

We are proposing a novel ETS Plugin Framework (**ETSPF**) whose purpose is to enhance document viewers, in tandem with scientific and multimedia applications, to improve test preparation. In so doing, this plugin framework would allow document viewers to launch and share data with a diverse array of applications that exist for both scientific and social science/humanities disciplines, such as chemistry, physics, biology, medicine, linguistics, sociology, and literature. Document viewers would therefore be able to support an interactive, multimedia reading/studying experience to an unprecedented degree. In particular, students preparing for exams would have at their disposal stimulating multimedia presentations that offer sophisticated data visualization and **3D** graphics tools, customized for individual subjects: e.g., **3D** molecular models for chemistry, or **3D** tissue models for biology. In addition to offering multimedia features, ETS plugins could likewise enhance document viewers with instructional features that are supplemental to the documents which students are reading; for example, review questions, assignment instructions, or definitions of important concepts.

ETSPF for Scientific and Technical Applications

The **ETSpf** toolkit

ETSPF does not refer to a single plugin, but to a toolkit for implementing multiple ETS plugins to be embedded in many different scientific and social-scientific applications. These plugins should be sufficiently similar to one another so that students or instructors familiar with an ETS plugin in one context would quickly understand how to use ETS plugins found in a different context. One important feature of this framework is that distinct ETS plugins would be able to communicate with one another. For example, plugins for document viewers would send data to plugins for scientific or multimedia applications. In this way, students would be able to access multimedia content linked to the documents (viz., the test-preparation materials) that they are currently studying.

How **ETSpf** enables multi-application networking and interoperability

For a concrete example of advanced functionality that can be achieved by connecting two distinct EPF plugins, consider a student reading through the ETS **GRE** Chemistry practice test. This book has sample multiple-choice questions such as (on page 11, number 4), "**The molecular geometry of thionyl chloride, SOCl₂, is best described as (A) trigonal planar, (B) T-shaped, (C) tetrahedral, (D) trigonal**

pyramidal, or (E) linear". To understand this question and its corresponding multiple-choice answers, it would help students to be able to view a **3D** model of thionyl chloride, which can be done with the aid of molecular visualization software, such as IQmol. To support this functionality, our plugin within the document-viewer application (here **XPDF**) would launch IQmol, sending data through a corresponding ETS plugin embedded in IQmol. Specifically, question 4 in the practice test may be associated with a Molecular Data file for SOCl₂; the **XPDF** plugin would launch IQmol, sending along a data package identifying this SOCl₂ file to IQmol's own plugin, with

Figure 1: A Thionyl Chloride Question in XpdfReader

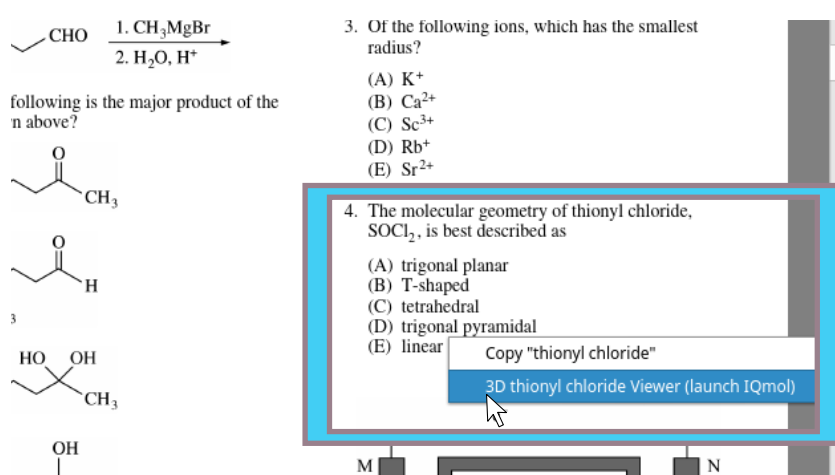
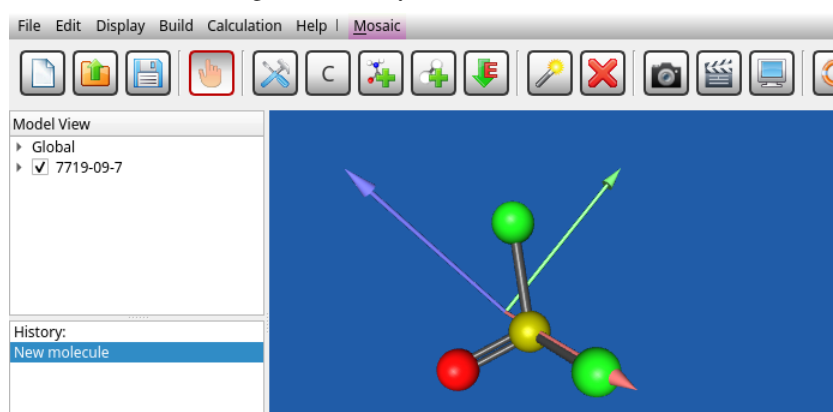


