Journal of Digital Information, Vol 2, No 4 (2002)

# Virtual Telescopes in Education

**Susan Hoban, Marie desJardins, Nora Farrell, Priyang Rathod, Joel Sachs, Suryakant Sansare, Yelena Yesha, John Keating*, Bart Busschots*, Johanna Means*, Gilbert Clark**, Louis Mayo*** and Willard Smith+**
University of Maryland Baltimore County, GEST Center,
1450 South Rolling Road, Tech Center 3.002, Baltimore, MD 21227, USA
Emails: hoban@umbc.edu, mariedj@csee.umbc.edu, nfarre1@gl.umbc.edu, prathod1@csee.umbc.edu, jsachs@csee.umbc.edu, ssansa1@csee.umbc.edu, yeyesha@csee.umbc.edu
*Department of Computer Science, National University of Ireland, Maynooth, Co. Kildare, Ireland
Emails: john.keating@may.ie, bart@cs.may.ie, jmeans@cs.may.ie
**President, Telescopes In Education Foundation, 1160 E. Calaveras St., Altadena, CA 91001, USA
Email: gilclark@earthlink.net
***Raytheon ITSS, Code 630, Goddard Space Flight Center, Greenbelt, MD 20771, USA
Email: lmayo@pop600.gsfc.nasa.gov
+Center of Excellence in Information Systems, Tennessee State University, 330 10th Avenue N., Nashville, TN 37203-3401, USA
Email: smith@coe.tsuniv.edu

## Abstract

Virtual Telescopes in Education is providing the services required to operate a virtual observatory comprising distributed telescopes, including an interactive, constraint-based scheduling service, data and resource archive, proposal preparation and review environment, and a VTIE Journal. A major goal of VTIE is to elicit from learners questions about the nature of celestial objects and the physical processes that give rise to the spectacular imagery that catches their imaginations. Generation of constrained science questions will assist learners in the science process. To achieve interoperability with other NSDL resources, our approach follows the Open Archives Initiative and the W3C Semantic Web activity.

## 1 Introduction

The NSDL-funded project Virtual Telescopes in Education (VTIE) has its genesis in an extant project, Telescopes in Education, which began in 1992. TIE provides remote access to telescopes, allowing students to point telescopes and take images of celestial objects from their classrooms. The first telescope to participate in TIE was the 24-inch telescope at the Mount Wilson Observatory, coming online in 1993. Since that time, students from hundreds of schools in the USA, Australia, Canada, England and Japan have remotely controlled the telescope. To help teachers and students understand what they are doing and how to use the software that points the telescope, the TIE project and members of the growing TIE community have developed a suite of guidebooks, sample projects and lesson plans (Clark 2002). TIE began (and continues) with NASA funding. Recently, the TIE project spun off from NASA/JPL forming the non-profit TIE Foundation.

TIE is experiencing tremendous growth. Over the past few years, more than 20 observatories have been outfitted with TIE standard hardware and software interfaces. Currently, these telescopes operate as independent observatories with little leverage of resources, communication or coordination. This paper outlines Virtual Telescopes in Education (VTIE), a project that will integrate the TIE affiliates into one virtual observatory, through a Web-based interface that will provide automated scheduling of the telescopes for students and teachers, and access to a library of data and resources for lifelong learners.

### 1.1 VTIE Philosophy and Approach

VTIE has educational, science and information systems goals. From the perspective of education, we design the system with the learner as the primary user. An objective of the VTIE system is to enable learners to participate as scientists. This overlaps with our science goal: to present science as a process. VTIE seeks to engage learners by eliciting questions about astronomy, allowing them to make observations of celestial sources to address their questions, and to facilitate the reporting of their results. Finally, VTIE is committed to maximizing ease of use and interoperability. The interface and all associated tools will be Web-based. To achieve interoperability with other NSDL resources, our approach follows the Open Archives Initiative and the W3C Semantic Web activity.

