued together by the Internet. The selection of chemical formats was guided by the following constraints:

- The format should offer functionality that enables at least the recording of chemical structures and ideally also reactions and properties.
- Functionality of the format should not be limited to a specific chemical software tool or a specific area of chemistry. (Formats supported by multiple software tools preferably from different vendors were favored.)
- Both format and software tools, which support it, should not be obsolete.
- Specifications of the format should be available at the main website of the format or published by a scientific journal.

To design concepts for the new format, the selected chemical formats were analyzed to discover their benefits and issues together with the main currently implemented concepts for common chemical data. Both XML and non-XML formats were analyzed, as we wanted to verify whether XML offers real benefits for chemical formats. It was important for main design decisions concerning the new format in our mind, since the usage of XML technology theoretically brings various advantages.[4] For example the basic built-in validation should be offered automatically by XML, because a schema, which defines the structure of some XML format, can be used to validate that data conform to such a format.

CHEMICAL FORMATS ANALYSIS IMPLEMENTATION AND PROCEDURE

The analysis procedure we devised required gathering, processing and storing of important data about each analyzed format. When gathering initial information our search revealed also non-chemical, but related and potentially useful, formats (e.g. those that could be combined with or integrated into a new chemical XML format). Thus, we wanted to store data about both related and analyzed formats in a readily usable form.

This was done using a combination of custom XML files and Google Spreadsheets (a part of Google Docs web-based office suite interconnected with Google Drive cloud-based storage[5,6,7]). It enabled effective processing, updating and storage of all data gathered and produced during the analysis. At first we took advantage of Google Data API (Application Programming Interface) and created Python modules for conversion and synchronization of data between the Google Drive and local XML files. Later updates to Google Drive made it possible to retire the synchronization module and remove the dependency on Google Data API (access to Google Data API provided Gdata Python Client Library[8]). However, the modular approach helped us to quickly adapt and instead rely on the conversion module when synchronizing our data between spreadsheet applications and XML files. In this way the data were stored in a compatible format that could be fed to automatized processing and output system using the standard XML tool chain (i.e. the infrastructure for XML available in programming languages and software tools, especially for parsing, navigation, transformation and validation of XML documents). Such mechanism has proven quite useful for converting data to other formats when a need arose. For example when LaTeX sources for the publishing of analysis results have been required it was an easy task to extend our Python modules with appropriate methods allowing to quickly generate an output in the desired format. As the analysis progressed, we integrated our software tools into a single reusable package called DATA FORMATS ANALYZER.

Requirements for modern chemical format

To identify advantages and drawbacks of given chemical formats in a more repeatable way, the comparison had to be based on objective criteria. We have devised a set of precisely defined requirements to evaluate not only format functionality, but also its other qualities. Our requirements are ordered into groups inspired by appropriate software quality attributes. The idea of software quality attributes was adapted from the book Software Architecture in Practice,[9] but to avoid confusion we include a detailed description for each requirements group:

- FUNCTIONALITY – Ideally a chemical format should offer not only sufficient expressiveness for storage of data, but also mechanisms to validate and annotate these data. Thus, we assessed what functionality the analyzed format provides for storage, validation and annotation of common chemical data and how efficiently this functionality is implemented. In other words, our functionality requirements are:

    1. The format should provide well defined functionality for common chemical data that enables recording of structures, reactions and properties.
    2. A strict validation of the format structure and stored data should be available to minimize the probability of an error.
    3. Adequate annotation possibilities should be included in the format (i.e. highly compact formats can rely on external annotations, while less compact formats should support both plain text and some widely used markup for additional formatting inside the annotations).

- MODIFIABILITY – We think a chemical format should be flexible and easily extensible to facilitate efficient interaction with modern online environment and other data formats. Because of that we investigated if data stored in the given format could be transformed into a form suitable for web browsers and if any modifications are necessary to make the format inter-operable. Extensibility of the format was evaluated too. This means our modifiability requirements mostly focus on how difficult it is to modify either the given format as such (i.e. extend it) or its instance containing data (i.e. transform it):

    4. Flexible interaction with modern web browsers should be supported by the format (i.e. transformation of chemical data from the format into a form viewable by web browsers should be as easy as possible, preferably using the built-in mechanisms available in current web browsers).
    5. The format should interact well with other data formats (i.e. both transformation of chemical data from the format into other data formats and directly combining other scientific data with chemical information stored in the format should be reasonably simple).
    6. Format functionality should be easily extensible in the future.

- USABILITY – Both users and potential developers of chemical software appreciate a plainly usable chemical format. Such a format should ideally be well structured, readable and properly documented to be searchable, easy to learn, simple to use and straightforward to implement. We also think that average users should not need to solve problems like: the file created by software A is not working in software B although both software A and B claim to support the format. Therefore, precise implementation in chemical software according to unambiguous specification is essential to maintain compatibility among different software tools using the format. Consequently, the usability requirements are as follows:

PeerJ PrePrints

PeerJ PrePrints

7. The format should make it easier for users to search for chemical data by storing the data in a structured way readable for both humans and computers.

8. It should be adequately simple to learn, use and implement the format functionality – without ambiguous, unnecessary and redundant parts.

9. The functionality of the format should be well documented – with detailed unambiguous description and examples.

- PERFORMANCE – The current computer hardware can overcome most performance problems an average user could have with regards to efficiency of chemical data formats. However, when the format is considered for the storage of large data collections (e.g. some big collection of chemical structures obtained for further research from chemical databases), performance problems especially with memory efficiency can occur. We opted against measurements of absolute memory efficiency, because it would be prohibitively difficult to cater for different functionality offered by various chemical formats. Instead our only performance requirement is:

10. A reasonable memory efficiency should be offered by the format with regards to its functionality.

- AVAILABILITY – In our opinion the ideal chemical format should be accessible to many users with different devices to cope with current heterogeneous computing environment. Since the number of devices and software tools available for the format is likely to change in the future, we focused more on whether the format is usable on different computing platforms. But besides the cross-platform technical compatibility, the ideal chemical format also should not be released under some restrictive proprietary license that limits its usage and development by the others. Only then the format can achieve stable future-proof availability as its development can continue even without the original developers. Thus, our availability requirements are the following:

11. Multiple computing platforms, including at least Windows, Mac and Linux, should be supported by the format.

12. The format should be publicly available under a clearly indicated license, which permits the free usage and implementation of the format in software tools and also grants the freedom to further develop the format.

These requirements complement each other and one needs to look at the whole set together when comparing various formats. Because we grouped the requirements according to the software qualities represented by them, strengths and weaknesses of analyzed chemical formats can be discerned more easily.

Analysis procedure

The overall process of analysis consisted of two main stages:

- STAGE 1:

1. Obtain all important data about the format.

2. Convert the data to appropriate format suitable for exporting to Google Spreadsheets and synchronize the data with Google Drive.

3. Correct and update the data in Google Spreadsheets using the Google Docs web interface and then convert and synchronize the data with the local backup.

4. Select the format for the second stage of the analysis, or exclude it depending on its functionality and main strengths and weaknesses.

- STAGE 2:
    5. Test and evaluate the format and describe its benefits and issues in detail.
    6. For XML formats identify and analyze main concepts for common chemical data.

In the first stage during step 1 we gathered various basic information about each format matching the constraints listed in the Background, especially including:

- Name of the format together with its abbreviation.
- Date the format was last updated.
- Current version of the format.
- Link to the main website of the format.
- Namespace of the format – only for XML formats.
- Link to the schema of the format – only for XML formats.
- Schema language used for defining the format – only for XML formats.
- Names of main software tools, which are using the format.
- Keywords describing the format and its functionality.
- Additional relevant links found to be related to the format.
- Short description of the format to quickly introduce it to the reader.

Additionally, for XML formats in step 1 we wrote the Python module for extracting data about format XML structure directly from its schema. This module gathers data about basic building blocks of any XML format defined using XSD (World Wide Web Consortium XML Schema Definition Language). The data output goes into our custom XML files suitable for further processing and also into an interactive XHTML (Extensible Hypertext Markup Language) reference providing the overview of format XML structure. Direct support for RELAX NG (Regular Language for XML Next Generation), DTD (Document Type Definition) or other XML schema languages could be added, however we found it easier to use Trang instead. It is a tool that can convert between common XML schema languages.[10] This made it possible to simply convert the other XML schema languages to XSD, from which our module can extract relevant data about each attribute, element and type as follows:

- FOR ATTRIBUTES:
    - Name of the attribute.
    - Names of parent elements of the attribute.
    - Description of the attribute from documentation annotations.
- FOR ELEMENTS:
    - Name of the element.
    - Names of all attributes of the element.
    - Names of all children elements of the element.
    - Names of all parent elements of the element.
    - Description of the element from documentation annotations.

PeerJ PrePrints

- FOR TYPES:
  - Name of the type with prefix ”C-” or ”S-” to distinguish complex and simple types.
  - Names of all attributes of the type – only for complex types.
  - Names of all children elements of the type – only for complex types.
  - Names of all attributes using the type – only for simple types.
  - Names of all elements using the type.
  - Description of the type from documentation annotations.

In steps 2 and 3 all obtained data about the format were processed using our Python modules except for the manual corrections done via the Google Docs web interface. During step 3 local XML files and optionally also interactive XHTML references were updated (again using our Python modules) with modified data from Google Spreadsheets. For XML formats the interactive XHTML references simplified the analysis by providing the overview of format XML structure. In addition, if the descriptions from the format schema are included, the references offer an advanced starting point for creating useful resources for the particular format. Further information about the references and preparing such resources from them is described in additional file 1.

The data from the first stage of the analysis enabled us to select formats for the second stage in step 4. Besides that, with our Python module for conversion, the information obtained in the first stage were reformatted to serve as an overview of currently established formats for common chemical data.

During the second stage we carried out the detailed analysis of formats with the most general-purpose chemical functionality. This was done through the Benefits and issues analysis and Concept analysis in steps 5 and 6 respectively.

Benefits and issues analysis

For each format analyzed in detail the benefits and issues analysis in step 5 was basically about extending the set of main strengths and weaknesses found for that format in step 4. Thus, as in step 4, we assessed how the format fulfills our Requirements for modern chemical format. When necessary the format functionality was also tested in practice. Compared to step 4 from the first stage, in step 5 we examined the formats very thoroughly to obtain much more detailed sets of benefits and issues.

Concept analysis

The concept analysis in step 6 was possible, because all XML formats store data consistently in a structured way with one standard XML markup syntax (as defined in the official specifications[11,12]). Furthermore attributes and elements in XML markup can be viewed as parts, which provide specific portions of overall functionality in some XML format. Using a repeatable procedure based on such views, we assigned rudimentary concepts to the attributes and elements from CDXML, CML and XDfile formats. Then, the similar rudimentary concepts were repeatedly merged and refined until we obtained the set of sufficiently general and yet still descriptive main concepts. Subsequently the concepts in this set were organized into the following broad categories: chemical concepts with prefix ”C-” (e.g. C-BOND), data concepts with prefix ”D-” (e.g. D-METADATA) and general concepts with prefix ”G-” (e.g. G-IDENTIFIER). Since additional prefixes can be envisaged to be useful in studies of different domains, we believe that categorization makes concepts clearer and will simplify their reuse.

The main goal of concepts for a given domain is to group similar functionality together and to identify high priority constructs, which can be compared across various XML formats and should be supported when developing a new format. By utilizing the procedure described at the beginning of this section, concepts from XML formats are obtained relatively effectively, because of suitable structure of these formats and their standardized nature enforced by XML syntax. The concepts can be then used for the development of both XML or non-XML formats. In the case of a new non-XML format one just needs to implement the concepts in accordance with the requirements of the given non-XML syntax. Our experience with the concept analysis approach leads us to firm believe that it may be useful also outside chemistry area. To adapt our approach to a different domain requires the selection of suitable concept categories and identification of appropriate concepts in relevant XML formats, but the methodology remains the same.

This ensures the concept analysis can be adapted to match the needs of its user – be it a developer of a new format or a researcher evaluating XML formats from given domain. If just a rough analysis of available XML formats is required, quickly establishing a lower number of more general concepts helps to speed up the whole process and still allows identification of the main functionality of analyzed formats. More detailed analysis requires higher number of specialized concepts and can be based on previous rough analysis (i.e. general concepts from rough analysis could be divided to get the specialized concepts).

CHEMICAL FORMATS ANALYSIS RESULTS

Because the analysis results are quite detailed, we categorized them into three sections. This enabled us to accurately describe our findings, especially for the in-depth analysis in the second stage. Thus, while the Overview of analyzed formats contains information accumulated during the first stage of the analysis, next two sections, the Benefits and issues analysis results and Concept analysis, include findings from the second stage.

Overview of analyzed formats

For each format analyzed in detail we offer the overview of basic information, gathered in the first stage of the analysis.

