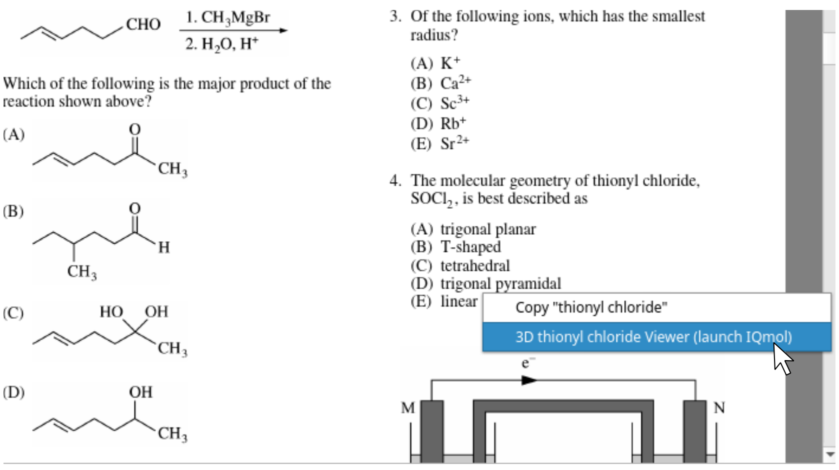
An ETS Plugin Framework for Enhanced Test Preparation

We propose an ETS Plugin Framework (**ETSPF**) whose goal is to integrate applications (for example, **PDF** viewers) for viewing documents viewers with scientific and multi-media applications by enabling these applications to share data, and also to incorporate **GUI** components specific to education and test-prep. ETS plugins would allow scientific and applications to be used as teaching tools, and also allow document viewers to present multimedia content to a degree that is not currently possible — in particular, interoperating with a diverse array of applications supporting domain-specific multimedia file formats. In effect, ETS plugins transform document viewers into specialized e-readers with with novel test-preparation features. For example, while studying for exams, students would have at their disposal interactive multimedia presentations which would offer sophisticated data visualization and **3D** graphics tools.

ETSPF for Scientific and Technical Applications

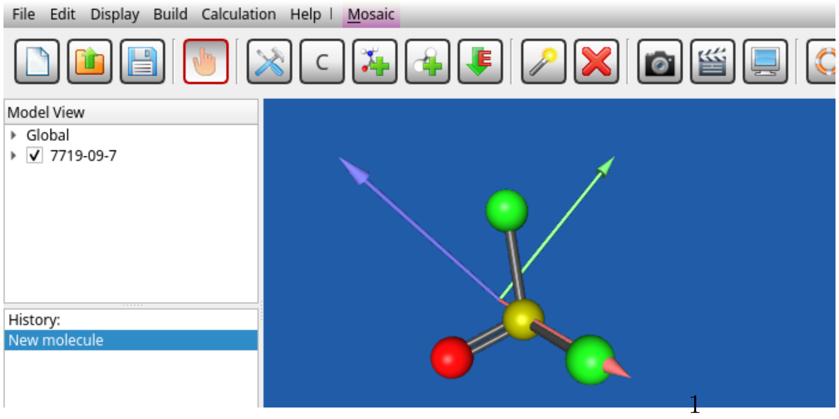
How ETSpf enables multi-application networking

ETSPF refers not to a single plugin, but a toolkit for implementing ETS plugins to be embedded in many different applications. These plugins should be sufficiently similar that students or instructors familiar with an ETS plugin in one context (chemistry, for example) would quickly understand how to use plugins present in a different context. An important **ETSPF** feature is that distinct ETS plugins would be able to communicate with each other. In particular, plugins for document viewers would send data to plugins for scientific or multimedia applications, so that students could access multimedia content linked to manuscripts, such as test-preparation materials, that they are currently reading.