This paper describes the philosophy and resulting technical approach to the development of VTIE. We will speak of VTIE when we are referring to the parts of the system developed under the NSDL activity, and TIE when we are referring to the activities managed by the TIE Foundation. Also, we will speak of students and learners, as well as of teachers and facilitators. TIE currently works primarily with schools, thus certain (access restricted) functions will be specifically for teachers and their students. VTIE, following NSDL philosophy, seeks to promote lifelong learning outside the formal classroom setting, so we sometimes speak of functionality for learners and those who may be facilitating their learning. We have tried to use "students" vs "learners" where the activity is restricted to the formal classroom setting.

## 2 Components

The components of VTIE described in this paper are the customizable virtual observatory (MyVTIE), tools for interactive proposal and paper generation and review, the resources library, and the suite of tools for managing the telescope resources.

### 2.1 Customizable Virtual Observatory

#### 2.1.1 Design approach

VTIE's goal is to design and implement an interface that is simple and easy to use (see e.g. Nielsen 2000). We approach the design and implementation of the user interface employing the following method:

- Define the primary goal of the site
- Identify and research the target audience
- Create a comprehensive architectural design
- Develop a prototype and perform testing at every step
- Incrementally implement customization and personalization

The first step in the design of an effective user interface is to define the primary goal of the site and the intended audience. The primary goal of the VTIE site is to engage learners in the scientific process through astronomical observations. The intended audience is students, teachers and astronomy enthusiasts (including astronomy clubs, individuals, etc.).

Online learning differs from the traditional educational experience for both the teacher and student. The student has more freedom for self-directed learning while assuming the responsibility for taking an active part in the educational experience. Overall design, navigation, the use of frames, fonts, and even specific colors, may provoke varying responses among users, suggesting that the ability to comprehend a site is a function of the user's expectations and experiences (Barber and Badre 1998). We will use a design method that will consider non-cognitive processes that affect learning, such as emotional experiences that affect motivation, independent learning, and independent problem solving (Martinez 2001).

The most basic aspect of a usable site is simple and consistent navigation. The user interface architecture is modular in design. Each page is composed of four vertical panels or columns. Each of the four panels is populated by modular components, widgets, which span one to four panels. Figure 1 is a screenshot of the VTIE portal prototype. Panels 2 and 3 contain one widget, Astronomy Picture of the Day. The widgets are containers that dynamically display information using dynamic html, javascript, and cascading style sheets. A generic widget is designed and from that special cases of the widget are built. One widget can be substituted for another within the same interface. The use of predefined modules as containers for dynamic content separates presentation from content. It allows for the gradual implementation of customized content within the existing design.


Figure 1. VTIE Portal Design, four panels

Individual pages are consistent in design, font and color, and information is organized in a way understandable by the user while taking advantage of prior experience and expectations. A common example is the iconographic use of an envelope to indicate the functionality of sending an email to the Web site administrator. Usability testing will reveal the validity of our definitions and decisions (see e.g. Nielsen 2000; Spool *et al.* 1999).

The VTIE portal comprises three access corridors: Public, Students and Teachers. Most of the VTIE resources (e.g. data, VTIE Journal, tutorials) will be available to the general public. However, access to the tools for preparing an observing proposal, and writing/submitting a paper to the VTIE Journal, as well as for submitting a request to schedule observations, will require registration (see Figure 2). The user information will be stored securely in the VTIE database, and will be made available only to authorized persons based on relevant privacy policies. Users will be required to log on for each session. Teachers will be required to give authenticating information, including their email address and the name of their school. Teachers will register each student. Students will use aliases exclusively. Registration will avoid the use of persistent cookies and will allow the system to identify the unique individual using the machine at that moment in time, as at school and even in a home environment, multiple users share a single computer. User preferences will be stored in a database, providing dynamic creation of their MyVTIE page upon each visit to the site.