ChemDraw Exchange (CDX) and ChemDraw Exchange Markup Language (CDXML)

CDX is the native file format of ChemDraw, and is guaranteed to save anything drawn in ChemDraw without a loss of data.[13] At the same time, however, its architecture was carefully designed to make it a flexible and general-purpose chemical format.[13] Because of its ability to incorporate custom information, and because it is in the public domain, CDX has been adopted by the U.S. Patent Office as its standard chemical format.[13]

CDXML is a variant of CDX that complies with the XML specification.[14,15,13] Similarly as CDX it is designed to save anything drawn in ChemDraw without a loss of information and also as a general-purpose chemical format.[13]

Updated: CDX: UNAVAILABLE; CDXML: 2003-04-23
Version: CDX: UNAVAILABLE; CDXML: 4.8
Website: http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/index.htm
Namespace: Default empty namespace, because no specific namespace is defined. (only for CDXML)
Schema: http://www.cambridgesoft.com/xml/cdxml.dtd (only for CDXML)
Schema – Language: DTD
Software (CDX and CDXML): ChemDoodle,[16,17] ChemDraw,[15,13] ChemSketch,[18] Instant JChem,[19] Marvin Applets, Marvin Beans,[20] Open Babel[21,3]
Software (CDX): CDXHexDumper[13]
Keywords: CDX, ChemDraw Exchange, CDXML, ChemDraw Exchange Markup Language, chemical graph, 2D structure, 3D structure, substructure, crystal structure, polymer structure, structure property data, chemical reaction, reaction property data, chemical query, spectroscopy data
Links: http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/IntroCDX.htm
http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/IntroCDXML.htm

Chemical Markup Language (CML)

CML covers disciplines from macromolecular sequences to inorganic molecules and quantum chemistry.[22,23] It provides a general-purpose chemical functionality for working with atoms, molecules, spectra and other analytical and crystallographic information.[22,23,24] CML can offer extensibility for the future[22,24] and can also import legacy files with any desired chemical ontology from other software without information loss.[22,24]

From the first version officially published in 1999[22,25,26] CML underwent the long evolution.[27] The format was redefined using XSD and modularized starting with CML 2.[28,29] This created a family of markup languages with separated schemas: CMLComp for computational chemistry; CMLCM for condensed matter; CMLCore for molecules and their structure; CMLQuery for queries; CMLReact

PeerJ PrePrints

for reactions and their mechanisms;[30] CMLSpect for spectroscopy;[31] Polymer Markup Language for representation of polymers;[32] and Scientific, Technical and Medical Markup Language for non-chemical concepts (e.g. numeric data, scientific units, metadata/dictionaries, etc.).[33] With versions 2.4 and 3 it seems that CML reintroduced the concept of single schema.[34,35]

Updated: 2012-02-20
Version: 3
Website: http://www.xml-cml.org
Namespace: http://www.xml-cml.org/schema
Schema: http://xml-cml.org/schema/schema3/schema.xsd
Schema – Language: XSD
Software (CML): Avogadro,[36,37] ChemDoodle,[16] ChemSketch,[18] JChemPaint,[38,39] Jmol,[40,39] Jumbo,[41,26] Marvin Applets, Marvin Beans,[20] Open Babel[21,3]
Keywords: CML, Chemical Markup Language, CMLComp, CMLCM, CMLCore, CMLQuery, CMLReact, CMLSpect, PML, Polymer Markup Language, STMML, Scientific, Technical and Medical Markup Language, chemical graph, 2D structure, 3D structure, substructure, crystal structure, polymer structure, structure property data, chemical reaction, reaction property data, chemical query, computational chemistry data, spectroscopy data
Links: http://sourceforge.net/projects/cml
http://cml.sourceforge.net
http://cml.sourceforge.net/schema

Chemical Table File (CTfile) and XML Data File (XDfile)

CTfile formats are widely used in the chemical software industry and are suitable for chemical structures, reactions, and structure properties data.[42,43,44] This family of chemical text formats includes: Molfile, RGfile, Rxnfile, SDfile, RDfile and XDfile.[43,44]

XDfile is based on XML and can contain structures or reactions that use any of the CTfile formats, Chime strings, or SMILES strings.[44] Thus, XDfile inherits the functionality of the embedded format for the structure or reaction.[44]

Updated: CTfile: 2011-12-02; XDfile: 2011-03-28
Version: CTfile: V3000; XDfile: UNAVAILABLE
Website: http://accelrys.com/products/informatics/cheminformatics/ctfile-formats/no-fee.php
Namespace: http://my-company.com/xdfile (only for XDfile)
Schema: Available after submitting your personal information using the registration form at the format website. (only for XDfile)
Schema – Language: XSD
Software (CTfile and XDfile): Accelrys Draw, Accelrys Isentris[45]
Software (CTfile): ChemDoodle,[16] ChemSketch,[18] JChemPaint,[38] JME Molecular Editor,[46,47] Jmol,[40,39] Marvin Applets, Marvin Beans,[20] Open Babel,[21,3] PerlMol,[48] PyMOL,[49,50] RasMol[51]
Keywords: CTfile, Chemical Table File, XDfile, XML Data File, Molfile, RDfile, RGfile, Rxnfile, SDfile, chemical graph, 2D structure, 3D structure, substructure, polymer structure, structure property data, chemical reaction, reaction property data, chemical query
Links: http://web.uni-plovdiv.bg/ksx/students/ISIS/ctfile.pdf

Information obtained for the formats excluded form the second stage is in additional file 2. Here we just briefly list these formats and the reasons for their excluding:

- Crystallographic Information File (CIF) is the standard format for recording crystallographic information for chemical structures.[52,53,54,55,56] Even macromolecular structures are supported by the closely related macromolecular CIF format,[57,58] but other general-purpose chemical functionality is not offered (e.g. there is no support for reactions). Thus, we did not include CIF in the second stage of the analysis.

PeerJ PrePrints

- International Chemical Identifier (InChI) provides a machine-readable identifier that encodes the molecular information of chemical substances and can be used in printed and electronic data sources.[59,60,61] This is certainly very useful for searching chemical structures in databases and on the web in general. However, InChI does not match our criteria for a general-purpose chemical format suitable for further analysis in the second stage. For example it lacks the support for storing reactions and structures with properties or precise coordinates in 2-dimensional or 3-dimensional space.

- NCBI Abstract Syntax Notation 1 (NCBI ASN.1) and NCBI Extensible Markup Language (NCBI XML) can be regarded as a group of ASN.1 and XML formats for various NCBI (National Center for Biotechnology Information) data.[62,63,64,65] Because NCBI data include also nucleotide and protein sequences, biochemical structures, genomes, MEDLINE records and so on,[62] ASN.1 and XML formats for chemical data are only one part of significantly larger specification. For NCBI XML these specifications are automatically generated and it results in various limitations, as we describe in additional file 2. This and some missing general-purpose chemical functionality (e.g. no support for reactions) meant we excluded NCBI ASN.1 and XML formats from the second stage of the analysis.

- Protein Data Bank (PDB), Protein Data Bank Exchange Dictionary Macromolecular Crystallographic Information File (PDBx/mmCIF) and Protein Data Bank Markup Language (PDBML) store data from the Worldwide Protein Data Bank archive.[66,67,68,69,70] For each deposition these data include crystallographic information, primary and secondary structure information, sequence database references, where appropriate, and ligand and biological assembly information, details about data collection and structure solution, and bibliographic citations.[66,67] Our reasons for not analyzing PDB, PDBx/mmCIF and PDBML further were in part the same as for the CIF format and also similar to the case of NCBI ASN.1 and XML formats (e.g. there are similar issues with automatically generated PDBML specifications as for NCBI XML).

- Simplified Molecular Input Line Entry System (SMILES) with all its modified and extended versions (e.g. Daylight SMARTS and SMIRKS,[71] OpenSMILES,[72,73,74] etc.) and SYBYL Line Notation (SLN) can both record chemical structures, queries and reactions in a very compact line notation form.[75,71,76,77,78] Although these line notation formats offer quite rich functionality, they focus on the concise representation of chemical data. Therefore, SMILES or SLN cannot really replace the complete functionality of a general-purpose chemical format, when it comes for example to recording structures with various properties and additional data. Even though SLN supports custom attributes and can store structures with coordinates in 2-dimensional or 3-dimensional space, the authors of SLN correctly point out that the format may loose its simple interpretability and readability in such complex use cases.[78] Consequently we regard SMILES and SLN together with InChI as very useful formats, which can complement rather than replace the functionality of general-purpose formats for common chemical data. Since for the second stage we wanted only the most general-purpose chemical formats, SMILES and SLN were excluded.

- Tripos Mol 2 File (Mol2) provides portable representation of chemical structures used by SYBYL-X Suite.[79,80] We again found that some general-purpose functionality, like the support for chemical reactions, is missing, thus Mol2 format was not analyzed further.

Benefits and issues analysis results

Our assessments for each format analyzed in detail are categorized into benefits and issues in the following sections. We merged the instances, in which the similar benefit or issue affected several requirements, or was found in different formats. Thus, some benefits or issues are relevant for more than one format as is indicated in their section titles.

BENEFIT 1 (CDX, CDXML, CML, CTfile, XDfile)

Affects: Requirement 1 (FUNCTIONALITY)
Summary: Chemical structures with or without coordinates can be recorded.

PeerJ PrePrints

BENEFIT 2 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: Polymer structures and related data can be recorded.

BENEFIT 3 (CDX, CDXML, CML)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: Crystal structures and related data can be recorded.

BENEFIT 4 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: Various properties of chemical structures can be recorded.

BENEFIT 5 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: Chemical reactions and related data can be recorded.

BENEFIT 6 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: Various properties of chemical reactions can be recorded.

BENEFIT 7 (CML)
Affects: Requirement 1 (FUNCTIONALITY)
Summary: The format functionality is very flexible as data can be recorded with desired chemical ontology[22,24] and using various conventions.[22,28,81]

BENEFIT 8 (CDXML, XDfile)
Affects: Requirement 2 (FUNCTIONALITY)
Summary: The validation of format basic structure is supported and can be performed with standard XML validators.

BENEFIT 9 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 3 (FUNCTIONALITY)
Summary: Plain text annotations are supported.

BENEFIT 10 (CDX, CDXML, CML)
Affects: Requirement 3 (FUNCTIONALITY)
Summary: Additional features are supported for annotations.
Detail (CDX, CDXML): Additional formatting features for annotations are supported through the properties (in CDX) or attributes (in CDXML) describing the content visualization.
Detail (CML): Usage of other XML formats with unique namespaces is enabled in annotations. With these formats (e.g. XHTML, SVG (Scalable Vector Graphics), MathML (Mathematical Markup Language), etc.) additional formatting, visualization and other features can be provided.

BENEFIT 11 (CDXML, CML, XDfile)
Affects: Requirements 4 and 5 (MODIFIABILITY); Requirements 7 and 8 (USABILITY)
Summary: The standard XML tool chain (i.e. the infrastructure available in programming languages and software for XML processing, especially for parsing, navigation, transformation and validation of XML documents) can be used to process and implement the format more easily.

BENEFIT 12 (CDX, CDXML, CTfile, XDfile)
Affects: Requirements 4 and 5 (MODIFIABILITY)
Summary: It is possible to implement the automatized transformation of data from this format into other data formats, as its structure is quite precisely defined.
Detail (CDX, CTfile): For implementing the transformation various programming languages can be used. With sufficient effort one could even generate the web browser friendly preview of chemical data stored in the format (e.g. directly in a web browser by using JavaScript, or other ECMAScript implementation, to output XHTML with SVG or JavaScript rendering).

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1335v1 | CC-BY 4.0 Open Access | rec: 30 Aug 2015, publ: 30 Aug 2015
PeerJ PrePrints

Detail (CDXML, XDfile): Mechanisms specific for XML (e.g. XSLT (Extensible Stylesheet Language Transformations)) can be used in addition to various programming languages. This makes the implementation of such transformation easier. For example with XSLT only the rules for the transformation (i.e. the XSLT stylesheet) need to be defined, while a generic XSLT processing tool parses the input, modifies it and generates the output as specified by the rules in the XSLT stylesheet. The web browser friendly preview of chemical data stored in the format can be also generated directly in a web browser using XSLT together with mentioned possibilities of JavaScript, or other ECMAScript implementations.

BENEFIT 13 (CML, XDfile)
Affects: Requirement 5 (MODIFIABILITY)
Summary: The format has its unique namespace, so it can be directly combined with other XML data formats (e.g. XHTML, SVG, MathML, etc.) in complex XML documents.

BENEFIT 14 (CML)
Affects: Requirement 6 (MODIFIABILITY)
Summary: The format is relatively easily extensible, because it is based on XML and does not depend on any legacy non-XML format specifications.
Detail (CML): For example, new attributes and elements can be added to any XML format (not maintaining compatibility with some legacy non-XML format specifications) to extend its functionality without changing the existing attributes and elements. This avoids breaking the existing functionality of the format. And because attributes and elements also represent the particular functionality in XML formats, software can simply process just the attributes and elements representing the supported functionality.

BENEFIT 15 (CDXML, CML, XDfile)
Affects: Requirements 7 and 8 (USABILITY)
Summary: The XML markup briefly describes the stored data, so the format is more readable.
Detail (CDXML, CML, XDfile): It can help users to search the relevant data in this format as well as learn the format.

BENEFIT 16 (CTfile)
Affects: Requirements 8 and 9 (USABILITY)
Summary: The format is adequately simple to learn, use and implement, because it has precisely defined structure and its functionality is well described with examples in its documentation.

BENEFIT 17 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 10 (PERFORMANCE)
Summary: The format offers a reasonable memory efficiency with regards to its functionality.
Detail (CDX, CDXML, CML, CTfile, XDfile): Although the XML markup consumes some additional memory, current computer hardware easily compensates for it. However, note that the XML markup of some formats can affect memory efficiency more negatively in some cases, as we described in additional file 2.

BENEFIT 18 (CDX, CDXML, CML, CTfile, XDfile)
Affects: Requirement 11 (AVAILABILITY)
Summary: Specialized chemical software required for the practical usage of the format is available.
Detail (CDX, CDXML, CML, CTfile): We found such software at least for Windows, Mac and Linux platforms.
Detail (XDfile): We found such software at least for Windows platforms.

