An ETS Plugin Framework for Enhanced Test Preparation

We are proposing a novel ETS Plugin Framework (**ETSPF**) whose purpose is to enhance document viewers in tandem with scientific and multi-media applications. In so doing, this plugin framework would allow document viewers to launch and share data with a diverse array of applications created for both scientific and social science disciplines, such as chemistry, physics, biology, medicine, linguistics, and sociology. Document viewers would therefore be able support an interactive, multimedia reading 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: e.g., review questions or assignment instructions.

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 (chemistry, for example) 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, that students would be able to access multimedia content linked to the documents (e.g., test-preparation materials) that they are currently studying.

How ETSpf enables multi-application networking and inter- operability

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 multiple-choice questions such as (on page 11, number 4), "**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 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 $SOCl_2$; the **XPDF** plugin would launch IQmol and send along a data package identifying this $SOCl_2$ file to IQmol's ETS plugin, with instructions to load the file into an IQmol session (see Figure 2). 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 an interactive **3D** graphic representing thionyl chloride. (Of course, analogous func-

Figure 1: A Thionyl Chloride Question in XpdfReader

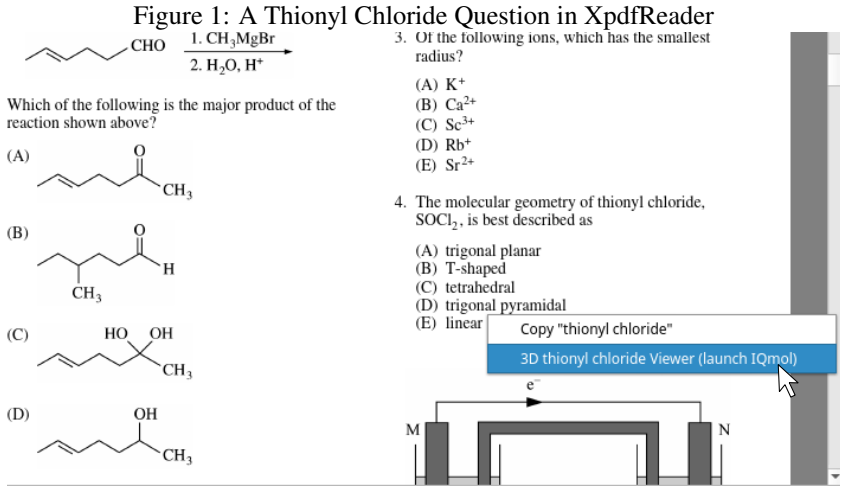
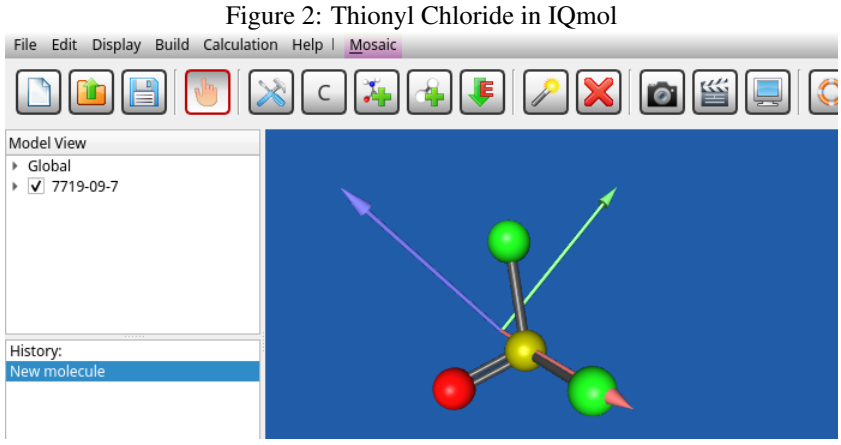


Figure 2: Thionyl Chloride in IQmol



tionality 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 data sent between plugins may sometimes be more extensive than a simple 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 would be equipped with the ability to send information not only about the present request but about the student's prior usage; in particular the fact that they 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 that the student can then browse back to their prior application-states (see the Model View panel on Figure 5) when needed.