For a concrete example of advanced functionality that can be achieved by connecting two distinct **ETSPF** plugins, consider a student reading through the ETS **GRE** Chemistry practice test. This book has sample questions such as (number 4, page 11) **The molecular geometry of thionyl chloride, $SOCl_2$, is best described as** *(A) trigonal planar, (B) T-shaped, (C) tetrahedral, (D) trigonal pyramidal, or (E) linear*. To understand this question/answer, it may help students to view a **3D** model of thionyl chloride, which can be done with the aid of molecular visualization software, such as IQmol. Accordingly, this specific question in the book may be associated with Molecular Data file for $SOCl_2$ (this file is available from the "Chemical Abstracts Service" database). The relation between the specific textual location (where the practice Question 4 is presented) and the supplemental Molecular Data file would be asserted in the Semantic Document Infoset, and read by a document viewer (e.g., **XPDF**). The **XPDF** plugin would then launch IQmol and send the molecular file to the IQmol ETS plugin, with instructions to load this file into an IQmol session (see Figure 1). The end result would be that the student, with a single click (such as selecting a visualization action from a context menu on the practice question) has access to

Figure 1: Thionyl Chloride in IQmol



an interactive 3D graphic representing thionyl chloride. (Of course, analogous functionality would be available for any chemical compound with multimedia files in such formats as Molecular Data, Protein Data Bank, or Chemical Markup Language).

ETSpf features for keeping track of students' previous activity.

The data sent between ETSPF applications may be more complex than a request to open a single multimedia file. Suppose a student reading through the GRE Chemistry practice exam launches IQmol a second time — perhaps in conjunction with a later question (number 95 in the test — see figure at right) about the molecular structure of lactose. In this case, the plugin can send information not only about the present request but about the student's prior usage; in particular the fact that he or she had previously viewed the $SOCl_2$ file. The ETSPF plugin on the IQmol side can then load the prior file along with the new one, so the student can browse back to prior application-states if desired (see the Model View panel on Figure 3).

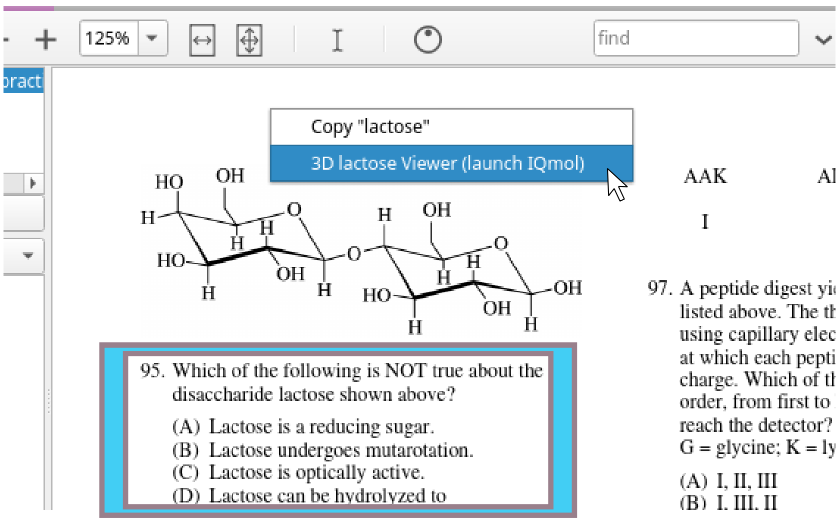
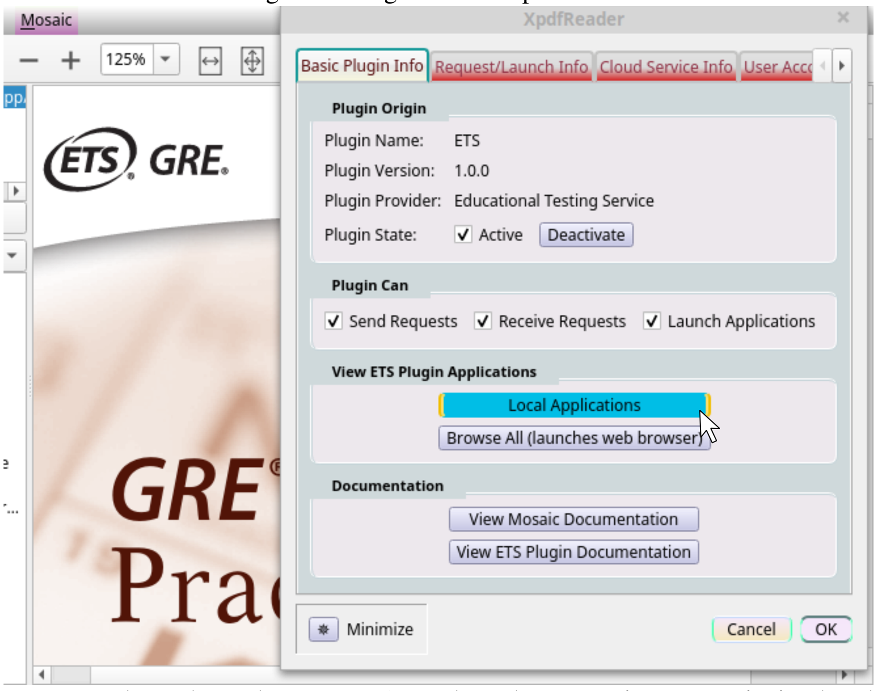