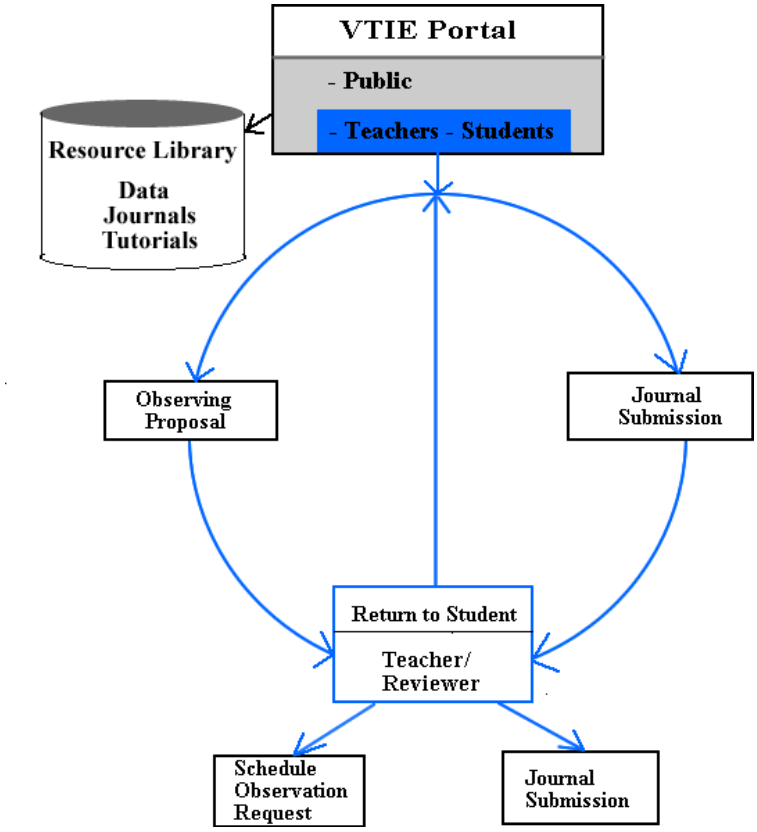


Figure 2. VTIE Portal Functionality

### 2.1.2 Customization and Personalization: MyVTIE

Customization is defined as active or user-directed design of the content or display format of a Web site. Personalization is passive: the software tracks and records the user's movements within a site and creates a profile that presents the user with pages that reflect the most-often viewed items (Nielsen 1998). The practice of customization and personalization of Web sites has generated highly polarized commentary among well-known HCI experts and users alike. Businesses are spending increasingly more money to personalize their Web sites and to provide customization for users to keep up with their competitors, often without researching precisely how or what to personalize nor how, or if, it will benefit their users. Some statistics suggest that personalization, even on highly ranked sites, is underutilized, and that those sites are less visited than their non-personalized counterparts. NetRatings' top personalized site My.Yahoo.com had 5.3 million unique users in one month, yet 44.8 million unique visitors chose the generic Yahoo! instead. Similarly, the second top rated site, My.AOL.com, saw 2.5 million unique visitors in a month while 48 million visitors flocked to the plain vanilla AOL network (Eads 2000).

VTIE has developed an incremental, phased customization plan. Each phase will be added based on research, testing and analysis of benefit to the user. The initial implementation will provide customization in the students' corridor only. Based on lessons learned during this process, the customization capability will be added to the teachers' corridor in the second release.

We plan customization options for content areas only. Although options to change font color and size, background colors and other similar features are well-liked by users, they do not add substantive value. Users may be tempted to tweak their superficial settings to the possible exclusion of content.

In the students' corridor, entrance points to three default content areas will be available: VTIE Resource Library, Proposal Generation, and Paper Writing. Students will have the option to customize in the following areas:

- *My proposal status.* This will be a module that graphically represents the current state of a learner's proposal, including date submitted, date under review, date returned for revisions, and telescope time assignment.
- *My journal papers.* Similar to the proposal status, a module will display the current state of a learner's submission to the VTIE Journal.
- *My reading list*. Learners will be able to add their own bookmarks to a personalized bibliography with the ability to annotate the entries. Learners will be empowered as contributors to the virtual observatory, adding value to the entire network of users by allowing their bookmarks to be viewable publicly.
- *VTIE Messaging.* The prototype will use email; we may move to a discussion forum in the second release.
- *See the telescopes.* Students can choose to view photos and descriptions of the telescopes in the TIE network. The capabilities and instrumentation for each telescope will be included.
- Interesting online astronomy components, e.g. Astronomy Picture of the Day, retrievable from the popular NASA site on a daily basis. The photo of the day is current data, relevant, and aesthetically pleasing.