BENEFIT 19 (CDX, CDXML, CML)
Affects: Requirement 12 (AVAILABILITY)
Summary: The format is freely available with specifications.
Detail (CDX, CDXML): It is released into the public domain from its website.
Detail (CML): It is obtainable from its website as an open source software.

ISSUE 1 (CDXML)

Affects: Requirement 1 (FUNCTIONALITY); Requirements 7 and 8 (USABILITY)

Summary: Chemical information is mixed with embedded binary objects and data for content visualization as in corresponding binary CDX specifications.

Detail (CDXML): It may not be an issue for CDX format, which anyway will not be readable and searchable for users, because of its binary structure, and where no standard means exist for referring to external files or for describing content visualization externally (e.g. by CSS (Cascading Style Sheets)). But it negatively affects the usability of an XML format. The raw XML structure of the format is quite difficult for human users to understand and search. Additionally the format is more difficult to learn and implement with all its functionality. In our opinion binary objects should be included in appropriate external files or replaced by a textual form that can be better validated and manipulated by XML tools. Also the markup dealing with content visualization should be separated like in MathML or SVG. MathML offers the special presentation markup in a separate RELAX NG schema,[82] while SVG has optional presentation attributes separated in DTD modules by the functionality.[83] Both MathML and SVG documentations clearly separate the markup dealing with content visualization from the content markup.

ISSUE 2 (CDXML, CML)

Affects: Requirements 1 and 3 (FUNCTIONALITY); Requirement 8 (USABILITY)

Summary: The format contains some redundant parts, as can be seen from our Concept analysis.

Detail (CDXML, CML): Accumulation of redundant parts is sometimes inevitable when the format is being actively developed, because the new functionality is added while older parts cannot be removed immediately to maintain backwards compatibility. However, with increasing number of redundant parts the format can become ambiguous and confusing for both users and especially potential implementers. Consequently in a format with well defined functionality unnecessary and redundant parts should be kept at minimum. Thus, we describe in the Concept analysis section how some concepts could be implemented more effectively (i.e. without or at least with limited amount of unnecessary redundancy).

ISSUE 3 (CDX, CDXML, CTfile, XDfile)

Affects: Requirement 1 (FUNCTIONALITY)

Summary: Associating scientific units with the values of properties is not properly enforced by the format.

Detail (CDX, CTfile): It could be improved by adding a mechanism dedicated to defining scientific quantities and units.

Detail (CDXML, XDfile): For XML formats it is currently possible to integrate UnitsML (Units Markup Language), an existing mechanism dedicated to defining scientific quantities and units.[84] In the Concept analysis section we discus UnitsML further.

ISSUE 4 (CDX, CDXML, CTfile, XDfile)

Affects: Requirement 1 (FUNCTIONALITY)

Summary: Electrons participating in chemical bonds cannot be recorded.

Detail (CDX, CDXML, CTfile, XDfile): This prevents more precise description of chemical structures with complex delocalized or other bonding.

ISSUE 5 (CML)

Affects: Requirement 1 (FUNCTIONALITY)

Summary: The format functionality is defined in many cases deliberately fuzzily and the content model is mostly removed in version 3[34] that abandons any mandatory tree structures.

Detail (CML): It can be seen in version 3 or 2.4 of the format schema, where many parts are defined with minimal restrictions and often using lax validation mode. It is also reflected in documentation annotations of some attributes (e.g. *constraint*, *convention*, *duration*, *state*, *symbol*, etc.) or elements (e.g. *action*, *object*, *observation*, *system*, *title*, etc.) in schema version 3 or 2.4. In addition schema version 3 enables all elements to contain each other. In other words the format has no mandatory XML tree structure and the nesting of elements is left completely up to the user. As a result the format structure can be highly variable and usually there is more than one way of recording the particular chemical information. This makes the format confusing and its validation

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1335v1 | CC-BY 4.0 Open Access | rec: 30 Aug 2015, publ: 30 Aug 2015

more complicated (see ISSUES 10 and 13). Although there is an ongoing effort to avoid these problems (also to address external suggestions[85,86]) using a mechanism for denoting standard conventions,[22,28,81,87] the mechanism is still not fully implemented to address all complex problems stemming from the highly unrestricted and variable structure of the format. Initially in version 1 the usage of *convention* attribute was limited to certain elements.[87] With version 2, when the format was redefined using XSD instead of DTD,[28] the *convention* attribute started to be used on more elements as various communities adapted the format in their domains of chemistry.[87] In version 3 schema it is enabled for nearly all elements. The main issue is that until version 3 no official list of conventions was composed and users, including developers of chemical software, had to use their own conventions. While with version 3 schema users can still create and use custom conventions, the documentation now lists some official conventions[87] aimed at adding additional rules and bringing more order to the format. However, the listed conventions appear to be mostly unfinished. All are marked as draft recommendations and some are missing (see ISSUE 6). Until the necessary official conventions are developed and their validation is implemented, problems related to the very variable structure of the format (ISSUES 6, 10, 13 and 22) will remain and make it difficult to reliably work with this format.

ISSUE 6 (CML)

Affects: Requirement 1 (FUNCTIONALITY)

Summary: The format contains unfinished experimental parts and official conventions without marking the released versions as pre-release, experimental or unstable.

Detail (CML): Some unfinished or experimental parts can be found in version 3 or 2.4 of the format schema by checking the documentation annotations of some attributes (e.g. *countExpression*, *idgen*, *preserve*, etc.) or elements (e.g. *bandList*, *join*, *kpointList*, *potentialList*, *transitionState*, etc.). Also all official conventions introduced with version 3 are marked as draft recommendations. Probably because the significant amount of the format functionality is not included in these conventions. For example the molecular convention does not include the format support for chemical structures with complex delocalized or other bonding. And some conventions are clearly waiting to be developed (e.g. the convention dealing with the format support for recording crystallographic information or reactions etc.).

ISSUE 7 (XDfile)

Affects: Requirement 1 (FUNCTIONALITY); Requirements 4 and 5 (MODIFIABILITY); Requirements 7 and 8 (USABILITY)

Summary: The format does not provide any new functionality for storing chemical data, but instead it just inherits the functionality of the embedded format and adds various metadata.

Detail (XDfile): Especially in the case of embedded CTfile formats it can be more difficult to implement the automatized transformation of data from this format into other data formats. Because the large chunks of chemical information in one of the CTfile formats (e.g. Molfile, Rxnfile etc.) can be stored inside *field* elements, it is then more complicated to formulate detailed transformation rules for smaller chunks of information. In other words any detailed processing of the format (including the transformation into other formats) requires parsing the XML structure of this format and then also the structure of embedded non-XML formats. Furthermore the usage of embedded formats makes the raw XML structure of the format harder to understand and search for human users. Additionally if one is not already familiar with the embedded formats this format can be more difficult to learn and implement with all its functionality.

ISSUE 8 (CDX, CTfile)

Affects: Requirement 2 (FUNCTIONALITY)

Summary: No built-in validation capabilities are provided by this format and we also did not find standard software tools dedicated to validation of this format.

ISSUE 9 (CDXML, XDfile)

Affects: Requirement 2 (FUNCTIONALITY)

Summary: Chemical information cannot be validated precisely.

Detail (CDXML): This is because the format uses only a grammar-based validation. However, to validate chemical information more precisely sometimes requires defining more complex constraints between appropriate attributes or elements. Such constraints could be implemented in a custom validation tool using some programming language, but then one ends up reimplementing parts of the validation infrastructure already available with XML. This is why we think it is better to combine standard grammar and pattern-based XML schema languages (as described in the detail of ISSUE 10) to achieve precise validation of XML formats.

Detail (XDfile): It is because chemical information is stored using embedded chemical formats not based on XML and the format uses only a grammar-based validation. While it should be possible to validate some of the embedded formats (e.g. SMILES) relatively precisely without adding new parts to this format, it may be less simple for more structured embedded formats (e.g. CTfile formats). One option could be to validate the more structured data (especially in *field* elements) with some external validation tool specifically designed for the given embedded format. Another, we think better, option is to use the validation infrastructure already available in XML (see the detail of ISSUE 10). To implement the precise validation using a combined expressive power of standard grammar and pattern-based XML schema languages, chemical information should be ideally stored in appropriate attributes and elements for each small chunk of information, because then it is usually easier to formulate the validation constraints for the given attribute or element. However, in this format large chunks of information can be stored inside the *field* element. The large chunks can include for example the contents of a file that uses one of the CTfile formats (e.g. Molfile, Rxnfile etc.).

ISSUE 10 (CML)

Affects: Requirement 2 (FUNCTIONALITY)

Summary: The CMLLite validator service intended for flexible validation based on conventions[86] forces the user to switch away from standard XML validation tools.

Detail (CML): It is because the format schema validation capabilities were reduced to provide just a markup vocabulary with minimal restrictions, while the CMLLite validator will cover most of the validation process when necessary official conventions are implemented (see ISSUE 6 for details about the status of official conventions). However, the CMLLite validator is currently implemented using a custom unit test approach in XSLT and Java.[86] This means there is no schema for validation based on conventions that could be used with standard XML validation tools. Therefore, users of the format will have to rely on the online CMLLite validator service or they will need to implement their CMLLite-based validator using the available Java library.[86] The later will be probably necessary when validation will need to be performed offline or for many data files that have to be validated often. We think that by combining standard XML schema languages one could implement similar validation capabilities, achieve better compatibility and lower development overhead (especially with increasing number of conventions in the future). For example pattern-based XML schema languages like Schematron can provide interesting possibilities,[88,89,90] especially if combined with other grammar-based XML schema languages like RELAX NG[91,92] or XSD.[93,94] Then there is NVDL (Namespace-based Validation Dispatching Language), which enables the validation of XML documents with multiple namespaces by loading the appropriate schema for the content from each namespace.[95,96] NVDL can be also used to validate the content from one namespace by more than one schema,[96,97] which could be useful for combining the validation possibilities provided by grammar and pattern-based XML schema languages. Authors of this format however found Schematron too difficult to debug and to scale poorly with the complexity of validation rules.[86] We do not subscribe to this notion. In our new chemical XML format we implemented pattern-based validation using Schematron and combined it with grammar-based XML schema languages and NVDL.[98] Although the validation process contains complex rules to validate chemical information in our format and also other data in integrated XML formats, we found it adequately fast, easily extensible and compatible with the validation infrastructure already available in XML.

ISSUE 11 (CDX, CDXML, CTfile, XDfile)

Affects: Requirement 3 (FUNCTIONALITY); Requirement 4 (MODIFIABILITY)

Summary: Some widely used markup (e.g. XHTML) is not expected to be utilized inside the annotations in this format.

Detail (CDX, CDXML, CTfile, XDfile): For example the XHTML markup would make it possible to include hyperlinks and other useful formatting features[99] into the annotations. It would also ensure that all markup inside the annotations is compatible with web browsers without any additional transformations.

ISSUE 12 (CTfile)
Affects: Requirement 3 (FUNCTIONALITY)
Summary: Plain text annotation strings are limited in length and can be only placed at predefined locations (usually header comment lines) in format textual structure.

ISSUE 13 (CML)
Affects: Requirements 4 and 5 (MODIFIABILITY); Requirements 7 and 8 (USABILITY)
Summary: The raw XML structure of the format can be very variable, because even for the most recent version 3 official conventions still appear to be mostly unfinished and custom conventions are used instead.
Detail (CML): With custom conventions it is for example possible to declare that one *molecule* element uses "convention:Molfile" convention and put contents with Molfile V3000 syntax inside that element. But any user can just as easily decide to declare that different convention, which he or she developed and published on personal website, will be used for example in all *bond* elements inside some other *molecule* element. Even such far fetched scenario would be valid (according to schema version 3 or 2.4) as the user fulfilled all the requirements enforced by the format schema version 3 or 2.4 and also did as official documentation annotation for the *convention* attribute says: "...the author must ensure that the semantics are openly available and that there are mechanisms for implementation ...".[34] Consequently, such variability significantly complicates the implementation and reliable processing of the format. For example, it is more difficult to implement the automatized transformation of data from this format into other data formats. And as described in BENEFIT 12, generating of the web browser friendly preview of chemical data stored in the format is a closely related task that is equally affected by this. Additionally the format is quite difficult to understand and search especially for an average human user as its highly variable structure can be confusing to learn. Finally the unrestricted and variable structure of the format can cause incompatible implementations among different chemical software tools (see ISSUE 22).

ISSUE 14 (CDX, CTfile)
Affects: Requirements 5 (MODIFIABILITY)
Summary: It is not possible to use a single file to directly combine other scientific data with chemical information stored in this format (at least not in a similar way to what XML namespaces offer with regards to directly combining XML formats in complex XML documents, as described in the detail of ISSUE 15).

ISSUE 15 (CDXML)
Affects: Requirement 5 (MODIFIABILITY)
Summary: The format lacks a unique namespace, which complicates its usage when chemical information needs to be combined with other data.
Detail (CDXML): The unique namespace would enable the format to be directly combined with other XML data formats (see BENEFIT 13). It would also enable possible integration of some XML formats (e.g. XHTML, UnitsML, etc.) into this format.

ISSUE 16 (XDfile)
Affects: Requirement 5 (MODIFIABILITY)
Summary: The format namespace is improperly defined in a way that assumes users will adjust it before using the format.
Detail (XDfile): This could lead to multiple more or less similar, but potentially incompatible formats (depending on modifications added by the particular user). Except for some test case scenarios (e.g. practically comparing different variants of the format) we fail to see the purpose of such namespace usage. Instead, we think it is usually much more useful to define a unique namespace for the format so that it can be directly combined with other XML data formats (see the detail of ISSUE 15).