Figure 2: Thionyl Chloride in IQmol



instructions to load the file into an IQmol session (see Figure 2). The student could initiate this process by selecting a context-menu option (called "3D thionyl chloride Viewer" in Figure 1, the highlighted option where the cursor is pointing). The end result, then, would be that the student, with a single click, has access to an interactive **3D** graphic representing thionyl



chloride. The same functionality would be available for any chemical compound which has associated data in formats such as Molecular Data, Protein Data Bank, or Chemical Markup Language.

ETSpf features for keeping track of students' previous activity.

The previous case-study involving Thionyl Chloride exemplified a simple data structure transmitted between ETS plugins: specifically, the name of a single file to open. However, in other cases, the information sent between plugins might be more complex and detailed. To accommodate this, all ETS plugins would require a *common* vocabulary for representing multi-part data structures. For example, if a stu-

dent views a *second* molecule in IQmol, the document viewer should identify not only *that* file, but any *previous* files they had viewed, so that the student could conveniently refer back to those previous files as desired. This would be the case where a student reading through the GRE Chemistry practice exam chooses to launch IQmol a second time — perhaps in conjunction with a later question about the molecular structure of lactose, such as question number 95 in the test (see Figure 3 above). In this case, the plugin would send information not only about the present (lactose) request but also about the SOCl₂ (Thionyl Chloride) file that they had viewed earlier. This is visible within the Model View panel at the top-left on Figure 5, where the SOCl₂ file is listed above the checked lactose file (the lactose file is checked because it is the one currently seen in the view-port). As this example illustrates, the plugins sharing data about a student's prior actions makes both applications more interactive, ensuring that students benefit from a flexible and responsive User Experience.

Figure 3: A Lactose Question in XpdfReader

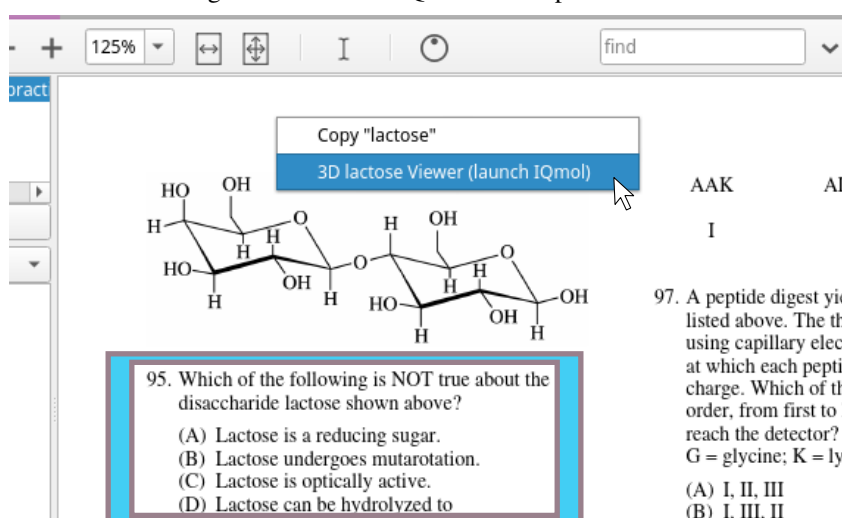
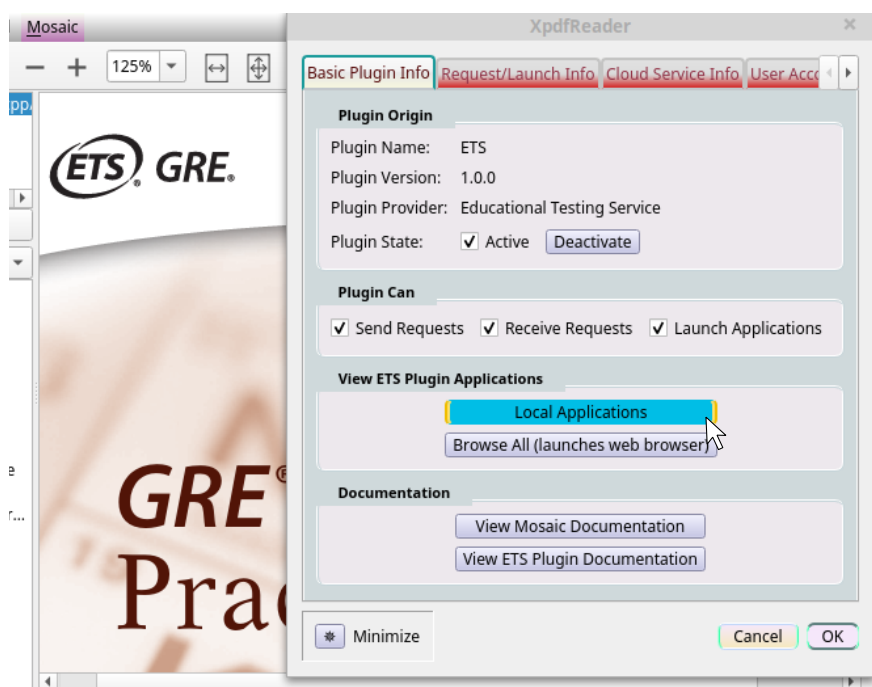