Allowing customization of every feature can result in information overload, causing some users to retain their default settings. The first customization option implemented will be *See the telescopes*. Astronomy Picture of the Day will also be implemented in the first prototype. The remaining elements that will be customizable will be based on how the first two options are being used.

In the teachers' corridor, the default content areas are: VTIE Resource Library, Proposal Review, Paper Review and Scheduling Tools.

## 2.2 Interactive guidance for scientific inquiry

As exciting as it is to point a 24-inch telescope and obtain a beautiful image with a 30 second exposure, such activities

constitute only one component of VTIE's main goal. In fact, our goal is to move students beyond a star party and into scientific investigation. Thus, just as in the professional world, a request for time on a telescope must come in the form of a proposal, which poses the scientific question that the learner seeks to answer and outlines the observations to be made to address some aspect of that question. In some cases, students will come to VTIE having already conceived of scientific questions and associated observation requests. In many cases, however, they will not. To encourage the learners, we do not wish to say: "Come back when you have a good question", nor do we wish to simply provide them with a list of pre-designed questions/observations (the traditional science lab approach). Instead, VTIE guides learners through the process of asking a question, providing a dynamic view of available information based on their choices along the way. Figure 3 outlines the steps involved in the end-to-end process of inquiry that VTIE strives to facilitate.

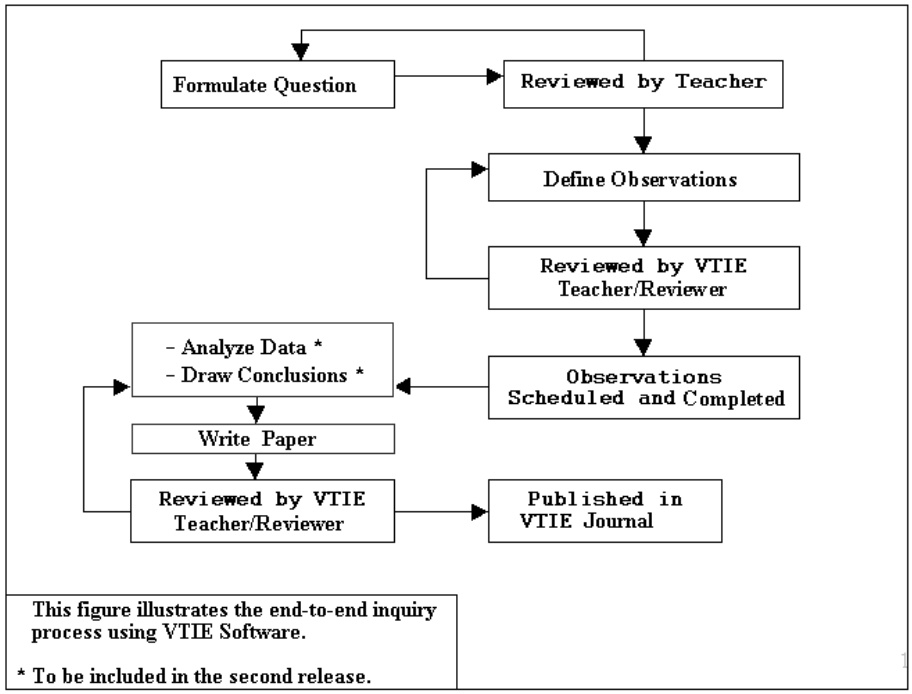


Figure 3. End-to-end VTIE Inquiry Process

### 2.2.1 Question Provoking Tutorials

Upon indication of interest in a particular content area, the learner will be presented with a collection of micro-tutorials. Ideally, these tutorials will not only stimulate an interest in a particular area, but will provoke questions which can be investigated with VTIE observations. An effective way to create such tutorials is to begin with an existing online exposition on a particular topic, and then add hyperlinks to encyclopedia entries of all astronomical terms that appear in the exposition. Clicking on a term brings up a pop-up window with the appropriate encyclopedia entry. Initially, we are testing using the *Encyclopædia Britannica*. Ultimately, however, we plan to use non-commercial, online resources.