PeerJ PrePrints

ISSUE 17 (CDX, CTfile)

Affects: Requirement 6 (MODIFIABILITY)

Summary: The format is less extensible than a comparable XML format, which is not dependent on some legacy non-XML format specifications.

Detail (CDX): Although the format is precisely defined, and therefore, relatively extensible for a binary format, compared to well defined text formats (especially those based on XML) modifying and extending it is still much more difficult. This is explained quite nicely in The Art of Unix Programming book along with other differences between binary and text formats.[100]

Detail (CTfile): The format is relatively extensible, because it is a well defined text format, but XML has even more advantages with regards to extensibility (see BENEFIT 14).

ISSUE 18 (CDXML)

Affects: Requirement 6 (MODIFIABILITY)

Summary: The practical extensibility of the format is still dependent on changes in corresponding binary CDX specifications.[14]

Detail (CDXML): Without the dependency on CDX the future extensibility of this format could be relatively good, as it is quite precisely defined XML format.

ISSUE 19 (XDfile)

Affects: Requirement 6 (MODIFIABILITY)

Summary: The practical extensibility of the format chemical functionality is completely dependent on changes in embedded chemical formats, which are not based on XML.

ISSUE 20 (CDX)

Affects: Requirements 7 and 8 (USABILITY)

Summary: The raw binary structure of the format is very difficult to understand and search.

Detail (CDX): Even with the help of CDXHexDumper software tool it is quite cumbersome exercise compared to just looking at a text format, especially one that is based on XML. Thus, the format is quite difficult for human users to understand and search, and additionally, it is also more difficult to learn and implement with all its functionality.

ISSUE 21 (CTfile)

Affects: Requirement 7 (USABILITY)

Summary: The stored data items in this format often lack descriptions, so it is harder to understand and search the format without thoroughly looking in its documentation.

Detail (CTfile): In other words some mechanism with the benefits similar to those described in BENEFIT 15 for the XML markup could help here.

ISSUE 22 (CML)

Affects: Requirement 8 (USABILITY); Requirement 11 (AVAILABILITY)

Summary: Implementation of this format is sometimes incomplete or not compatible among different chemical software tools.

Detail (CML): For example in March 2014 we tested creating a simple molecule (1H-pyrrole) using JChemPaint 3.3-1210 and then failed to open the resulting file in Marvin Beans 6.2.2. The problem was caused by the JChemPaint conventions, which were not supported in Marvin Beans (and as of March 2015 the problem still persists in most recent JChemPaint 3.3-1210 and Marvin Beans 15.3.9.0). After we manually deleted elements *list* and *moleculeList* specifying conventions used by JChemPaint and left the child element *molecule* intact Marvin Beans loaded the file. When we submitted contents of the file created by JChemPaint 3.3-1210 to the online CMLLite validator service (http://validator.xml-cml.org) there were multiple problems. Attribute *id* on *list* element had invalid space in its value and convention used by JChemPaint was not recognized (thus generating multiple errors and a warning). Furthermore in May 2014 we noticed the online CMLLite validator service seems not to be available. Despite the used link to the CMLLite validator (http://validator.xml-cml.org) is still listed in the main menu of the format website, only the error message ”Your request on the specified host was not found.” is currently displayed (as of March 2015). One can easily see that such problems with compatibility negatively affect the usability and availability of the format, because users cannot assume that files in this format created by their

PeerJ PrePrints

PeerJ PrePrints

software will be usable with another software their colleague uses (even if both programs seem to support this format).

ISSUE 23 (CDX, CDXML)
Affects: Requirement 9 (USABILITY)
Summary: The available online documentation of the format is incomplete.
Detail (CDX): There is a number of dead links (e.g. links to the documentation of the *kCDXObj_Arrow* object or the properties *kCDXProp_3DCenter*, *kCDXProp_Arrow_EquilibriumRatio*, *kCDXProp_Arrowhead_Type*, *kCDXProp_Atom_ShowNonTerminalCarbonLabels*, *kCDXProp_Closed*, *kCDXProp_SupersededBy*, etc.).
Detail (CDXML): Some attributes (e.g. *ChemPropFormula*, *ChemPropName*, *Connectivity*, *RingRadius*, etc.) and elements (e.g. *annotation*, *bioshape*, *stoichiometrygrid*, etc.) are not documented and there is a number of dead links (e.g. links to the documentation of the *arrow* element or the attributes *ArrowEquilibriumRatio*, *ArrowType*, *Center3D*, *Closed*, *ShowNonTerminalCarbonLabels*, *SupersededBy*, etc.).

ISSUE 24 (CML)
Affects: Requirement 9 (USABILITY)
Summary: The detailed description of elements and attributes is only available from documentation annotations in the schema source code, instead of being available in a more user friendly manner (e.g. as an online documentation with cross-references and examples).
Detail (CML): Getting the complete documentation for an element or attribute from the schema source code is a task more suitable for a dedicated program or script. For example to get the complete documentation for attributes, all occurrences and references to the given attribute must be checked for documentation annotations, because different usages of the particular attribute are not always described in one documentation annotation associated with the attribute definition (one can verify this by checking the documentation annotations for attribute *count*, *format*, *id*, *name*, *order*, *role*, *type*, *x3*, etc. in format schema version 3 or 2.4). In format schema version 3 even if documentation is extracted programmatically one cannot easily find elements and attributes related to the given functionality (e.g. recording of reactions) by looking for the main element (e.g. *reaction*) and checking its parent and child elements with their attributes, since the schema enables all elements to contain each other. Additionally for some parts documentation annotations are unclear or vague, as can be seen from the additional file 1.

ISSUE 25 (XDfile)
Affects: Requirement 9 (USABILITY)
Summary: The documentation available at the format website during our analysis only described embedded chemical formats, but the chapter about this format was excluded.
Detail (XDfile): We had to use an outdated version,[44] which contained the relevant chapter about this format. However, some elements and attributes were not documented properly in there. For example the outdated version does not list all attributes enabled by current format schema for elements *Dataset*, *FieldDef* or *XDfile* and also the description for attribute *length* is missing.

ISSUE 26 (CDX, CDXML)
Affects: Requirement 11 (AVAILABILITY)
Summary: On Linux platforms proprietary ChemDraw software is not available[101] and found alternatives offer less functionality.

ISSUE 27 (XDfile)
Affects: Requirement 11 (AVAILABILITY)
Summary: For Mac or Linux platforms specialized chemical software appears not to be available.

ISSUE 28 (CTfile, XDfile)
Affects: Requirement 12 (AVAILABILITY)
Summary: The specifications of this format are available under proprietary license agreement after submitting your personal information using a registration form at the format website.

Concept analysis results

The identified concepts helped us to evaluate the strengths and weaknesses of CDXML, CML and XDfile. Additionally, we also used these concepts to select and decide how to implement the functionality for our new format. It is called UCM (Universal Chemical Markup), because we chose to base it on XML in the light of Benefits and issues analysis results, as explained in the Discussion of analysis results. All concepts with assigned CDXML, CML and XDfile attributes and elements are listed in the following sections. The description of each concept explains whether and how we decided to implement it in UCM. Presented UCM attributes and elements names are for the first version, which is described in our next article.[98] To design this version we obviously went through multiple iterations, where various possibilities for implementing the chosen concepts were tested (the resulting basic UCM tree structure is in additional file 3). The question mark symbol after an attribute or element name assigned to some concept indicates that the available documentation did not provide enough unambiguous information about given attribute or element. Therefore, we tried our best to decide the appropriate concept based on schema definition and the name of such attribute or element.

C-BOND

The concept that denotes the functionality required for the recording of chemical bonds. The exception that does not contain any chemical category concepts is XDfile, which inherits the chemical functionality of the embedded non-XML format and adds various metadata. Other formats have at least one element with some attributes for describing the bond order and participating chemical nodes. While the overall functionality is similar, formats differ in the details of the C-BOND concept implementation (e.g. different values for some bond orders, different attributes or elements for denoting participating chemical nodes, etc.), as can be seen in documentations for particular attributes and elements. In UCM we focused on minimizing the ambiguity and, thus, for example only letter codes are enabled inside the UCM *order* attribute,[98] as opposed to CML, which according to its schema allows both number and letter codes to express the bond order. Our other aim was to precisely describe even the delocalized or other bonds, in which multiple chemical nodes participate. To do this in UCM we utilized and extended the C-PARTICLE concept and added the UCM *join* element for recording bonding connections between multiple chemical nodes.[98]

Attributes (CDXML 4-8): *BeginAttach*, *Connectivity?*, *EndAttach*, *Order*, *Topology*
Attributes (CML 3): *order*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *order*
Elements (CDXML 4-8): *b*, *crossingbond*
Elements (CML 3): *bond*, *bondArray*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *bond*, *join*

C-IDENTIFIER

This concept represents the functionality for storing various chemical identifiers including registry numbers (e.g. Chemical Abstracts Service Registry Number), structure-based identifiers (e.g. InChI) or systematic chemical names (e.g. Preferred IUPAC Name). At first we used a dedicated UCM element for this functionality, but after including the *format* and *type* attributes (see the D-METADATA concept) the *structure* element could be reused instead. Details about how some UCM elements may be used in more than one context are available in our next article.[98]

Attributes (CDXML 4-8): *ChemPropFormula?*, *ChemPropName?*, *Formula*, *RegistryAuthority*, *RegistryNumber*
Attributes (CML 3): *concise*, *formula*, *inline*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): *regnum*
Elements (CML 3): *formula*, *identifier*, *name*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *structure*

C-ISOMERISM

The concept expressing the functionality for describing isomers. Both CDXML and CML can record at least the stereochemistry of a chirality centre or bond and CML in addition supports denoting tautomers. UCM 1-1-1 enables the recording of stereochemical configuration too and the support for denoting tautomers is planed for future UCM versions. Examples in our next article illustrate how UCM 1-1-1 can express the stereochemistry of a chirality centre or bond, as well as the twist conformation of a bidentate ligand or even the absolute configuration of three bidentate ligands.[98] In addition UCM 1-1-1 supports recording the stereochemistry of a square planar complex, an octahedral complex or a chiral axis.[98]

Attributes (CDXML 4-8): *AS*, *BS*, *BondCircularOrdering*, *BondOrdering*, *DihedralIsChiral*, *EnhancedStereoGroupNum*, *EnhancedStereoType*, *HDash*, *HDot*
Attributes (CML 3): *chirality*, *conventionValue*, *tautomeric*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *sense*
Elements (CDXML 4-8): NONE
Elements (CML 3): *atomParity*, *bondStereo*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *stereo*

C-NODE

The concept that represents the functionality for recording chemical nodes (i.e. the nodes of chemical graph). It usually represents a monoatomic particle (e.g. an atom or a monoatomic ion), but in the case of CDXML it can even be a molecular fragment. Chemical nodes in analyzed formats express the monoatomic particles or molecular fragments using a predefined set of enabled text symbols (usually denoting the given chemical element). UCM differs form these formats, as the C-PARTICLE concept functionality implemented in UCM is utilized for creating the reusable definitions of chemical nodes composed from protons, neutrons and electrons.[98] By enforcing the precise definitions of chemical nodes UCM can provide advanced built-in validation capabilities[98] and also does not need attributes or elements dedicated to storing information about valence, isotopes, chemical element names, and so on.

Attributes (CDXML 4-8): *AbnormalValence*, *AtomNumber*, *Attachments*, *Charge*, *Element*, *ElementList*, *FreeSites*, *GenericList*, *GenericNickname*, *Geometry*, *ImplicitHydrogens*, *Isotope*, *IsotopicAbundance*, *NodeType*, *NumHydrogens*, *Radical*, *RingBondCount*, *SubstituentsExactly*, *SubstituentsUpTo*, *Translation*, *UnsaturatedBonds*
Attributes (CML 3): *elementType*, *hydrogenCount*, *isotopeNumber*, *occupancy*, *spin*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *charge*
Elements (CDXML 4-8): *n*
Elements (CML 3): *abundance*, *atom*, *atomArray*, *isotope*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *node*

C-PARTICLE

This concept groups together functionality related to the description of subatomic particles. Among the analyzed formats it is partially implemented in CML, which can record electrons. In UCM the C-PARTICLE concept plays an important role of complementing the C-BOND and C-NODE concepts, because it enables the precise recording of information about protons, neutrons or electrons.[98] Such information are utilized in UCM for the precise description and validation of chemical bonds and nodes.[98]

Attributes (CDXML 4-8): NONE
Attributes (CML 3): *spin*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): NONE
Elements (CML 3): *electron*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *particle*, *share*

PeerJ PrePrints

C-REACTION

The concept denoting the functionality required for the recording of chemical reactions. Both CDXML and CML offer also functionality for storing related information like reaction steps. UCM 1-1-1 does not support reactions, but in future UCM versions we plan add the necessary attributes and elements, as well as validation rules to enable the support for chemical reactions together with various related data.