Figure 4: Plugin Info in XpdfReader



(with 2D or 3D views via surfaces, scatter-plots, bar charts, etc.). Nevertheless, certain functionality would be shared among all ETS plugins, which would include common data-sharing vocabulary (as mentioned above), as well as dialog windows to show basic plugin information (see Figure 4) alongside a more detailed review of the data that has been transmitted between applications via plugins. Specifically, the "Request/Launch 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 6).

In general, the functionality provided by each ETS plugin would depend in part on the 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 load quantitative data sets

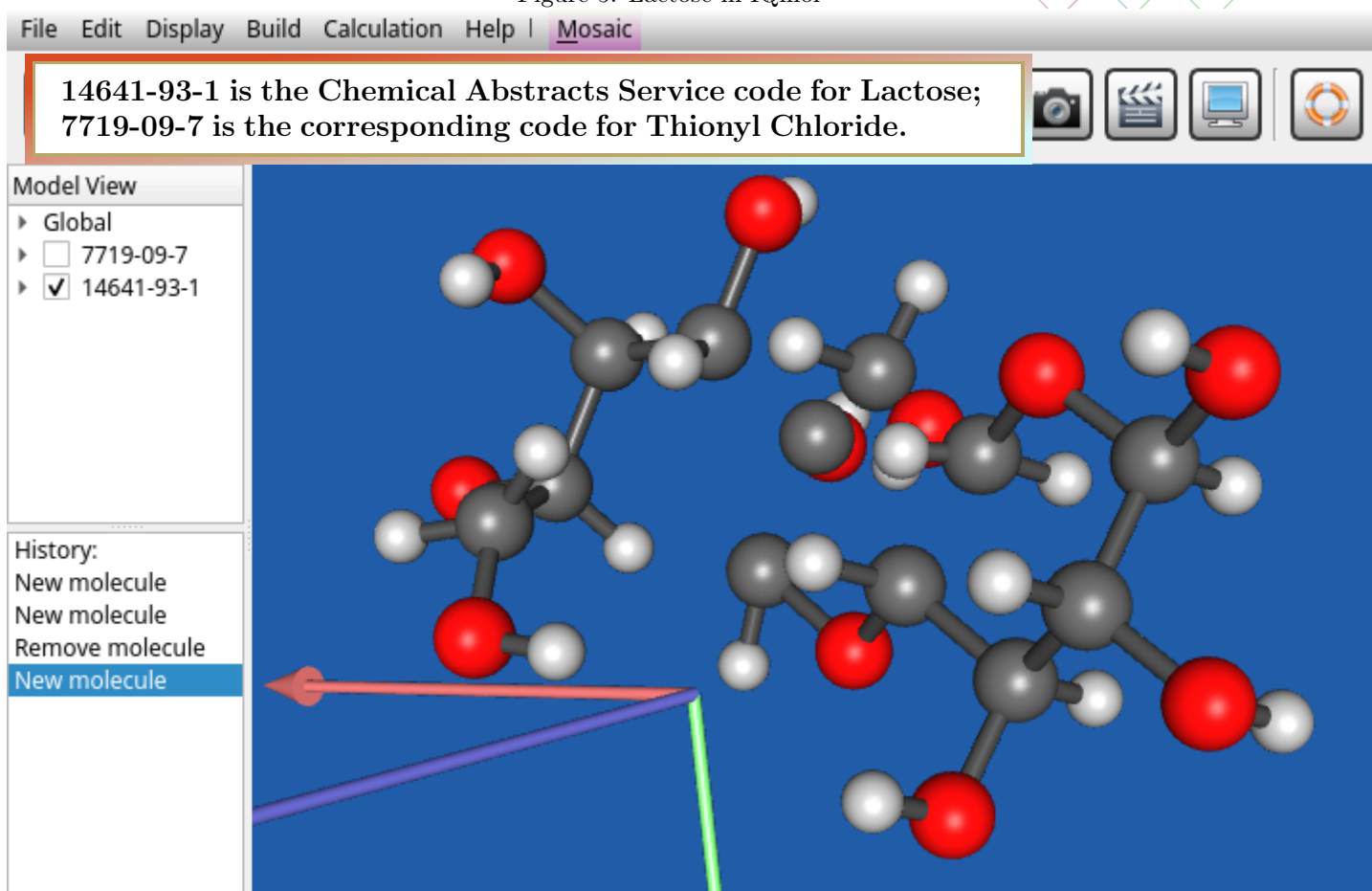
ETSpf data in embedded files

ETSpf Tools for Composing Test-Preparation Materials

In most cases, ETS plugins for document viewers such as **XPDF** would draw information from **PDF** files (or files in other formats, e.g. **EPUB** or **HTML**) to implement teaching enhancements, such as integration with scientific and multimedia applications. This **ETSPF**-specific data can be placed in a