For the first VTIE release, we will use the project descriptions/lesson plans developed by the original TIE program for the content expositions. When a user requests a project description, the description is scanned for terms that appear as entries in the underlying encyclopedia. When a match is found, an appropriate link is added on the fly to the page that is about to be served to the user. The tutorials are not retained (although we will employ caching heuristics to improve performance). In addition to obviating the need to add links manually to each page to which we are adding value, our automation mechanism also enables linking to different content for different users. That is, our aim is to have not one underlying encyclopedia, but several. Each will address a different audience: primary school, high school, etc. Thus the same underlying exposition can serve as a tutorial for a variety of diverse audiences.

### 2.2.2 Interactive proposal generation

#### *2.2.2.1 Formulate a Question*

After learners (Note1) have explored ideas through the tutorials, they will be expected to formulate a question about the object of interest such as "What is the temperature of Polaris?" While the possible questions are numerous, the choices are constrained by the types of measurements that can be made with the instrumentation available at TIE sites (see *Define Observation* below). Two strategies for preventing learners from asking unanswerable questions are evident. On the one hand, the system could warn learners to be sure to pose only questions that can be addressed by the available observational capabilities. If they ask a question that is impossible to answer, the system would then indicate that they should try again and possibly give hints to improve the question. However, the set of unanswerable questions is infinite. Additionally, this approach would involve analyses of proposals by the system with a thorough knowledge of the instrumentation and capabilities provided by each of the telescopes. An automated system would require substantial natural language processing, and is still likely to be imperfect. On the other hand, an interface could guide learners to ask an answerable question and to design an observation that addresses the question that has been posed (Note 2). VTIE has adopted this latter approach in the development of the *Proposal Generation Interface* (PGI).

After the learners have formulated a question about the object of interest, they will use the PGI to state their proposal, such as "I want to determine the temperature of Polaris". The interface will guide the construction of the sentence with choices of inputs. Our approach is to present the learner with a dynamically changing, pre-formatted menu system, based on decision trees, that will only ever provide valid options to the user. The preformatted sentence will include blanks that can be filled-in from drop-down menus. As the user enters the first field, the options in all the other fields will adapt dynamically so that only valid questions can be posed. (A similar construction is used for defining observations as shown in Figure 4.) In our example, the learner would be presented with a sentence of the form:

I want to <analyze> the <physical quantity> of <celestial object>

and would choose: analyze = "determine", physical quantity = "temperature", and celestial object = "Polaris". An example where the later choices would adapt might be: analyze = "compare" that the <celestial object> option would expand into <celestial object1> with <celestial object2>.

For younger learners, the questions and proposals will be simpler than for high school or undergraduate users. The system is designed at two levels initially, targeting, primary school and high school reading levels. The levels are reviewed for style and vocabulary by educators

The PGI provides for the inclusion of a paragraph explaining the proposal and the learner's motivations. In this paragraph, the learner provides an hypothesis that observations can address. This hypothesis is essential to indicate understanding of the statement constructed using the software and to prevent random clicking.

#### *2.2.2.2 Define Observation*

As stated previously, there are a limited number of questions that the system can answer, since only certain measurements can be made using the existing instrumentation. Namely, observable quantities are currently limited to angular size, shape and flux (surface brightness for resolved sources) through a limited set of filters (Note 3).

As with the *Formulate a Question* function, *Define Observations* generates potential observations from extensive predefined relationships that are guaranteed to be valid given TIE resources. A diagram showing the menu-approach for constructing proposed observations is given in Figure 4. To engage the learners in the process, beyond simply clicking on choices, the hypothesis and motivation contributions are sought. The hypotheses should relate to the observations and demonstrate that the learner understands the experiment. Proposals should be returned for re-working if the proposed observations do not address the hypothesis (Note 4).