Attributes (CDXML 4-8): *ComponentIsHeader?*, *ComponentIsReactant?*, *ReactionStepAtomMap*, *ReactionStepAtomMapAuto*, *ReactionStepAtomMapManual*, *ReactionStepProducts*, *ReactionStepReactants*, *RxnChange*, *RxnParticipation*, *RxnStereo*
Attributes (CML 3): *scheme*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): *scheme*, *sgcomponent?*, *step*, *stoichiometrygrid?*
Elements (CML 3): *mechanism*, *mechanismComponent*, *product*, *reactant*, *reaction*, *reactionScheme*, *reactionStep*, *reactiveCentre*, *spectator*, *transitionState*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

C-STRUCTURE

The concept that represents the functionality related to describing various chemical structures. CDXML and CML obviously offer more functionality for chemical structures (e.g. support for recording polymers, crystallographic information, etc.) when compared to UCM 1-1-1. But UCM 1-1-1 can provide more precise description of bonding in chemical structures together with much better validation of these structures.[98] In future UCM versions we plan to implement the missing functionality with focus on precise validation.

Attributes (CDXML 4-8): *Absolute*, *AminoAcidTermini?*, *BioShapeType?*, *BracketUsage*, *ComponentOrder*, *ConnectionOrder*, *CylinderDistance?*, *CylinderHeight?*, *CylinderWidth?*, *Dipole?*, *DNAWaveHeight?*, *DNAWaveLength?*, *DNAWaveOffset?*, *DNAWaveWidth?*, *EnzymeHeight?*, *EnzymeReceptorSize?*, *EnzymeWidth?*, *GolgiHeight?*, *GolgiLength?*, *GolgiWidth?*, *GproteinLowerHeight?*, *GproteinUpperHeight?*, *HelixProteinExtra?*, *ImmunoglobinHeight?*, *ImmunoglobinWidth?*, *LinkCountHigh*, *LinkCountLow*, *MarkerAngle?*, *MarkerOffset?*, *MembraneElementSize?*, *MembraneEndAngle?*, *MembraneMajorAxisSize?*, *MembraneMinorAxisSize?*, *MembraneStartAngle?*, *NeckHeight?*, *NeckWidth?*, *NumberBasePairs?*, *PipeWidth?*, *PolymerFlipType*, *PolymerRepeatPattern*, *Racemic*, *RegionEnd?*, *RegionOffset?*, *RegionStart?*, *Relative*, *RingRadius?*, *RLogicGroup?*, *RLogicIfThenGroup?*, *RLogicOccurrence?*, *RLogicRestH?*, *SequenceType?*, *SRULabel*, *Valence*
Attributes (CML 3): *formalCharge*, *hydrogenCount*, *irreducibleRepresentation*, *kpoint*, *latticeType*, *periodic*, *pointGroup*, *pointGroupMultiplicity*, *spaceGroup*, *spaceGroupMultiplicity*, *spaceType*, *spinMultiplicity*, *symmetryOriented*, *weight*, *z*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *charge*
Elements (CDXML 4-8): *altgroup*, *bioshape?*, *bracketattachment*, *bracketedgroup*, *fragment*, *plasmidmap?*, *plasmidmarker?*, *plasmidregion?*, *rlogic?*, *rlogicitem?*
Elements (CML 3): *band*, *cellParameter*, *crystal*, *fragment*, *join*, *kpoint*, *lattice*, *latticeVector*, *molecule*, *region*, *sample*, *substance*, *symmetry*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *structure*

D-METADATA

The concept for the functionality related to metadata. Compared to analyzed formats, which have dedicated attributes and elements for various metadata, UCM focuses only on recording metadata necessary for the purposes of the format itself. It is because we believe XML comments and UCM annotations provide sufficient possibilities for including any metadata that are otherwise unnecessary for the correct functioning of UCM. It may seem inadequate, but since UCM annotations support XHTML markup[98] one can easily utilize the XHTML *meta* element to record various metadata in a robust way. Thus, UCM

PeerJ PrePrints

1-1-1 has only three dedicated metadata attributes (*format*, *type* and *version*) to store the format and type of some UCM elements and the used version of UCM respectively.[98]

Attributes (CDXML 4-8): *CartridgeData*, *ChainAngle*, *ChemicalPropertyIsActive*, *charset*, *CreationDate*, *CreationProgram*, *CreationUserName*, *Edition*, *EditionAlias*, *Footer*, *Header*, *IgnoreWarnings*, *Integral*, *InterpretChemically*, *ModificationDate*, *ModificationProgram*, *ModificationUserName*, *name*, *Name*, *NeedsClean?*, *Persistent*, *TagType*, *Warning*
Attributes (CML 3): *constraint*, *content*, *convention*, *dataType*, *delimiter*, *dictionaryPrefix*, *end*, *fileId*, *format*, *fractionDigits*, *ft*, *IgnoreUnconnectedAtoms*, *inherit*, *IsEdited?*, *IsHidden?*, *IsReadOnly?*, *label*, *name*, *namespace*, *objectClass*, *orientation*, *pattern*, *process*, *start*, *symbol*, *totalDigits*, *type*, *version*
Attributes (XDfile 20110328): *dateOrder*, *decimalSeparator*, *encoding*, *isIndexed*, *isKey*, *isPrimaryKey*, *javaFormat*, *length*, *maxLength*, *molFormat*, *molVersion*, *nativeName*, *nullsAllowed*, *precision*, *rxnFormat*, *scale*, *timeFormat*, *timeOrder*, *VERSION*
Attributes (UCM 1-1-1): *format*, *type*, *version*
Elements (CDXML 4-8): NONE
Elements (CML 3): *label*, *metadata*, *module*, *object*, *parameter*, *particle*
Elements (XDfile 20110328): *CreateDate*, *CreateTime*, *CreatorName*, *DataSource*, *FieldDef*, *Metadata*, *ParentDef*, *ProgramSource*, *Source*
Elements (UCM 1-1-1): NONE

D-PROPERTY DATA

The concept that groups together the functionality necessary to store a variety of data related to measured or calculated properties. Here we also list the CDXML attributes and elements for including the embedded binary objects, because these objects could be regarded as additional properties with data related to the chemical content in a CDXML file. As explained in ISSUE 1, it is our opinion that such binary objects should be included in external files or replaced by a form that can be better validated and manipulated by XML tools. Anyway, while CDXML and CML have many attributes and elements to store various properties with their related data, XDfile uses a far lower number of attributes and elements more universally. Although CDXML and CML may express some additional data more precisely with their dedicated attributes and elements, we chose to implement this concept in UCM using just a few universal attributes and elements. Compared to XDfile we utilized UnitsML for expressing scientific quantities in a robust way. UnitsML makes it possible to precisely define various scientific quantities with their names, associated units and other data[84,98] (see the D-PROPERTY UNITS concept). UCM 1-1-1 supports defining both property conditions (e.g. standard temperature and pressure) and errors (e.g. standard deviation of the mean).[98] In future UCM versions we plan to add support for graph properties showing the relationship between two or three properties and for properties with predefined textual values.

Attributes (CDXML 4-8): *attribute*, *BasisObjects*, *BMP*, *ChemicalPropertyType*, *ChemPropAnalysis?*, *ChemPropBoilingPt?*, *ChemPropCLogP?*, *ChemPropCMR?*, *ChemPropCritPres?*, *ChemPropCritTemp?*, *ChemPropCritVol?*, *ChemPropEForm?*, *ChemPropExactMass?*, *ChemPropGibbs?*, *ChemPropHenry?*, *ChemPropLogP?*, *ChemPropMOverZ?*, *ChemPropMeltingPt?*, *ChemPropMolWt?*, *ChemPropMR?*, *ChemProptPSA?*, *Class*, *CompressedEnhancedMetafile?*, *CompressedOLEObject?*, *CompressedWindowsMetafile?*, *ConstraintMax*, *ConstraintMin*, *ConstraintType*, *EnhancedMetafile*, *GIF*, *JPEG*, *MacPICT*, *OLEObject*, *OrbitalType*, *OriginFraction*, *PDF*, *PNG*, *PointIsDirected*, *Rf*, *SGDataType?*, *SGDataValue?*, *SGPropertyType?*, *SolventFrontFraction*, *Tail*, *TIFF*, *TopLeft*, *TopRight*, *UncompressedEnhancedMetafileSize?*, *UncompressedOLEObjectSize?*, *UncompressedWindowsMetafileSize?*, *Value*, *Weight*, *WindowsMetafile*, *XAxisLabel*, *XLow*, *XSpacing*, *YAxisLabel*, *YLow*, *YScale*
Attributes (CML 3): *columns*, *constantToData*, *error*, *errorBasis*, *errorValue*, *errorValueArray*, *integral*, *l*, *length*, *lm*, *m*, *matrixType*, *max*, *maxExclusive*, *maxInclusive*, *maxLength*, *maxValueArray*, *measurement*, *min*, *minExclusive*, *minInclusive*, *minLength*, *minValueArray*, *multiplierToData*, *n*, *peakHeight*, *peakMultiplicity*, *peakShape*, *rows*, *shape*, *size*, *state*, *tableType*, *term*, *value*, *xMax*, *xMin*, *xValue*, *xWidth*, *yield*, *yMax*, *yMin*, *yValue*, *yWidth*
Attributes (XDfile 20110328): *name*, *type*
Attributes (UCM 1-1-1): *quantity*

Elements (CDXML 4-8): *chemicalproperty*, *constraint*, *embeddedobject*, *objecttag*, *represent*, *sgdatum?*, *spectrum*, *tlclane*, *tlcplate*, *tlcspot*
Elements (CML 3): *amount*, *angle*, *array*, *atomicBasisFunction*, *atomType*, *basisSet*, *bondType*, *definition*, *dictionary*, *eigen*, *entry*, *gradient*, *length*, *matrix*, *peak*, *peakGroup*, *peakStructure*, *potential*, *potentialForm*, *property*, *torsion*, *scalar*, *spectrum*, *spectrumData*, *table*, *tableCell*, *tableContent*, *tableHeader*, *tableHeaderCell*, *tableRow*, *torsion*, *xaxis*, *yaxis*
Elements (XDfile 20110328): *Data*, *Field*, *Parent*, *Record*
Elements (UCM 1-1-1): *property*, *values*

D-PROPERTY UNITS

This concept denotes the functionality for recording the scientific units that need to be associated with property data. With the exception of CML, which has its mechanism of dictionaries and necessary dedicated attributes and elements for defining and working with scientific units, the remaining formats do not properly enforce the recording of scientific units. CDXML has only two dedicated attributes (*XType* and *YType*) specifying units for axes used by CDXML *spectrum* element. XDfile with its optional *units* attribute, which can have any string value (according to XDfile schema), only provides a basic possibility of mentioning the units without their precise definition. Because UCM integrates UnitsML, there are no UCM attributes or elements needed for implementing this concept. Instead UnitsML markup inside an UCM *define* element, may be utilized to define various scientific units in terms of predefined basic units.[98] UnitsML provides predefined SI (International System of Units) base and SI derived units as well as widely used non-SI units.[84] In our next article we discus practical UCM examples showing how UnitsML can be easily utilized.[98]

Attributes (CDXML 4-8): *XType*, *YType*
Attributes (CML 3): *abbreviation*, *constantToSI*, *dimensionBasis*, *isSI*, *multiplierToSI*, *peakUnits*, *power*, *preserve*, *recommendedUnits*, *siNamespace*, *siNamespaceArray*, *units*, *xUnits*, *yUnits*
Attributes (XDfile 20110328): *units*
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): NONE
Elements (CML 3): *dimension*, *unit*, *unitType*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

G-AMOUNT

The concept that represents the amount (e.g. count, fraction, etc.) for some objects. CDXML implements the *RepeatCount* attribute to store how many times a *bracketedgroup* element is repeated. This enables for example the recording of polymers in CDXML.[15] Both CML and XDfile offer attributes such as *count* or *totalRecords* that can be used on more than one element, as can be seen in the documentations for the given attributes. In UCM 1-1-1 the *counts* attribute is enabled just on *particle* elements and the *fractions* attribute is utilized on *particle* and *share* elements.[98] However, we specifically designed these attributes as generally as possible to ensure they are usable on any elements that may need them in future UCM versions.

Attributes (CDXML 4-8): *RepeatCount*
Attributes (CML 3): *count*, *countExpression*, *number*, *ratio*
Attributes (XDfile 20110328): *totalRecords*
Attributes (UCM 1-1-1): *counts*, *fractions*
Elements (CDXML 4-8): NONE
Elements (CML 3): NONE
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

G-ANNOTATION

The concept denoting the annotation functionality. CDXML and XDfile have relatively few attributes and elements dedicated to annotations. CML, on the other hand, could easily implement this concept using lower number of attributes and elements. Even if some special kinds of annotations are required we think an attribute on one dedicated annotation element could denote the specific purpose of that

PeerJ PrePrints

element. In UCM *description* elements we also implemented the support for BibTeXML (BibTeX Markup Language) literature references and XHTML markup,[98] because we found such functionality is missing in analyzed formats, and because we think a scientific data format should offer the possibility of adding proper bibliography references to annotations.

Attributes (CDXML 4-8): *Comment*, *Content?*, *Keyword?*
Attributes (CML 3): *duration*, *endCondition*, *role*, *startCondition*, *title*
Attributes (XDfile 20110328): *name*
Attributes (UCM 1-1-1): *litrefs*
Elements (CDXML 4-8): *annotation?*, *t*
Elements (CML 3): *action*, *description*, *documentation*, *observation*
Elements (XDfile 20110328): *Description*
Elements (UCM 1-1-1): *description*

G-CONTAINER

The concept of a general container element used either as root element or to group together various data in the format XML structure. Of course each XML element that can contain some child elements (according to the schema of the given XML format) could be regarded as container element, but we were interested here just in elements existing mainly to serve as container elements. While CDXML, XDfile and UCM have relatively few dedicated container elements, in the case of CML this concept could be easily implemented using lower number of elements.