Figure 5: Lactose in IQmol

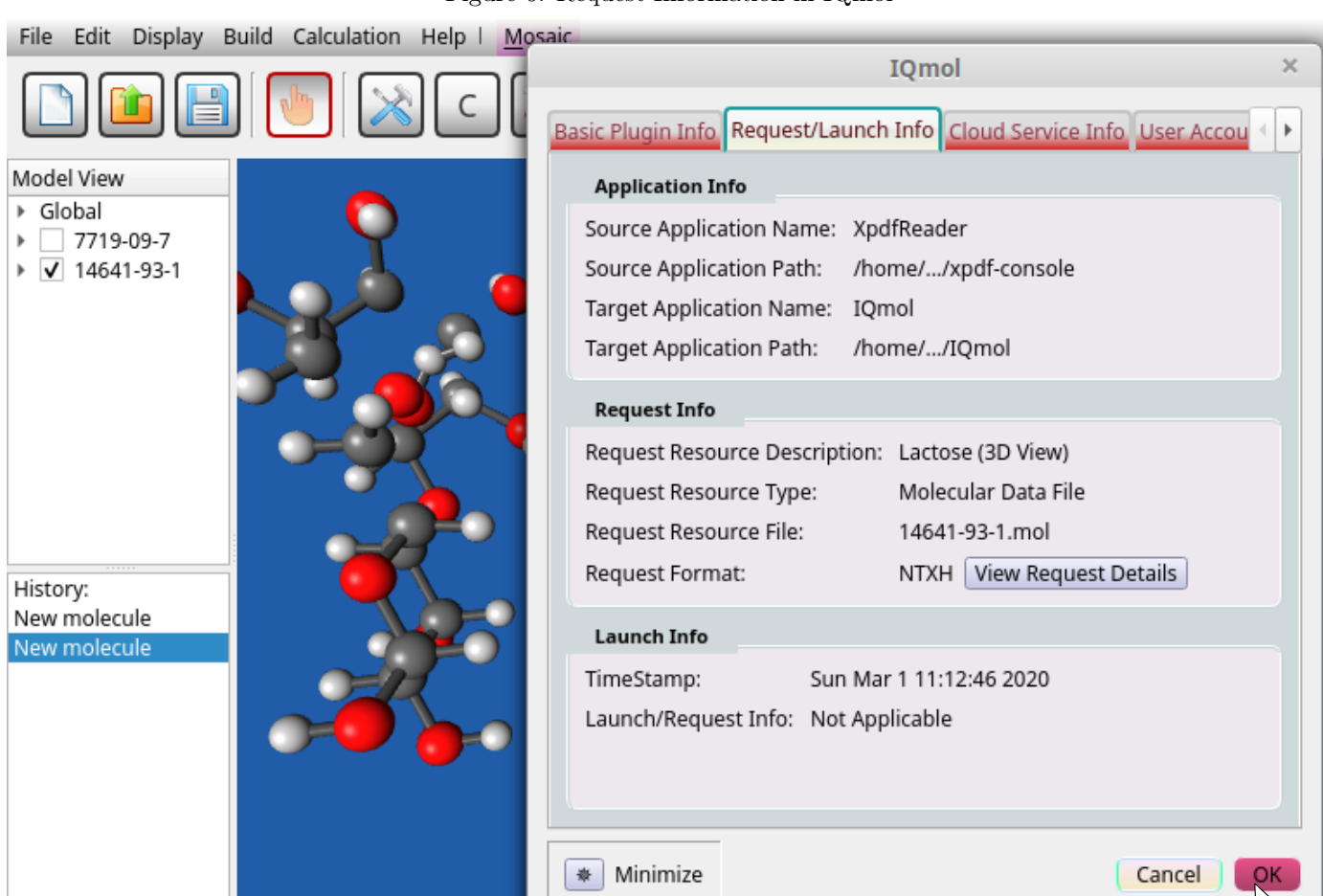


separate file embedded in **PDF** or **EPUB** documents, or (in **HTML**) inserted as non-display contents. When a document is opened, the ETS plugin would then extract the embedded file so as to read **ETSPF**-specific data about the document — in particular, to identify **PDF** coordinates for document elements requiring special **ETSPF** actions. For questions 4 and 95 as illustrated above, the relevant **ETSPF** action would be an option to view the question-specific molecular files in IQmol. **ETSPF** data is needed in order to map the textual boundaries of the question (and its multiple-choice answers) to on-screen coordinates, so that context menus can be customized for each question.

Semantic Document Infosets (SDIs)