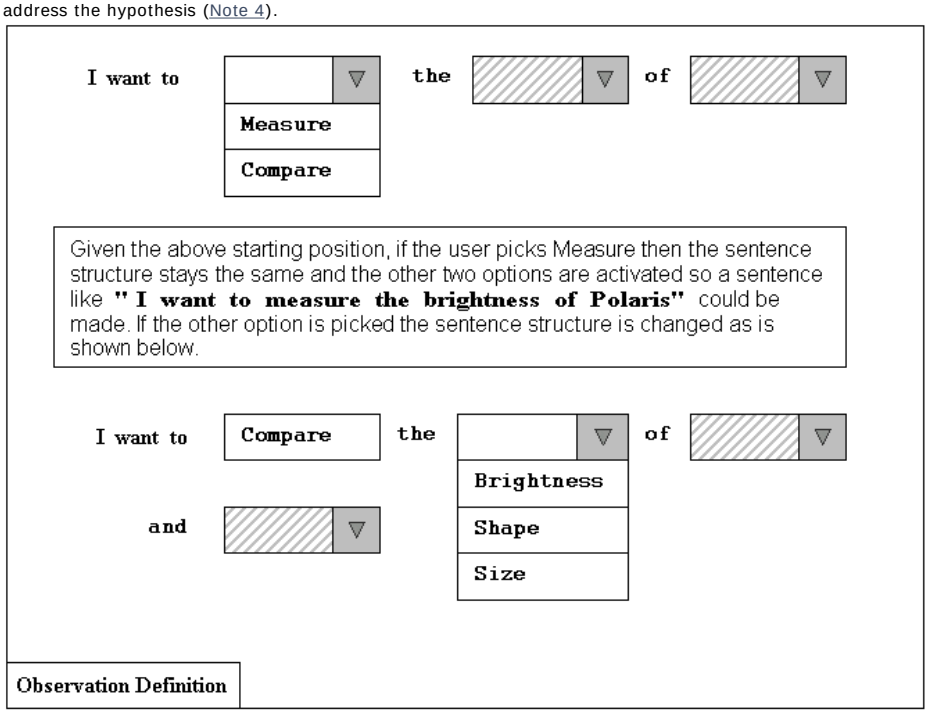


Figure 4. Operation of *Define Observation* function in the VTIE *Proposal Generation Interface*

When all component sections have been completed, the learner submits the final proposal. Following the process in professional astronomy, the proposal must be reviewed (most likely by the teacher in TIE, or some combination of the teacher and the student's peers). Once the reviewer is satisfied with the soundness of the proposal, a Proposal ID will be allocated, and the teacher may submit a request for an observation using the Interactive Schedule Picker and Viewer tools. Authorized users, in most cases this means teachers only, will have permission to resquest an observation. Teachers are encouraged to conduct this step in the presence of the learners whenever possible to encourage participation in the process. Once the observations are scheduled, the learners will remotely control the telescope and operate the CCD camera to obtain the images. This step is outside VTIE: the TIE affiliates are controlled by software developed by Software Bisque, Inc.

## 2.3 Reporting Results: Writing a VTIE Paper

Once the learners have their images, they will analyze the data, draw conclusions and compose a paper based on their findings. They will compose the paper online using the *Paper Generation Tool* (Note 5). Upon successful review, these papers will become part of the online VTIE Journal.

Learners, now authors, will be presented with a dynamically-generated HTML form which contains all the elements of a typical paper, for example Title, Abstract, Keywords, Paper Body (Introduction, Observations, Analysis, Conclusions), Figures, References, etc. The Title and Abstract and several of the keywords will be automatically extracted from the successful Observing Proposal that lead to the results being reported. Authors will be provided with context-sensitive DHTML menus to assist with text formatting, spell-checking and image insertion. The general layout for the paper will be provided, thus focusing the authors on writing rather than presentation. Furthermore, when the author moves to a new element on the page (for example, keyword) they will be presented with context-sensitive functional explanations for that particular element ("Keywords aid indexing."). *Save*, *Revert* and *Submit* options will be provided. These options allow authors to work incrementally on a paper and revert to earlier versions using the version control system automatically provided. The *Submit* option invokes the *Paper Submission Tool*, which 1) sends the paper to a reviewer (in TIE, usually the teacher or a combination of the teacher and the author's peers) and 2) stores the components of the paper, each separately tagged as XML. For the prototype system, we will convert the HTML form-based (CGI) submissions to XML and save the XML to a database. The XML objects will be used to dynamically generate the final product, manage augmented submissions (documents under review, version control), and provide a rich searching tool for research purposes (e.g. searching the

database for all observations of Polaris). Figure 5 shows an overview of the process of generating a paper, storing the individual components, and presenting a complete PDF version for the VTIE Journal.