Figure 3: A Lactose Question in XpdfReader

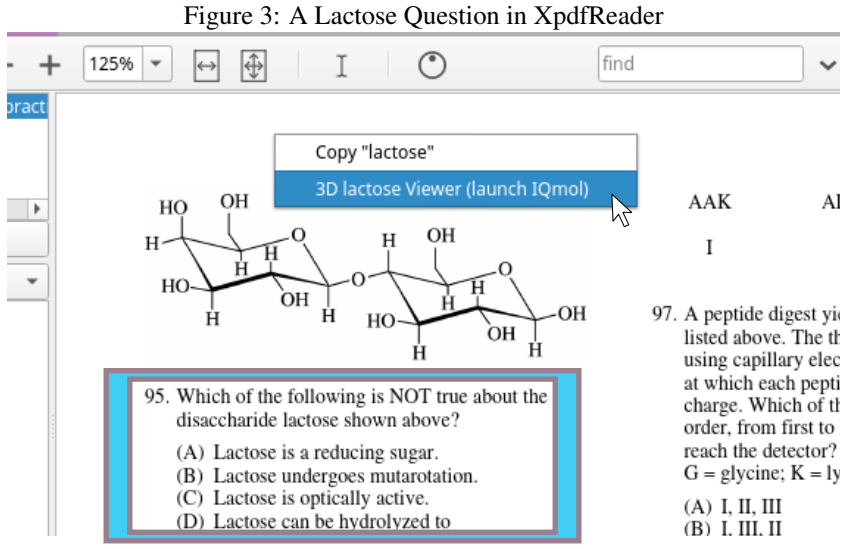


Figure 4: Plugin Info in XpdfReader

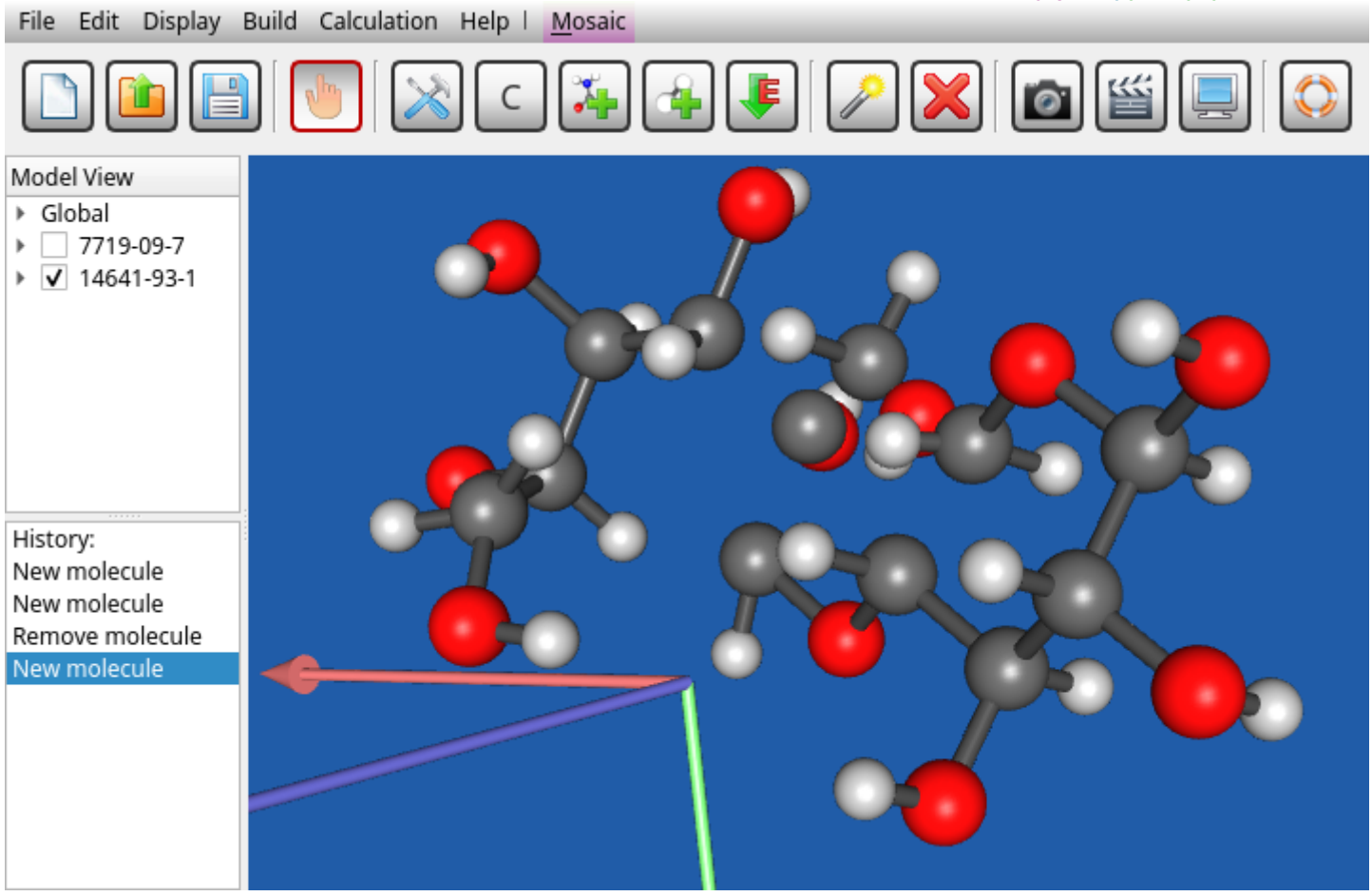
In general, the functionality provided by each ETS plugin would depend in part on the domain of the host application in which the plugin is embedded. For example, an IQmol plugin would load cheminformatic files and may activate IQmol's analytic capabilities in the domain of chemistry, whereas a plugin for applications in the domain of quantitative/statistical analysis and data visualization, such as ParaView, would load quantitative data sets (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 a dialog window to show basic plugin information (see Figure 4) as well as a more detailed review of data 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).