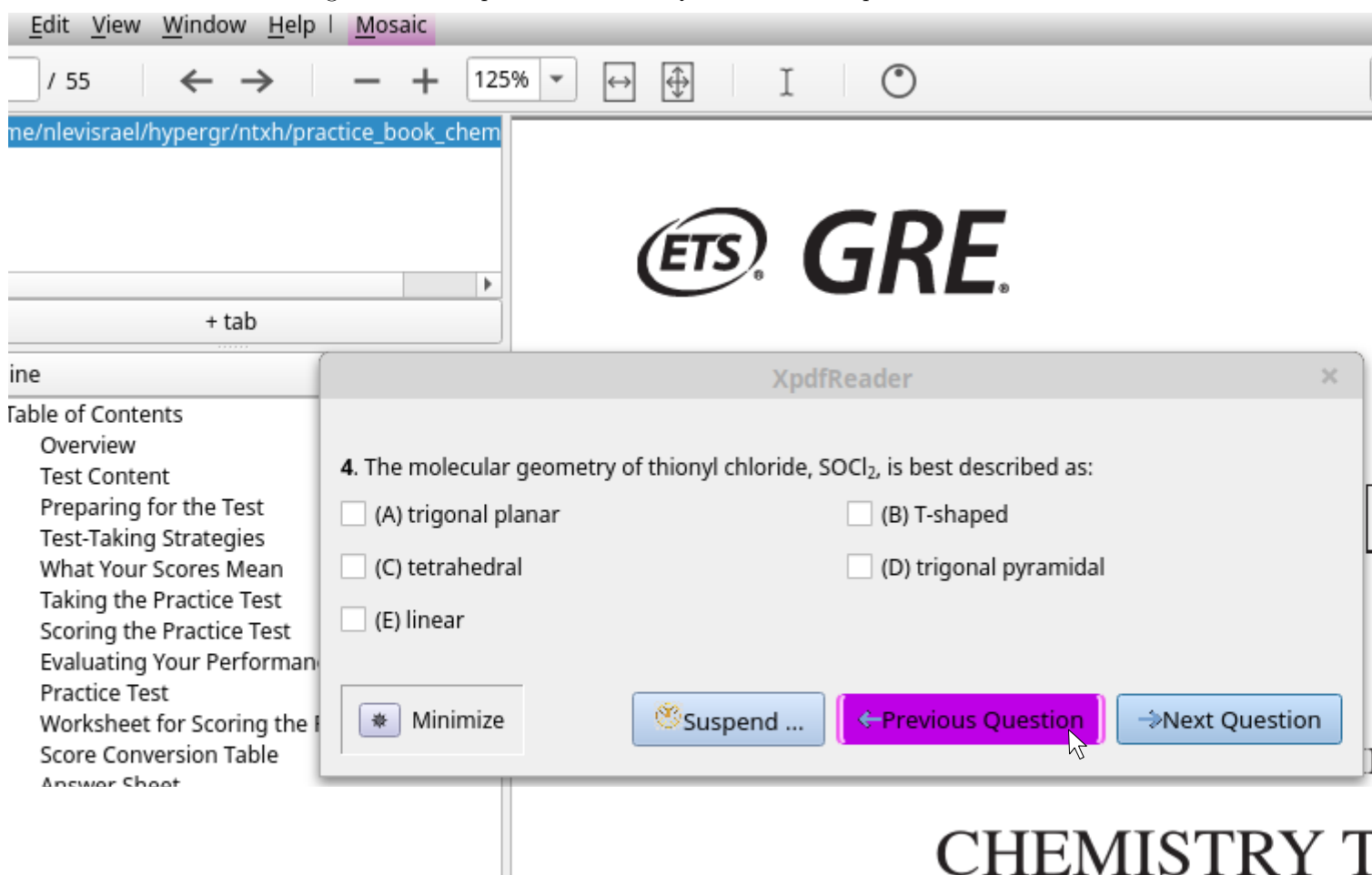
To support these capabilities, **ETSPF** would include tools to help compose publications (such as test-preparation materials) that embed what we term a "Semantic Document Infoset" (**SDI**), which effectively divides manuscripts into textual units (subsections, paragraphs, sentences, quotations, bullet lists, 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

Figure 6: Request Information in IQmol



may use to augment the underlying document with additional instructional and/or multimedia features — review questions, student instructions, glossaries, reading assignments, and so forth. The **SDI** can be used to guide **ETSPF** plugins when sharing data between applications — in Figure 1, for instance, selecting the Molecular Data file to send to IQmol based on the screen coordinates of the context menu — but also to enhance the presentation of content within the host application. For example, Figure 7 shows how an **ETSPF** plugin could provide an alternative interface for viewing practice-test questions, where readers can consider one question at a time, isolated in its own window, which may help them focus attention on each question in turn.

Figure 7: A Sample Practice-Test Question within XpdfReader



Using
L^AT_EX to
generate
SDI info-
sets