All VTIE Journal papers will be viewable in PDF format. We will generate an HTML version for viewing on the screen; however, PDF provides a better format for printing, and the learners may wish to have a hard copy on which they can write their names (recall, online, the student's names may not be displayed). Each paper will be automatically generated on request from XML components as stored by the Paper Submission Tool. For the prototype system, we will generate a LaTeX source file from the XML objects, and using a custom Style file generate a Postscript file from which a PDF document will be automatically generated. It is expected that later in the project we will produce tools to produce PDF directly from the XML objects.

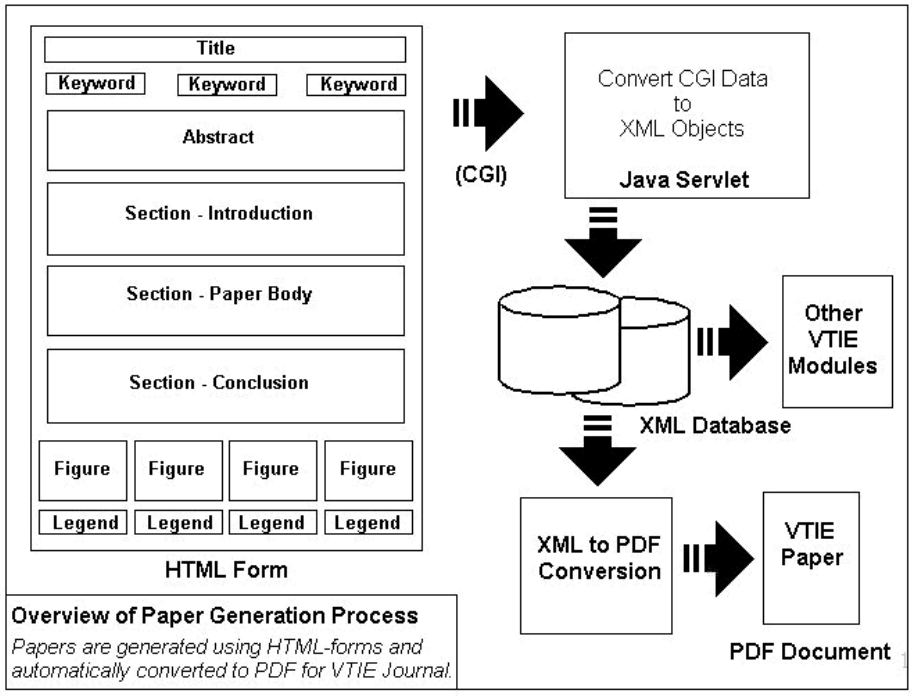


Figure 5. Overview of the VTIE Paper Generation Process

## 2.4 Review: Proposals and Papers

In the *Reviewing Tool*, VTIE seeks a simple and effective method for communication between authors and reviewers. The paper will be dynamically generated for the reviewer from the XML object database together with a template to aid the reviewing process. Reviewers will have the ability to attach virtual-notes to the document at various points with comments about that section. DHTML menus will be provided to add, edit, or remove virtual notes to the document. Reviewers will also be provided with the same *Save*, *Revert* and *Submit* options mentioned earlier as, in general, the process may be spread over many sessions. The *Submit* option adds to the database the reviewer's comments as XML objects associated with the paper. The author will be informed (via email at first, potentially migrating to a discussion forum) that the comments are available for inspection. The comments will be automatically inserted as visible, movable DHTML Layers overlaying the paper, which is returned to the author as an HTML-encoded form, following the format used in the *Paper Generation Tool*. The author must address every comment by completing a text-input box attached to every comment. Each comment will have a check-box which the author may click to indicate that the comment has been addressed, and once checked, the comment will disappear from view. Paper writers may not just click on the comment to close it, as built in intelligence (implemented in JavaScript) will determine if the user has indeed changed the text or replied to the comment prior to being provided with the *Close Comment* check-box. All comments may be made visible at any time using a menu item elsewhere on the document.