Figure 2: Plugin Info in XpdfReader

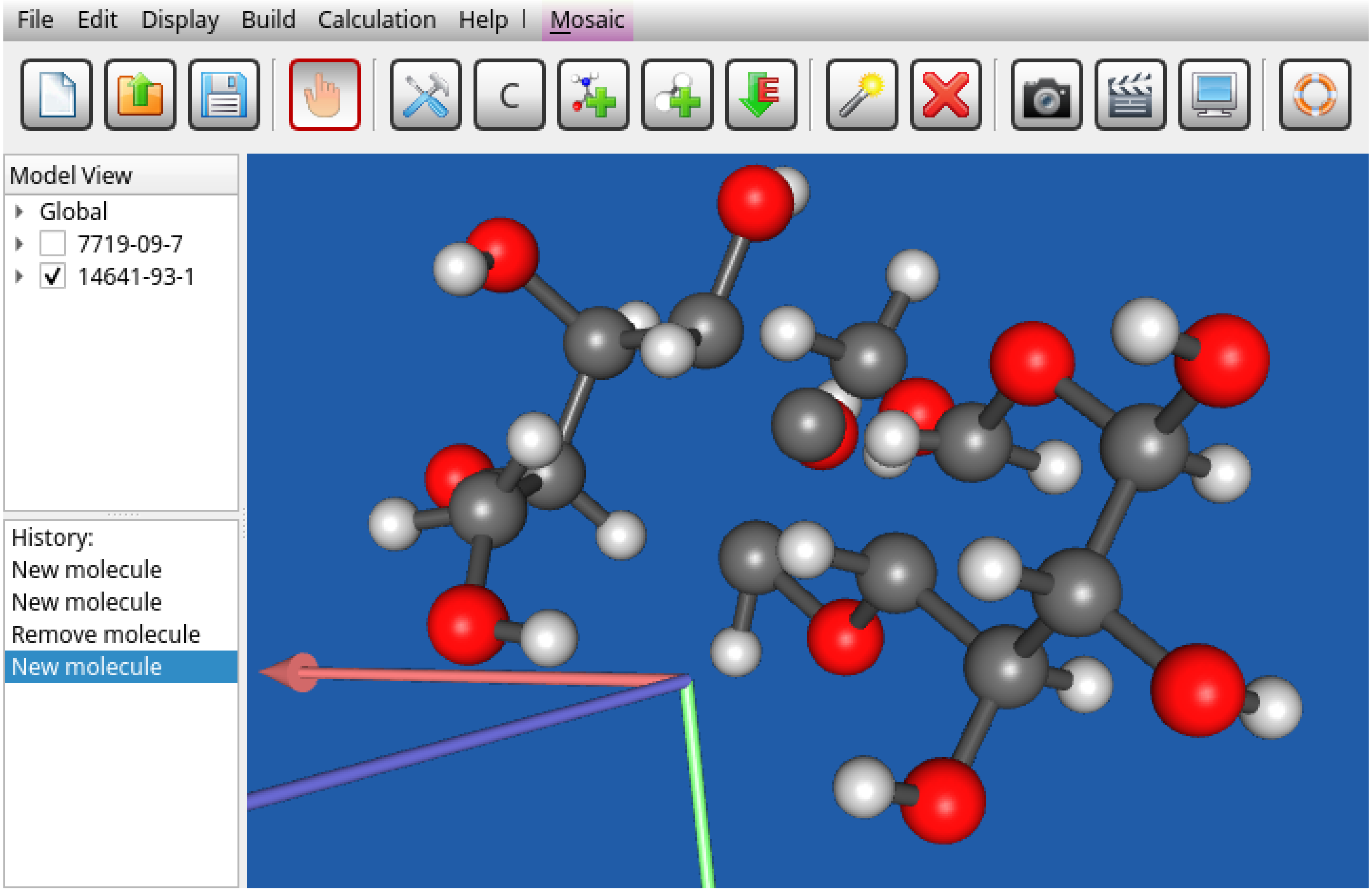


In general, the functionality provided by each ETS plugin will depend in part on domain of the host application in which the plugin is embedded. For example, an IQmol plugin would load cheminformatic files and may activate IQmol's analytic capabilities in the domain of chemistry, whereas a plugin for applications in the domain of quantitative/statistical analysis and data visualization (such as ParaView) would open quantitative data sets with 2D or 3D views (via surfaces, scatter-plots, bar charts, etc.) and perhaps activate statistical calculations. Nevertheless, certain functionality would be shared among all ETS plugins, which would include a dialog window to show basic plugin information (see Figure 2) as well as a more detailed review of data transmitted between applications via plugins. Specifically, the "request info" tab would allow students, instructors, and plugin developers to see information about the request which prompted the current application to be launched and/or to open a specific file (see Figure 5).

ETSpf Tools for Composing Test-Preparation Materials

In addition to linking document viewers with scientific and multimedia applications, ETS plugins can add features to document viewers so that they become more effective teaching tools. For example, plugins can introduce question/answer windows that show practice questions (and their multiple-choice answers) one at a time. As depicted in Figure 4, these question/answer windows can be

Figure 3: Lactose in IQmol



used in combination with book-like test-preparation materials (here the **GRE** Chemistry practice test published by ETS). Students can browse through the test using the main **PDF** viewer, but can also choose to view each question in its own window, which may help them stay focused on each question in isolation.