Attributes (CDXML 4-8): NONE
Attributes (CML 3): NONE
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): *CDXML*, *group*, *sequence*
Elements (CML 3): *actionList*, *arrayList*, *atomTypeList*, *bandList*, *bondTypeList*, *cml*, *conditionList*, *fragmentList*, *isotopeList*, *kpointList*, *list*, *metadataList*, *moleculeList*, *parameterList*, *peakList*, *potentialList*, *productList*, *propertyList*, *reactantList*, *reactionList*, *reactionStepList*, *spectatorList*, *spectrumList*, *stmml*, *substanceList*, *system*, *tableRowList*, *unitList*, *unitTypeList*
Elements (XDfile 20110328): *Dataset*, *XDfile*
Elements (UCM 1-1-1): *define*, *ucm*

G-COORDINATES

The concept of coordinates in 2-dimensional or 3-dimensional space. It enables storing the positions of chemical nodes or it can be used for non-chemical objects like points. In the case of CML this concept could be easily implemented using lower number of attributes. On the other hand, CDXML uses just two attributes (*p* for 2-dimensional and *xyz* for 3-dimensional space coordinates) and XDfile again utilizes the functionality of the embedded non-XML format. UCM has separate attributes (*x*,*y* and *z*) for coordinates in 3-dimensional space,[98] as the *z* attribute can be always set to zero value in the case of 2-dimensional space.

Attributes (CDXML 4-8): *p*, *xyz*
Attributes (CML 3): *dimensionality*, *periodicity*, *x2*, *x3*, *xFract*, *y2*, *y3*, *yFract*, *z3*, *zFract*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *x*, *y*, *z*
Elements (CDXML 4-8): NONE
Elements (CML 3): *zMatrix*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

G-GEOMETRY

This concept groups together the geometric functionality related to representing various graphical shapes. The G-GEOMETRY concept, as opposed to the G-VISUALIZATION concept, contains mainly attributes and elements denoting graphical shapes that can be utilized for describing chemically relevant information (e.g. the unit cell of a crystal). With the exception of XDfile there is quite wide palette of such attributes

PeerJ PrePrints

and elements in both CDXML and CML. UCM 1-1-1 has just the *point* element for the recording of chemical structures with important places, which are outside the scope of UCM *node* elements.[98] However, in future UCM versions we can add other attributes or elements when the need to extend the geometric functionality arises.

Attributes (CDXML 4-8): *AngularSize*, *ArrowEquilibriumRatio*, *ArrowheadCenterSize*, *ArrowheadHead*, *ArrowheadTail*, *ArrowheadType*, *ArrowheadWidth*, *ArrowShaftSpacing*, *ArrowType*, *BottomLeft*, *BottomRight*, *BoundingBox*, *BracketType*, *Center3D*, *Closed*, *CornerRadius*, *CurvePoints*, *CurvePoints3D*, *CurveSpacing*, *CurveType*, *GeometricFeature*, *GraphicType*, *Head3D*, *HeadCenterSize*, *HeadSize*, *HeadWidth*, *LineType*, *LineWidth*, *MajorAxisEnd3D?*, *MinorAxisEnd3D?*, *NoGo*, *OvalType*, *RectangleType*, *RelationValue*, *Side*, *SymbolType*, *Tail3D*
Attributes (CML 3): *box3*, *point3*, *sphere3*, *vector3*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): *arrow*, *border*, *curve*, *geometry*, *graphic*
Elements (CML 3): *line3*, *plane3*, *point3*, *sphere3*, *transform3*, *vector3*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): *point*

G-IDENTIFIER

The concept of a general identifier. Such a unique identifier is usually used to refer to element holding specific data in the given XML document. While XDfile does not implement the functionality of G-IDENTIFIER and G-REFERENCE concepts, in CDXML and CML G-IDENTIFIER concept could be easily implemented with lower number of attributes. For UCM we found that one *id* attribute is sufficient. Another important aspect is ensuring that the identifier is unique. XML provides the ID type or xml:id mechanism to control the identifier uniqueness.[11,12,102] However, the main schemas of CDXML and CML do not utilize this, as for example the *id* attribute in both formats is not defined using the ID type nor does it use the xml:id mechanism. It is of course possible to check the uniqueness of the identifier through other approaches (e.g. by dedicated Schematron validation patterns or by some custom-built validation routines in external software). But unless some very important design requirement prevents the usage of the ID type or xml:id mechanism, we would suggest it is better to go with these standard ways provided by XML technology. The UCM *id* attribute is defined to be of type ID,[98] which enabled us to utilize the IDREFS type for UCM *idrefs* attribute (see the G-REFERENCE concept for details).

Attributes (CDXML 4-8): *CrossReferenceIdentifier*, *id*, *SequenceIdentifier*
Attributes (CML 3): *atomID*, *bondID*, *id*, *idgen*, *serial*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *id*
Elements (CDXML 4-8): NONE
Elements (CML 3): NONE
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

G-REFERENCE

The concept that represents the functionality necessary to refer to a general identifier. Again in the case of CDXML and CML this concept could be implemented using lower number of attributes. UCM utilizes one *idrefs* attribute to refer to all UCM elements.[98] It also has the *litrefs* and *quantity* attributes for referencing the elements form BibTeXML and UnitsML respectively.[98] Similarly to the ID type (mentioned in the G-IDENTIFIER concept) XML offers IDREF and IDREFS attribute types, which ensure that referenced identifiers exist in the given XML document.[11,12] While the analyzed formats do not use this, the UCM *idrefs* attribute benefits from such automatically gained validation functionality, as we defined it using the IDREFS type.[98] The *litrefs* and *quantity* attributes, on the other hand, rely on the specific UCM Schematron validation rules.[98]

Attributes (CDXML 4-8): *AltGroupID*, *B*, *BondID*, *BracketedObjectIDs*, *ChemicalPropertyDisplayID*, *ComponentReferenceID?*, *CrossReferenceContainer*, *CrossReferenceDocument*, *CrossReferenceSequence*, *E*, *font*, *GraphicID*, *InnerAtomID*, *object*, *SupersededBy*

PeerJ PrePrints

Attributes (CML 3): *atomMap*, *atomRef*, *atomRef1*, *atomRef2*, *atomRefArray*, *atomRefGroup?*, *atomRefs*, *atomRefs2*, *atomRefs3*, *atomRefs4*, *atomSetRef*, *bondMap*, *bondRef*, *bondRefs*, *dictRef*, *electronMap*, *form*, *from*, *fromContext*, *fromSet*, *fromType*, *href*, *isotopeListRef*, *isotopeRef*, *kpointRef*, *linkType*, *moleculeRef*, *moleculeRefs*, *moleculeRefs2*, *parentSI*, *ref*, *regionRefs*, *to*, *toContext*, *toSet*, *toType*, *unitType*
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): *idrefs*, *litrefs*, *quantity*
Elements (CDXML 4-8): *crossreference*
Elements (CML 3): *atomSet*, *bondSet*, *link*, *map*
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

G-VISUALIZATION

The concept that is unique for CDXML and denotes the functionality for describing precisely how to visualize stored chemical information. We currently do not plan to implement this concept in UCM for at least two reasons. The first is that if the aim is to preserve the precise visualization, we believe it is often better to utilize the widely compatible external formats for images, animations and various multimedia in general. Our second reason is already described in ISSUE 1 – we think the visualization should be separated from the content in this case.

Attributes (CDXML 4-8): *alpha?*, *b*, *bgalpha?*, *bgcolor*, *BoldWidth*, *BondLength*, *BondSpacing*, *BondSpacingAbs*, *BoundsInParent*, *CaptionColor*, *CaptionFace*, *CaptionFont*, *CaptionJustification*, *CaptionLineHeight*, *CaptionSize*, *color*, *CrossingBonds*, *CrossingBondss?*, *Display*, *Display2*, *DisplayName?*, *DoublePosition*, *DrawingSpace*, *extent*, *ExternalConnectionType*, *face*, *FadePercent?*, *FixInPlaceExtent*, *FixInPlaceGap*, *FooterPosition*, *FractionalWidths*, *FrameType*, *g*, *GroupFrame*, *HashSpacing*, *HeaderPosition*, *Height*, *HeightPages*, *HideImplicitHydrogens*, *Justification*, *LabelAlignment*, *LabelColor*, *LabelDisplay*, *LabelFace*, *LabelFont*, *LabelJustification*, *LabelLineHeight*, *LabelSize*, *LineHeight*, *LineStarts*, *LipSize*, *MacPrintInfo*, *Magnification*, *MarginWidth*, *NumColumns*, *NumRows*, *PageDefinition*, *PageOverlap*, *PaneHeight*, *PositioningAngle*, *PositioningOffset*, *PositioningType*, *PrintMargins*, *PrintTrimMarks*, *r*, *ReactionStepArrows*, *ReactionStepObjectsAboveArrow*, *ReactionStepObjectsBelowArrow*, *ReactionStepPlusses*, *RotationAngle*, *ShadowSize*, *ShowAtomEnhancedStereo*, *ShowAtomNumber*, *ShowAtomQuery*, *ShowAtomStereo*, *ShowBondQuery*, *ShowBondRxn*, *ShowBondStereo*, *ShowBorders*, *ShowNonTerminalCarbonLabels*, *ShowOrigin*, *ShowRf*, *ShowSequenceBonds?*, *ShowSequenceTermini?*, *ShowSideTicks*, *ShowSolventFront*, *ShowTerminalCarbonLabels*, *size*, *SplitterPositions*, *TextFrame*, *Tracking*, *Transparent?*, *Visible*, *Width*, *WidthPages*, *WinPrintInfo*, *WindowIsZoomed*, *WindowPosition*, *WindowSize*, *WordWrapWidth*, *Z*
Attributes (CML 3): NONE
Attributes (XDfile 20110328): NONE
Attributes (UCM 1-1-1): NONE
Elements (CDXML 4-8): *color*, *colortable*, *font*, *fonttable*, *page*, *s*, *splitter*, *table*, *templategrid*
Elements (CML 3): NONE
Elements (XDfile 20110328): NONE
Elements (UCM 1-1-1): NONE

DISCUSSION OF ANALYSIS RESULTS

While in the first stage we basically accumulated information about analyzed formats, during the second stage we obtained very useful detailed data for designing our new chemical format. Information from the first stage may provide an overview of currently established formats for common chemical data. However, more important for us was the second stage and its two main outputs (i.e. the benefits and issues together with found concepts).

The set of benefits and issues, identified in general-purpose chemical formats from the second stage, makes it possible to compare both XML and non-XML formats to see if XML offers some benefits for chemical formats. Such a comparison enabled us to distinguish the benefits that clearly stem from the utilization of XML technology. These include the basic validation functionality and improvements in the modifiability and usability requirement categories. The basic validation (BENEFIT 8) is offered automatically by XML, because a schema defining the structure of some XML format, can be used to verify whether data conform to such a format. Another important benefit of XML is the standard XML tool chain that enables more effective processing and implementation of the given XML format resulting in the modifiability and usability improvements (BENEFITS 11 and 12). Also, one should not overlook the modifiability bonuses added by the XML namespaces (BENEFIT 13). Using namespaces it is possible to combine various XML formats in a single XML document,[103,104] which can combine various scientific data together or can help with their visualization (e.g. SVG and CML combination[105]). Additional modifiability and usability improvements arise from better extensibility (BENEFIT 14) and readability (BENEFIT 15) provided by the usage of XML technology. With this in mind, it may seem chemical XML formats should automatically fulfill our Requirements for modern chemical format better than non-XML formats. But, on closer look it is apparent some of these benefits were not found in all analyzed XML formats.

Discovered benefits and issues suggest the advantages of XML technology do not automatically translate into evident improvements, because various design choices can negate them partially or completely. One example of such a design choice is the dependency on legacy non-XML format specifications. With the exception of CML, all XML formats we analyzed depend on some legacy non-XML format specifications. Although this can lead to potential problems (ISSUES 1, 7, 18, 19 and some weaknesses of NCBI XML and PDBML described in additional file 2) it is a trade-off arising from specific situations and required use case scenarios (e.g. a format is developed for data structures used in existing software to limit changes in current infrastructure). Another example is designing a format with maximum flexibility. In general good flexibility is worth some trade-offs and it is especially important for a general-purpose chemical format expected to be useful in various domains of chemistry. However, as can be seen from ISSUE 5 (and ISSUES 6, 10, 13 and 22 related to it), aiming for very high flexibility can negatively affect other software qualities of the given XML format. Thus, based on the information obtained so far, we believe that with the benefits of XML in mind from scratch it should be possible to create a format, in which the advantages of XML are utilized to a greater extent. In order to test this we decided to design UCM as a new chemical XML format.

To avoid the potential issues of analyzed formats where possible, we used the set of concepts, found during the second stage in CDXML, CML and XDfile formats, to develop concepts for UCM and test their implementation. For the first version of UCM our aim was to design extensible core functionality, hence we tried to express the selected UCM concepts with as few attributes or elements as possible. Therefore, UCM concepts were often formulated using considerably less attributes or elements than we found in some analyzed formats (see concepts such as C-IDENTIFIER, C-NODE, D-PROPERTY DATA, G-ANNOTATION, G-CONTAINER, G-COORDINATES, G-IDENTIFIER and G-REFERENCE). This way we avoided unnecessary parts that would only add ambiguity. It also helped us to keep UCM concise, which made it easier to develop the precise validation, as described in our next article.[98] Although not all concepts we plan to eventually include in UCM made it to the first version, the basic UCM structure in additional file 3 is specifically designed to be easily extensible to add these later (especially the C-REACTION concept and additional functionality for C-STRUCTURE concept).