ETSPF implementations can include L^AT_EX 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 teaching content: review questions, glossaries, discussions of figure illustrations, etc. In documents where questions are printed as part of the publication text (for example, the ETS **GRE** practices), the L^AT_EX code can store the **PDF** coordinates for the questions 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, using the **SDI**, to sentences or paragraphs in those publications which are relevant to the review question that the student is currently reading/studying.

HTXN
(Hyper-
graph
Text
Encoding
Protocol)
Specifica-
tions

As an additional feature, **ETSPF** plugins would implement a protocol which we call **HTXN** (for "Hypergraph Text Encoding"). The goal of **HTXN** is to enable a new generation of publishing technologies which aspire to support multimedia reader experiences. In so doing, the traditional manuscript — the "primary" resource which is cited and downloaded — would then be networked with a package of supplemental (or "secondary") resources. However, at present, even when documents have supplemental files, it can be very difficult to transition from the primary to the secondary resource. To address this problem, **HTXN** is designed to rigorously document these multimedia networks, enabling e-readers and domain-specific applications to be integrated so that users may easily access multimedia content. 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 L^AT_EX and **XML**, for instance).

LTS can provide a demo with a more detailed overview of **ETSPF**, additional use-cases, technical information about plugin code, and sample **HTXN**-encoded documents.