ETSpf Tools for Composing Test-Preparation Materials

ETSpf data in embedded files

In most cases, ETSPF 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 separate file embedded in PDF or EPUB documents, or (in HTML) inserted as non-display contents. When a document is opened, the ETSPF 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

Figure 5: Lactose in IQmol

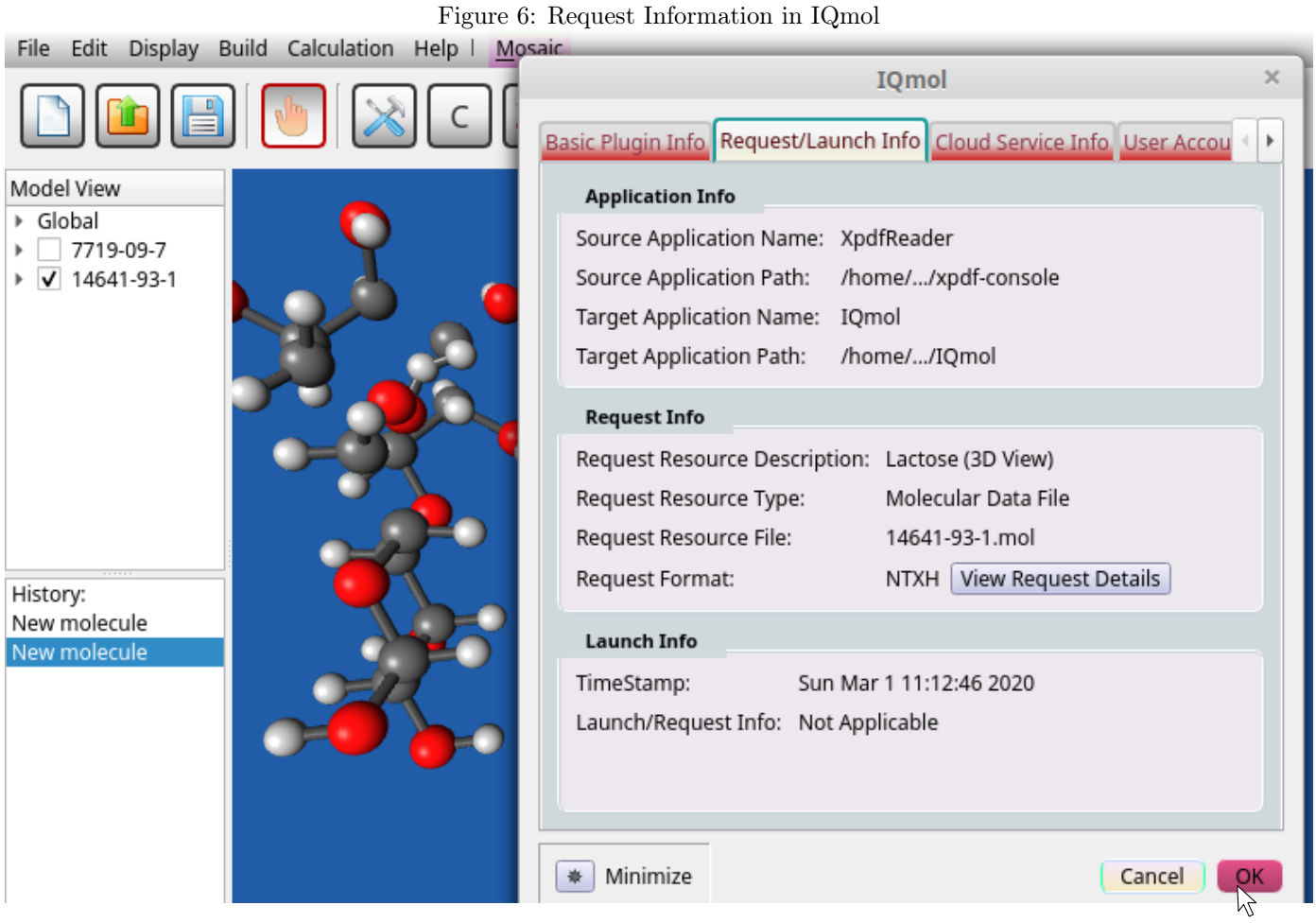


on-screen coordinates, so that context menus can be customized for each question.