The concepts also clearly demonstrate that CDXML, CML and XDfile approach the problem of storing chemical data differently. CDXML stores both chemical and content visualization data together (see G-GEOMETRY and G-VISUALIZATION concepts). CML, on the other hand, separates the content form presentation and focuses on storing chemical information with great flexibility (see BENEFIT 7), while XDfile just inherits the chemical functionality of the embedded non-XML format and adds metadata. Using XDfile the embedded chemical formats not based on XML can be combined in a single file, but the XML structure of XDfile does not contain any chemical category concepts.

For the analysis we devised a repeatable procedure and attempted to limit the influence of our subjective viewpoints. The analysis procedure consisted of extensible steps utilizing Python modules and custom XML files for effective processing of gathered data. Using the idea of software quality attributes,[9] we composed the set of Requirements for modern chemical format that served as the objective criteria for our assessments. But at the same time it is clear that any similar analysis will always have partially subjective nature. This is why all the assessments and concepts presented here are open to further investigation and should not be taken as definitive. In fact it is not even their purpose. Instead, our categorized assessments and concepts should ensure the readers can decide, as we did when designing

PeerJ PrePrints

UCM, whether the particular assessment or concept is relevant for their use case. Furthermore, all software tools we created for the analysis are available with complete source code and documentation under an open source license at http://www.universalchemicalmarkup.org. Already described interactive XHTML references and our custom XML data file templates are also obtainable from the website. Thus, the interested reader can reuse and adapt our tools and methods to analyze or design new formats with other software quality requirements.

CONCLUSIONS

We analyzed current formats for common chemical data and identified their strengths and weaknesses to design concepts for a new general-purpose chemical format. Information gathered in the first stage of the analysis may serve as an overview of currently established formats for chemical structures and in some instances for reactions, properties and other data too. Using this information we also selected the formats with the most general-purpose chemical functionality for the second stage. The analysis of selected formats in the second stage revealed detailed benefits and issues in these formats as well as useful concepts.

The analysis results confirmed the potentially significant benefits stemming from the usage of XML technology for a chemical format. However, our analysis also revealed that the developers generally need to design the format carefully to obtain the final result, where the benefits of XML are not erased by other design choices. Thus, keeping this in mind, we decided to use XML as a basis for UCM.

Our UCM concepts were designed specifically to utilize the XML benefits and to avoid the potential issues of analyzed formats where possible. It meant not only selecting the concepts from those found during the analysis, but also choosing how the concepts will be implemented. Using an iterative approach we came up with the specific concepts and XML structure for UCM 1-1-1, which we think provides very promising and extensible core functionality, as is further described in our next article.[98]

In addition, we believe the analysis procedure and its results could be reused in future research. Strengths, weaknesses and concepts identified in analyzed formats may be used to improve existing chemical formats or to design new ones. More importantly, our experience from this analysis suggests the procedure we designed and utilized here is adaptable for different domains both inside and outside the chemistry area. Selected formats and software quality requirements for a format in the given domain will of course differ, but the methodology can remain similar. Therefore, the software tools from our DATA FORMATS ANALYZER package may be reused to compare analyzed formats and to identify useful concepts in them.

AVAILABILITY AND REQUIREMENTS

Project name: DATA FORMATS ANALYZER 1-1-1
Project home page: http://www.universalchemicalmarkup.org/#DFA--1-1-1
Operating system(s): platform independent
Programming language: Python
Other requirements: Python 2.7 with lxml and gdata modules
License: GNU GPL 3
Any restrictions to use by non-academics: None

SUPPLEMENTAL INFORMATION

Additional file 1 – Interactive references

Detailed information about interactive references generated in the first stage of the analysis.

Additional file 2 – Formats excluded from second stage

Information obtained in the first stage of the analysis for the formats excluded form the second stage.

Additional file 3 – UCM tree structure

The basic UCM tree structure developed on the basis of our concept analysis.

PeerJ PrePrints

ACKNOWLEDGMENTS

The authors thank Professor David Sedmidubský for helpful comments and critical review of the manuscript.

COMPETING INTERESTS

The authors declare that they have no competing interests.

AUTHOR CONTRIBUTIONS

Jan Mokrý wrote the manuscript, introduced the idea of requirements for chemical format based on software quality atributes, performed the analysis of chemical formats, developed the concepts and tree structrure for UCM, developed all additional software tools and prepared the website (http://www.universalchemicalmarkup.org).

Miloslav Nič reviewed drafts of the manuscript, provided supervision and advice, especially with the concepts and tree structrure for UCM.

FUNDING

There was no external funding for this project. All financial resources needed to carry out this research were contributed by one of the authors, Jan Mokrý.

REFERENCES

1 RZEPA, H. S.; MURRAY-RUST, P.; WHITAKER, B. J. The Application of Chemical Multipurpose Internet Mail Extensions (Chemical MIME) Internet Standards to Electronic Mail and World Wide Web Information Exchange. *J. Chem. Inf. Comput. Sci.* 1998, vol. 38, pp. 976–982. Available also at: ⟨http://dx.doi.org/10.1021/CI9803233⟩.

2 RZEPA, H. S. *Chemical MIME Home page*. 2004. Available at: ⟨http://www.ch.ic.ac.uk/chemime/⟩.

3 *Supported File Formats and Options*. Open Babel. Available at: ⟨http://openbabel.org/docs/current/FileFormats/Overview.html⟩.

4 *XML Technology*. The World Wide Web Consortium (W3C). Available at: ⟨http://www.w3.org/standards/xml⟩.

5 *Google Announces Google Docs & Spreadsheets*. News from Google, 2006. Available at: ⟨http://googlepress.blogspot.cz/2006/10/google-announces-google-docs_11.html⟩.

6 *Get more done, together with productivity apps in Google Drive*. Google Drive. Available at: ⟨https://www.google.com/intl/en_US/drive/start/apps.html⟩.

7 *Google Drive Help Center*. Google Drive. Available at: ⟨https://support.google.com/drive/?hl=en⟩.

8 GREGORIO, J. et al. *Google Data APIs Python Client Library*. Google Data API Team, 2011. Available at: ⟨http://code.google.com/p/gdata-python-client⟩.

9 BASS, L.; CLEMENTS, P.; KAZMAN, R. Understanding Quality Attributes in *Software Architecture in Practice*. Boston: Addison-Wesley, 2003, pp. 71–98. ISBN 9780321154958.

10 CLARK, J. *Trang – Multi-format schema converter based on RELAX NG*. Thai Open Source Software Center Ltd, 2008. Available at: ⟨http://www.thaiopensource.com/relaxng/trang.html⟩.

11 BRAY, T. et al. *Extensible Markup Language (XML) 1.0 (Fifth Edition)*. The World Wide Web Consortium (W3C), 2008. Available at: ⟨http://www.w3.org/TR/2008/REC-xml-20081126⟩.

12 BRAY, T. et al. *Extensible Markup Language (XML) 1.1 (Second Edition)*. The World Wide Web Consortium (W3C), 2006. Available at: ⟨http://www.w3.org/TR/2006/REC-xml11-20060816⟩.

13 *CDX Format Specification: Introduction*. CambridgeSoft. Available at: ⟨http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/General.htm⟩.

14 *The CDXML text-based file format*. CambridgeSoft. Available at: ⟨http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/IntroCDXML.htm⟩.

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.1335v1 | CC-BY 4.0 Open Access | rec: 30 Aug 2015, publ: 30 Aug 2015

PeerJ PrePrints

15 *CDX Format Specification*. CambridgeSoft. Available at: ⟨http://www.cambridgesoft.com/services/documentation/sdk/chemdraw/cdx/index.htm⟩.

16 *ChemDoodle: Features - Universality*. iChemLabs. Available at: ⟨http://www.chemdoodle.com/features/universality⟩.

17 *ChemDoodle: Read and Write ChemDraw Files*. iChemLabs. Available at: ⟨http://www.ichemlabs.com/114⟩.

18 *ChemSketch: A complete software package for drawing chemical structures.* ACD Labs. Available at: ⟨http://acdlabs.com/products/draw_nom/draw/chemsketch⟩.

19 *Instant JChem Developer's Guide*. ChemAxon. Available at: ⟨http://www.chemaxon.com/instantjchem/ijc_latest/docs/developer/index.html⟩.

20 *File formats in Marvin*. ChemAxon. Available at: ⟨https://docs.chemaxon.com/display/marvinsketch/File+formats+in+Marvin⟩.

21 O'BOYLE, N. M. et al. Open Babel: An open chemical toolbox. *Journal of Cheminformatics*. 2011, vol. 3, pp. 33. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-3-33⟩.

22 MURRAY-RUST, P.; RZEPA, H. S. Chemical Markup, XML, and the Worldwide Web. 1. Basic Principles. *J. Chem. Inf. Comput. Sci.* 1999, vol. 39, pp. 928–942. Available also at: ⟨http://dx.doi.org/10.1021/CI990052B⟩.

23 MURRAY-RUST, P. *CML Frequently asked Questions*. 1997. Available at: ⟨http://www.ch.ic.ac.uk/omf/cml/doc/faq/index.html⟩.

24 MURRAY-RUST, P.; RZEPA, H. S. *What is CML?* Available at: ⟨http://www.ch.ic.ac.uk/rzepa/chimeral/documents/FAQ/FAQ.html⟩.

25 MURRAY-RUST, P.; RZEPA, H. S. *Chemical Markup Language. A Position Paper.* 2001. Available at: ⟨http://cml.sourceforge.net/historical/position.html⟩.

26 LIAO, Y.-M.; GHANADAN, H. The chemical markup language. *Analytical Chemistry*. 2002, vol. 74, pp. 389A–390A. Available also at: ⟨http://dx.doi.org/10.1021/AC0220676⟩.

27 MURRAY-RUST, P.; RZEPA, H. S. CML: Evolution and design. *Journal of Cheminformatics*. 2011, vol. 3, pp. 44. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-3-44⟩.

28 MURRAY-RUST, P.; RZEPA, H. S. Chemical Markup, XML, and the World Wide Web. 4. CML Schema. *J. Chem. Inf. Comput. Sci.* 2003, vol. 43, pp. 757–772. Available also at: ⟨http://dx.doi.org/10.1021/CI0256541⟩.

29 *Schemas for CML and Related Languages*. 2005. Available at: ⟨http://cml.sourceforge.net/schema⟩.

30 HOLLIDAY, G. L.; MURRAY-RUST, P.; RZEPA, H. S. Chemical Markup, XML, and the World Wide Web. 6. CMLReact, an XML Vocabulary for Chemical Reactions. *J. Chem. Inf. Model.* 2006, vol. 46, pp. 145–157. Available also at: ⟨http://dx.doi.org/10.1021/CI0502698⟩.

31 KUHN, S. et al. Chemical Markup, XML, and the World Wide Web. 7. CMLSpect, an XML Vocabulary for Spectral Data. *J. Chem. Inf. Model.* 2007, vol. 47, pp. 2015–2034. Available also at: ⟨http://dx.doi.org/10.1021/CI600531A⟩.

32 ADAMS, N. et al. Chemical Markup, XML and the World-Wide Web. 8. Polymer Markup Language. *J. Chem. Inf. Model.* 2008, vol. 48, pp. 2118–2128. Available also at: ⟨http://dx.doi.org/10.1021/CI8002123⟩.

33 MURRAY-RUST, P.; RZEPA, H. S. STMML. A markup language for scientific, technical and medical publishing. *Data Science Journal*. 2002, vol. 1, pp. 128–192. Available also at: ⟨http://dx.doi.org/10.2481/DSJ.1.128⟩.

34 MURRAY-RUST, P.; RZEPA, H. S. *Chemical Markup Language Schema*. Available at: ⟨http://www.xml-cml.org/schema/⟩.

35 MURRAY-RUST, P.; RZEPA, H. S. *Chemical Markup Language Specifications*. Available at: ⟨http://www.xml-cml.org/spec⟩.

PeerJ PrePrints

36 HANWELL, M. D. et al. Avogadro: An advanced semantic chemical editor, visualization, and analysis platform. *Journal of Cheminformatics*. 2012, vol. 4, pp. 17. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-4-17⟩.

37 *Avogadro: An open-source molecular builder and visualization tool*. 2012. Available at: ⟨http://avogadro.openmolecules.net⟩.

38 KRAUSE, S.; WILLIGHAGEN, E.; STEINBECK, C. JChemPaint - Using the Collaborative Forces of the Internet to Develop a Free Editor for 2D Chemical Structures. *Molecules*. 2000, vol. 5, pp. 93–98. Available also at: ⟨http://dx.doi.org/10.3390/50100093⟩.

39 WILLIGHAGEN, E. L. Processing CML conventions in Java. *Internet Journal of Chemistry*. 2001, vol. 4, pp. article 4. Available also at: ⟨http://www.openscience.org/~egonw/cml/cml_conventions.html⟩.

40 *Jmol: an open-source Java viewer for chemical structures in 3D*. Available at: ⟨http://www.jmol.org⟩.

41 MURRAY-RUST, P.; RZEPA, H. S. *Chemical Markup Language - CML-aware software*. Available at: ⟨http://www.xml-cml.org/tools/software.html⟩.

42 DALBY, A. et al. Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited. *J. Chem. Inf. Comput. Sci.* 1992, vol. 32, pp. 244–255. Available also at: ⟨http://dx.doi.org/10.1021/CI00007A012⟩.