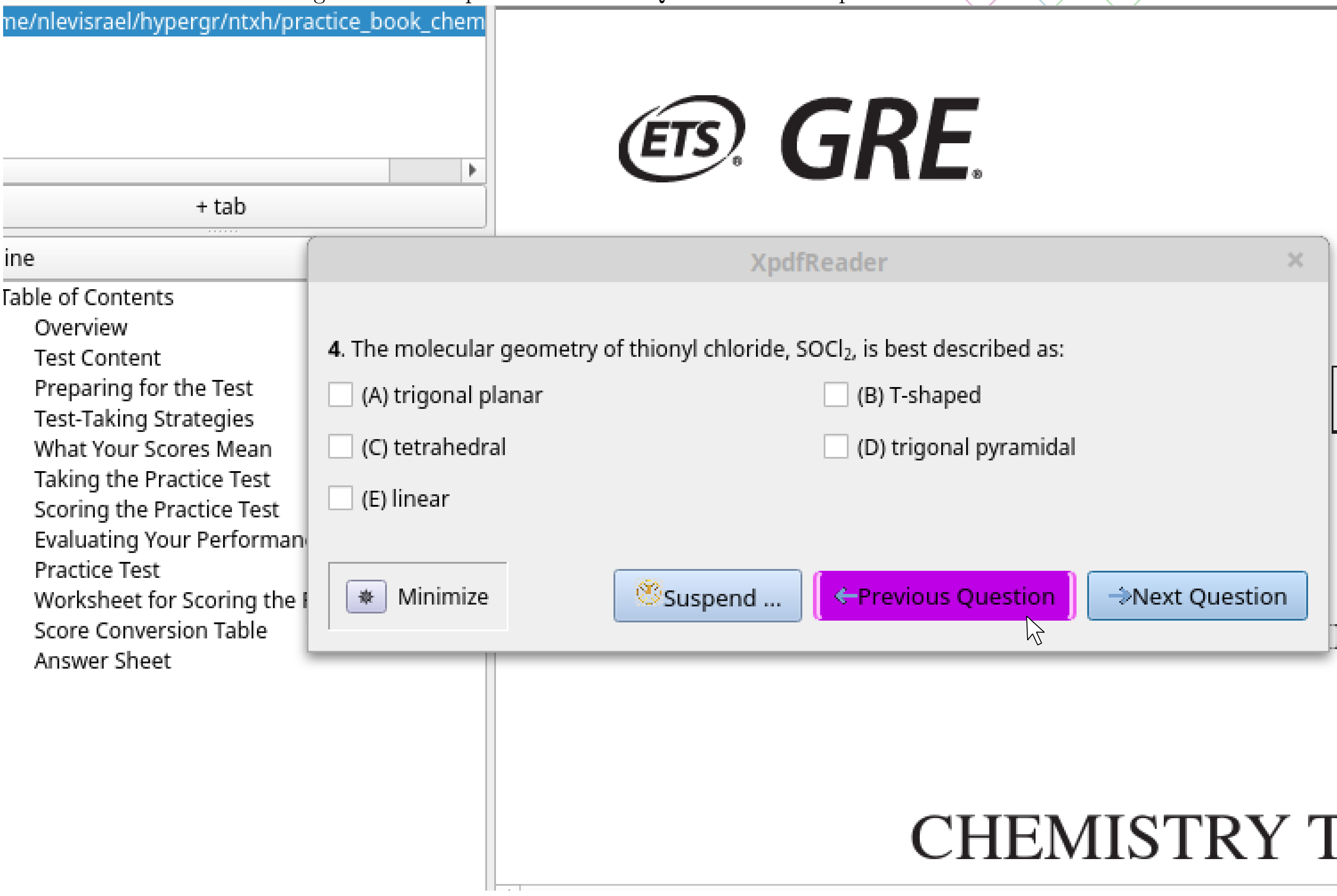
To support this technology, each publication could provide what we call a "Semantic Document Infoset" (**SDI**), which divides manuscripts into textual units (sections, paragraphs, sentences, etc.) and identifies document elements such as technical terms (which may be compiled into a glossary) and figure illustrations. **ETSPF** code can then examine a publication's **SDI** to generate machine-readable structural representations of publication manuscripts, which document viewers may use to augment the underlying document with additional instructional and/or multimedia features — review questions, student instructions, glossaries, reading assignments, and so forth.

ETSPF implementations can include LaTeX packages which automate the creation of **SDI** data (placed as an embedded file in the generated **PDF** document). This embedded data can then be read by **ETSPF** plugins to compose multi-application networking requests, populate question/answer windows, or introduce other kinds of pedagogic content (e.g., review questions, glossaries, or class discussion suggestions). In manuscripts where questions are printed as part of the publication text (for example, the ETS **GRE** practices), the LaTeX code can store questions' **PDF** coordinates so that the document automatically scrolls while students work their way through a practice test session. Alternatively, the same techniques can be used to add review questions and answers to documents which are not expressly designed as test-prep materials, such as textbooks and research papers. In this latter case, question/answer windows may be synced to sentences or paragraphs in those publications which are relevant to the review question that the student is currently reading in a question/answer window.

HTXN and multimedia content

To enable **SDI** infosets, **ETSPF** plugins would implement a protocol which we call **HTXN** (for "Hypergraph Text Encoding"). The goal of **HTXN** is to support the new generation of publishing technologies, where conventional document formats are increasingly being supplanted by digital, multimedia reader experiences. The traditional manuscript (the "primary" resource which is cited and downloaded) is, accordingly, often networked with a package of supplemental (or "secondary") resources. **HTXN** is designed to rigorously document these multimedia networks, enabling e-readers

Figure 4: A Sample Practice-Test Question within XpdfReader



and domain-specific applications to be integrated so that users may easily access multimedia content.