Semantic Document Infosets (SDIs)

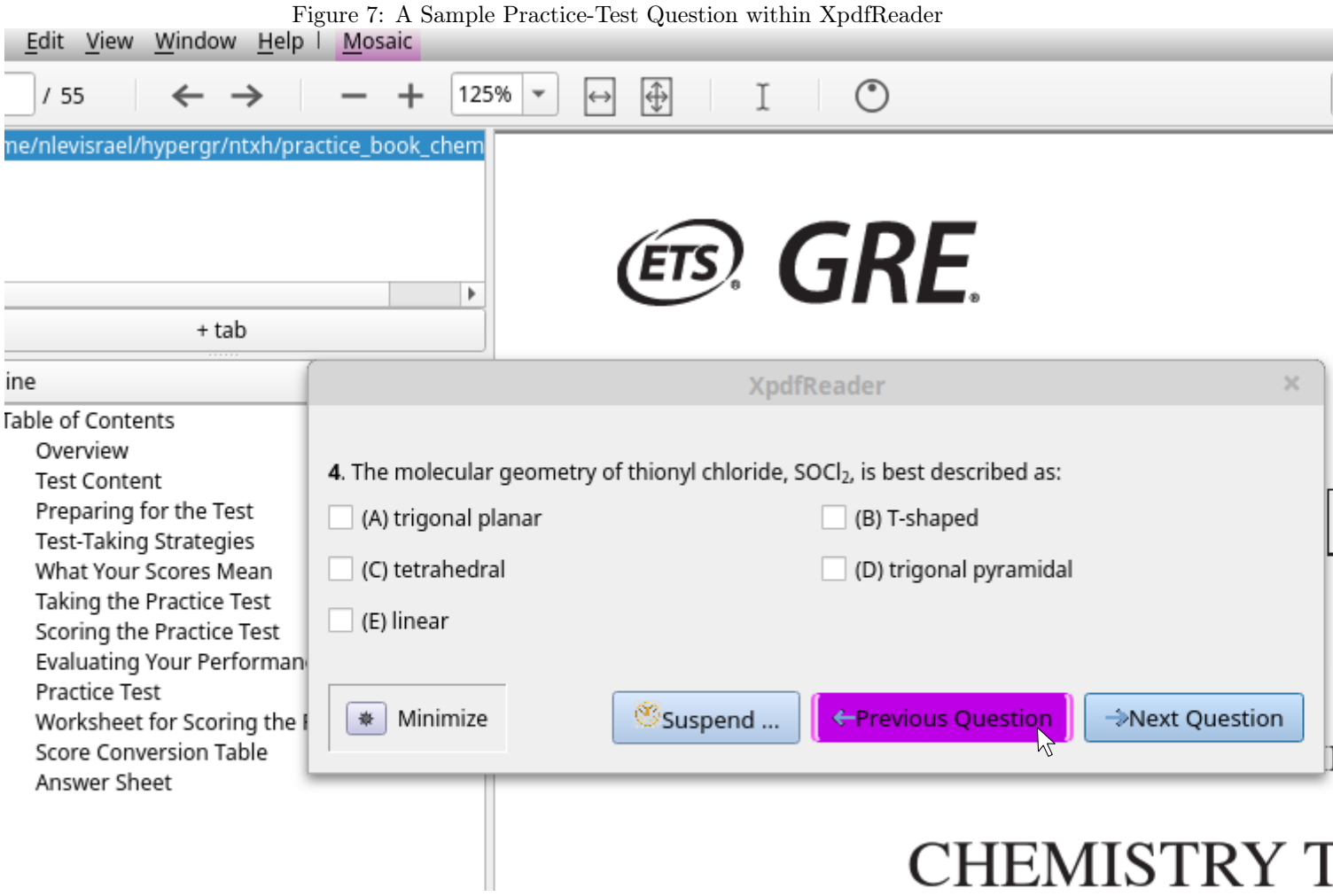
To support these capabilities, **ETSPF** would include tools to help compose publications (such as test-preparation materials) that embed what we term a "Semantic Document Infoset" (**SDI**), which effectively divides manuscripts into textual units (subsections, paragraphs, sentences, bullet lists, etc.)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

Figure 6: Request Information in IQmol



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 each question in turn, isolated in its own window.

Figure 7: A Sample Practice-Test Question within XpdfReader



Using LaTeX to generate SDI info-sets

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 teaching content: review questions, glossaries, class discussion suggestions, etc. In documents where questions are printed as part of the publication text (for example, the ETS **GRE** practices), the LaTeX 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 to sentences or paragraphs in those publications which are relevant to the review question that the student is currently reading.

***HTXN** (Hypergraph Text Encoding Protocol) Specifications*

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 LaTeX and **XML**, for instance). In the context of **ETSPF**, however, **HTXN** would be used to encode document information and text within the Semantic Document Infoset.

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.