43 *CTfile Formats*. Accelrys. Available at: ⟨http://accelrys.com/products/informatics/cheminformatics/ctfile-formats/no-fee.php⟩.

44 *Symyx - CTFile Formats*. 2007. Available at: ⟨http://web.uni-plovdiv.bg/ksx/students/ISIS/ctfile.pdf⟩.

45 *Datasheet: Isentris Personal Edition*. Accelrys. Available at: ⟨http://accelrys.com/products/pdf/isentris-personal.pdf⟩.

46 ERTL, P. Molecular structure input on the web. *Journal of Cheminformatics*. 2010, vol. 2, pp. 1. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-2-1⟩.

47 ERTL, P. *JME Molecular Editor*. 2012. Available at: ⟨http://www.molinspiration.com/jme⟩.

48 *PerlMol - Perl Modules for Molecular Chemistry*. Available at: ⟨http://www.perlmol.org⟩.

49 *PyMOL - View 3D Molecular Structures*. Available at: ⟨http://www.pymol.org/view⟩.

50 *PyMOLWiki - Command Line Options*. Available at: ⟨http://pymolwiki.org/index.php/Command_Line_Options⟩.

51 *RasMol and OpenRasMol - Molecular Graphics Visualisation Tool*. Available at: ⟨http://www.openrasmol.org⟩.

52 *Crystallographic Information Framework*. The International Union of Crystallography (IUCr). Available at: ⟨http://www.iucr.org/resources/cif⟩.

53 *CIF Version 1.1 Working specification*. The International Union of Crystallography (IUCr). Available at: ⟨http://www.iucr.org/resources/cif/spec/version1.1⟩.

54 HALL, S. R.; ALLEN, F. H.; BROWN, I. D. The crystallographic information file (CIF): a new standard archive file for crystallography. *Acta Crystallographica Section A Foundations of Crystallography*. 1991, vol. 47, pp. 655–685. Available also at: ⟨http://dx.doi.org/10.1107/S010876739101067X⟩.

55 BROWN, I. D. CIF (crystallographic information file): A standard for crystallographic data interchange. *Journal of Research of the National Institute of Standards and Technology*. 1996, vol. 101, pp. 341–346. Available also at: ⟨http://nistdigitalarchives.contentdm.oclc.org/cdm/ref/collection/p13011coll6/id/48323⟩.

56 BROWN, I. D.; MCMAHON, B. CIF: the computer language of crystallography. *Acta Crystallographica Section B Structural Science*. 2002, vol. 58, pp. 317–324. Available also at: ⟨http://dx.doi.org/10.1107/S0108768102003464⟩.

57 BOURNE, P. E. et al. The Macromolecular Crystallographic Information File (mmCIF) in *Macromolecular Crystallography Part B*. San Diego: Academic Press, 1997, pp. 571–590. Available also at: ⟨http://mmcif.wwpdb.org/docs/pubs/methods-enzymology-paper-1997.html⟩. ISBN 9780121821784.

PeerJ PrePrints

58 WESBROOK, J. D.; BOURNE, P. E. STAR/mmCIF: An Ontology for Macromolecular Structure. *Bioinformatics*. 2000, vol. 16, pp. 159–168. Available also at: ⟨http://dx.doi.org/10.1093/BIOINFORMATICS/16.2.159⟩.

59 *The IUPAC International Chemical Identifier (InChI)*. International Union of Pure and Applied Chemistry (IUPAC). Available at: ⟨http://www.iupac.org/home/publications/e-resources/inchi.html⟩.

60 *The InChI Standard - Technical FAQ*. InChI Trust. Available at: ⟨http://www.inchi-trust.org/technical-faq⟩.

61 *About the InChI Standard*. InChI Trust. Available at: ⟨http://www.inchi-trust.org/about-the-inchi-standard⟩.

62 *ASN.1 File Format (Summary)*. National Center for Biotechnology Information (NCBI), 2014. Available at: ⟨http://www.ncbi.nlm.nih.gov/Structure/asn1.html⟩.

63 *NCBI data specifications*. National Center for Biotechnology Information (NCBI), 2014. Available at: ⟨http://www.ncbi.nlm.nih.gov/data_specs⟩.

64 *NCBI Data in XML*. National Center for Biotechnology Information (NCBI), 2005. Available at: ⟨http://www.ncbi.nlm.nih.gov/data_specs/NCBI_data_in_XML.html⟩.

65 *NCBI Data in XML - NCBI ToolBox*. National Center for Biotechnology Information (NCBI), 2004. Available at: ⟨http://www.ncbi.nlm.nih.gov/IEB/ToolBox/XML/ncbixml.txt⟩.

66 *Protein Data Bank Contents Guide: Atomic Coordinate Entry Format Description*. The Worldwide Protein Data Bank (wwPDB), 2011. Available at: ⟨http://www.wwpdb.org/documentation/format33/v3.3.html⟩.

67 *The Worldwide Protein Data Bank File Format Documentation*. The Worldwide Protein Data Bank (wwPDB). Available at: ⟨http://www.wwpdb.org/documentation/file-format⟩.

68 *PDBx/mmCIF General FAQ*. Available at: ⟨http://mmcif.wwpdb.org/docs/faqs/pdbx-mmcif-faq-general.html⟩.

69 WESBROOK, J. et al. PDBML: the representation of archival macromolecular structure data in XML. *Bioinformatics*. 2005, vol. 21, pp. 988–992. Available also at: ⟨http://dx.doi.org/10.1093/BIOINFORMATICS/BTI082⟩.

70 *PDBML Schema Resources*. The Worldwide Protein Data Bank (wwPDB). Available at: ⟨http://pdbml.pdb.org⟩.

71 *SMILES – Simplified Molecular Input Line Entry System*. Daylight Chemical Information Systems, 2008. Available at: ⟨http://www.daylight.com/smiles⟩.

72 *OpenSMILES Home Page*. Blue Obelisk community. Available at: ⟨http://www.opensmiles.org⟩.

73 CRAIG, A. J. et al. *OpenSMILES Specification DRAFT*. Blue Obelisk community, 2007. Available at: ⟨http://www.opensmiles.org/spec/open-smiles.html⟩.

74 CRAIG, A. J. et al. *OpenSMILES specification*. Blue Obelisk community, 2012. Available at: ⟨http://www.opensmiles.org/opensmiles.html⟩.

75 WEININGER, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* 1988, vol. 28, pp. 31–36. Available also at: ⟨http://dx.doi.org/10.1021/CI00057A005⟩.

76 *Daylight Theory Manual - 3. SMILES - A Simplified Chemical Language*. Daylight Chemical Information Systems, 2008. Available at: ⟨http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html⟩.

77 ASH, S. et al. SYBYL Line Notation (SLN): A Versatile Language for Chemical Structure Representation. *J. Chem. Inf. Comput. Sci.* 1997, vol. 37, pp. 71–79. Available also at: ⟨http://dx.doi.org/10.1021/ci960109j⟩.

78 HOMER, R. W. et al. SYBYL Line Notation (SLN): A Single Notation To Represent Chemical Structures, Queries, Reactions, and Virtual Libraries. *J. Chem. Inf. Model.* 2008, vol. 48, pp. 2294–2307. Available also at: ⟨http://dx.doi.org/10.1021/ci7004687⟩.

79 *Tripos Mol2 File Format*. Tripos. Available at: ⟨http://tripos.com/index.php?family=modules,SimplePage,Mol2_File_Format2009⟩.

PeerJ PrePrints

80 *Tripos Mol2 File Format - Sample Mol2 File*. Tripos. Available at: ⟨http://www.tripos.com/mol2/mol2_format3.html⟩.

81 MURRAY-RUST, P.; RZEPA, H. S.; WRIGHT, M. Development of chemical markup language (CML) as a system for handling complex chemical content. *New J. Chem.* 2001, vol. 25, pp. 618–634. Available also at: ⟨http://dx.doi.org/10.1039/B008780G⟩.

82 CARLISLE, D.; ION, P.; MINER, R. *Mathematical Markup Language (MathML) Version 3.0 2nd Edition*. The World Wide Web Consortium (W3C), 2014. Available at: ⟨http://www.w3.org/TR/2014/REC-MathML3-20140410⟩.

83 DAHLSTRÖM, E. et al. *Scalable Vector Graphics (SVG) 1.1 (Second Edition)*. The World Wide Web Consortium (W3C), 2011. Available at: ⟨http://www.w3.org/TR/2011/REC-SVG11-20110816⟩.

84 *Units Markup Language (UnitsML)*. National Institute of Standards and Technology (NIST), 2011. Available at: ⟨http://unitsml.nist.gov⟩.

85 CHEPELEV, L. L.; DUMONTIER, M. Chemical Entity Semantic Specification: Knowledge representation for efficient semantic cheminformatics and facile data integration. *Journal of Cheminformatics*. 2011, vol. 3, pp. 20. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-3-20⟩.

86 TOWNSEND, J. A.; MURRAY-RUST, P. CMLLite: a design philosophy for CML. *Journal of Cheminformatics*. 2011, vol. 3, pp. 39. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-3-39⟩.

87 MURRAY-RUST, P. et al. The semantics of Chemical Markup Language (CML): dictionaries and conventions. *Journal of Cheminformatics*. 2011, vol. 3, pp. 43. Available also at: ⟨http://dx.doi.org/10.1186/1758-2946-3-43⟩.

88 *Schematron – A language for making assertions about patterns found in XML documents*. Available at: ⟨http://www.schematron.com/index.html⟩.

89 *ISO/IEC 19757-3:2006 Information technology – Document Schema Definition Language (DSDL) – Part 3: Rule-based validation – Schematron*. International Organization for Standardization (ISO), 2006. Available at: ⟨http://standards.iso.org/ittf/PubliclyAvailableStandards/c040833_ISO_IEC_19757-3_2006(E).zip⟩.

90 ROBERTSSON, E. *An Introduction to Schematron*. O'Reilly XML.com, 2003. Available at: ⟨http://www.xml.com/pub/a/2003/11/12/schematron.html⟩.

91 CLARK, J.; MAKOTO, M. *RELAX NG Specification*. Organization for the Advancement of Structured Information Standards (OASIS), 2001. Available at: ⟨http://relaxng.org/spec-20011203.html⟩.

92 VAN DER VLIST, E. *RELAX NG*. Sebastopol: O'Reilly Media, 2003. 304 pp. Available also at: ⟨http://books.xmlschemata.org/relaxng/page2.html⟩. ISBN 9780596004217.

93 FALLSIDE, D. C.; WALMSLEY, P. *XML Schema Part 0: Primer Second Edition*. The World Wide Web Consortium (W3C), 2004. Available at: ⟨http://www.w3.org/TR/2004/REC-xmlschema-0-20041028⟩.

94 LEE, D.; CHU, W. W. *Comparative Analysis of Six XML Schema Languages*. Department of Computer Science University of California, 2000. Available at: ⟨http://www.cobase.cs.ucla.edu/tech-docs/dongwon/ucla-200008.html⟩.

95 PAWSON, D. *An introduction to NVDL, ISO 19757-4*. 2008. Available at: ⟨http://www.dpawson.co.uk/nvdl/index.html⟩.

96 *ISO/IEC 19757-4 NVDL (Namespace-based Validation Dispatching Language)*. 2009. Available at: ⟨http://nvdl.org⟩.

97 *ISO/IEC 19757-4:2006 Information technology – Document Schema Definition Languages (DSDL) – Part 4: Namespace-based Validation Dispatching Language (NVDL)*. International Organization for Standardization (ISO), 2006. Available at: ⟨http://standards.iso.org/ittf/PubliclyAvailableStandards/c038615_ISO_IEC_19757-4_2006(E).zip⟩.

98 MOKRÝ, J.; NIČ, M. Universal Chemical Markup (UCM) - A new format for common chemical data. *Journal of Cheminformatics*. In press.

PeerJ PrePrints

99 PEMBERTON, S. et al. *XHTML 1.0: The Extensible HyperText Markup Language (Second Edition)*. The World Wide Web Consortium (W3C), 2002. Available at: ⟨http://www.w3.org/TR/2002/REC-xhtml1-20020801⟩.

100 RAYMOND, E. S. The Importance of Being Textual in *The Art of Unix Programming*. Boston: Addison-Wesley, 2003, pp. 107–111. Available also at: ⟨http://www.faqs.org/docs/artu/ch05s01.html⟩. ISBN 9780131429017.

101 *Desktop Support - System Requirements - ChemDraw*. CambridgeSoft. Available at: ⟨http://www.cambridgesoft.com/services/DesktopSupport/KnowledgeBase/SystemRequirements/Default.aspx?fid=14⟩.

102 MARSH, J.; VEILLARD, D.; WALSH, N. *xml:id Version 1.0*. The World Wide Web Consortium (W3C), 2005. Available at: ⟨http://www.w3.org/TR/2005/REC-xml-id-20050909⟩.

103 BRAY, T. et al. *Namespaces in XML 1.0 (Third Edition)*. The World Wide Web Consortium (W3C), 2009. Available at: ⟨http://www.w3.org/TR/2009/REC-xml-names-20091208⟩.

104 BRAY, T. et al. *Namespaces in XML 1.1 (Second Edition)*. The World Wide Web Consortium (W3C), 2006. Available at: ⟨http://www.w3.org/TR/2006/REC-xml-names11-20060816⟩.

105 MURRAY-RUST, P.; RZEPA, H. S. *Chemical Rendering using SVG and CML*. 2000. Available at: ⟨http://www.ch.ic.ac.uk/svg⟩.

PeerJ PrePrints