HTXN specifications

The **HTXN** protocol uses "standoff annotation" (i.e., character encoding and document structure are defined in isolation from one another), and can be employed to encode manuscripts in different markup formats (both LaTeX and **XML**, for instance). **HTXN** defines specifications for (1) encoding manuscripts' character data and (2) defining document structures via graphs whose nodes correspond to ranges in a character stream. **ETSPF** and **HTXN** are autonomous technologies (each may be used apart from the other), but they form a natural pairing, encompassing both the character/orthographic data and the document structure of manuscripts for publication.

Other Use-Cases for the ETS Plugin Framework

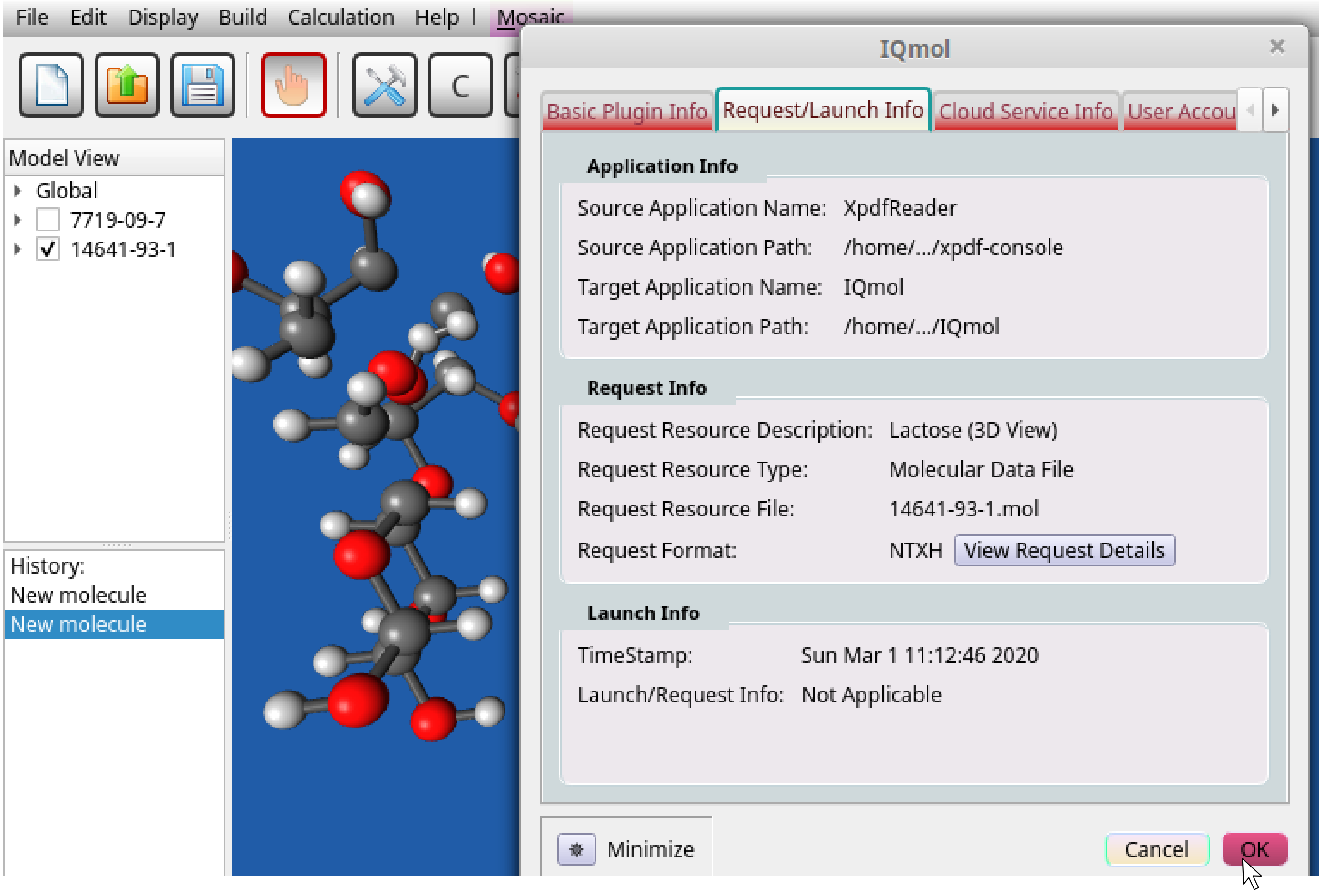
In addition to use-cases for ETS plugins reviewed above — students read textbooks, articles, or test-preparation materials which can be enhanced with multimedia content — consider the following scenarios:

Scenario 1: Using scientific applications as pedagogic tools Teachers often instruct students to download and install software relevant to the course curriculum, and this software can potentially be an essential part of the course content. Instructors may (1) use the visualization capabilities of these domain-specific applications to help students understand the concepts covered in class; (2) provide instruction in how to use the software as part of the curriculum; (3) evaluate students' understanding of the software as part of their assessment of students' mastery of the curriculum; and (4) use applications' analytic features as an overview of analytic or quantitative methodologies relevant to the course's subject matter. In the case of IQmol, features such as energy minimization, plotting orbitals, calculating vibrational frequencies, and many other chemphysical computations provide an overview of scientific concepts which might be covered in a Chemistry class.

To facilitate the use of scientific applications as teaching tools, **ETSPF** plugins effectively help instructors to personalize the applications which their students use in conjunction with course curricula. Again using IQmol as a case-study, an ETS plugin could show students a list of molecular examples discussed in class (based on data provided by the instructor) and allow students to view the corresponding **3D** molecular graphics in the application accordingly.

Scenario 2: Generating Test-Preparation Materials Computer code which operates on manuscripts composed in coordination with **ETSPF** can act in consort with **ETSPF** plugins. For teaching purposes, one may need to customize chapters, books, and articles by inserting review questions at strategic points in the text; compiling glossaries indicating when technical terms are first used; foregrounding graphics and figure illustrations and encouraging students to explore them in depth (for instance, performing for themselves the calculations which are plotted in a statistical diagram; or identifying the axes, scales of measurement, dimensions, free and dependent parameters, and other mathematical details of a quantitative illustration). Both of these kinds of operations — extracting or adding content — require computational access to a Semantic Document Infoset providing structured access to publication content. For example, review questions or glossary definitions may be linked to specific points in the text; this supplemental content may then be packaged along with the original publication. In short, with an appropriate **ETSPF** plugin, the document viewer can then process the review material and superimpose pedagogic questions, comments, or instructions on the underlying document text.

Figure 5: Request Information in IQmol



Scenario 3: Developing Educational Materials Targeting Specific Applications In this scenario, test-preparation materials are composed with the anticipation that readers will use specific software applications to enhance their reading experience. This sort of interoperability between publications and external software is assumed at a basic level as soon as documents are linked to specialized multimedia files, because readers will need an application which can open those files. For example, files in the ParaView Data (**PVD**) format are intended to be used by the ParaView software; as a result, documents which reference such files presuppose that readers will have ParaView installed on their computer, at least for the full reading experience which includes viewing ParaView graphics. Similarly, files in cheminformatic formats such as Protein Data Bank (.pdb)

or Chemical Markup Language (.cml) need to be opened with chemistry-related software (such as IQmol).

In some cases, however, authors may desire a more rigorous degree of interop between document viewers and scientific/multi-media software — beyond just sharing files with applications that can open the relevant file type. For instance, instructors may wish to foreground certain kinds of analyses which are pertinent to course curricula, or to group together files which are interrelated in the context of the course (e.g., based on study requirements for an upcoming class session). **ETSpf** would correspondingly allow more complex data structures to be shared share between applications, so that requests could trigger applications to enact multi-step response sequences: to reconfigure their state or layout, load groups of files, present specialized instructions to users, and so forth. ETS plugins would include a special format for encoding data structures to conveniently pass requests between host applications.