When all comments have been addressed, the author may re-submit the paper. As before, the paper will be converted to XML objects and a report containing the changes and comments will be automatically generated for the reviewer. If the reviewer is satisfied with the revised paper, the paper will be marked for publication. The author will be informed that their submission has been successful and a link to the paper in the VTIE Journal, which resides in the Resources Library, will be provided.

## 2.5 Resources Library

### 2.5.1 Storage Mechanism

VTIE seeks to blur the distinction between data and metadata, and XML storage is a component of our attempt to accomplish this. A debate rages in the database community. The point at issue is how to deal with XML - through building native XML stores (Jaggadish 2002), or through accommodation of XML data structures in an object-relational setting (Stonebraker 2001). Many of the reasons for rejecting an RDBMS approach do not apply to VTIE, as they involve overhead costs (for tree-table mapping, table joins, etc.) by which VTIE, being small, is unlikely to be plagued. Nevertheless, we plan on working with native XML storage, because relational, and even object-relational, metadata schemas may be too rigid. The self-defining nature of XML, on the other hand, permits the storage of documents constructed from elements that have not been encountered previously (Hawkins 2001).

Another reason to avoid an RDMS is the experimental spirit of NSDL (Hanson 2001). NSDL projects are encouraged to explore new technologies. Typically, hacks are built around an RDBMS solution to deal with data that is not relational (i.e. tabular) in nature. We plan to build hacks around our XML storage engine to deal with data that is not inherently hierarchical. We will deploy an RDBMS when necessary, but we hope that our efforts will help extend understanding of the capabilities and limitations of XML storage.

### 2.5.2 Resources

In VTIE, as in RDF (see Lassila and Swick 1999), everything is a resource; VTIE resources are created during every stage of the end-to-end VTIE process. Examples of resources include proposals, images, VTIE Journal papers, as well as the components of each of these, and eventually tools and services. As each resource comes into existence, its components are encapsulated in XML and are further annotated with appropriate RDF metadata, to permit queries such as the following to be answered:

- Retrieve proposals where the scientific justification is cosmological in nature
- Who is looking for supernovae?
- What exposure times have been used for imaging the Crab Nebula?
- What extra-galactic observations have been made from the Mt. Wilson telescope?
- Which telescopes are being used to search for comets?
- Which VTIE Journal papers resulted from superposition of multiple exposures?

We expect that querying existing proposals, images and VTIE Journal papers will be of use to learners as they compose their own observation proposals. We are designing the structure of VTIE resources with the needs of our query agents in mind. Since we are conforming to the standards of the emerging semantic Web, it is our hope that VTIE resources will be usable not only within VTIE, but by other NSDL and Web applications.

### 2.5.3 Query Agents

A traditional digital library draws a sharp distinction between data and metadata. For example, a database record typically will consist of metadata descriptors for a document and a pointer to the document. The most conservative vision of the semantic Web replicates this data/metadata distinction by wrapping a Web resource in RDF statements that describe the resource. The most aggressive vision of the semantic Web involves expressing every nuance of a document in DAML+OIL (see e.g. Horrocks *et al*. 2001), so that its semantics are completely intelligible to a software agent (Note 6).

VTIE is pursuing a fairly conservative approach, but with the flexibility to become more aggressive in particular situations. The point of our semantic markup is to extend the nature of queries that can be posed. We want learners to be able to query on concepts, rather than keywords. When a learner issues a query, the VTIE query agent expands the query according to relationships expressed in the back-end ontology. For example, the ontology might specify that planets are a subclass of non-sidereal astronomical objects, and that Jupiter is an instance of the class *planet*. The query "Show me all active proposals to observe non-sidereal objects" would therefore be expanded to a conjunction of queries that includes "Show me all proposals to observe Jupiter". All queries in the conjunction would be issued to the XML database.

In summary, the VTIE Resources Library provides an interface to a collection of ontologies and query agents that together make relevant resources retrievable.

### 2.5.4 Broader Integration into the Semantic Web

One of the main purposes of the semantic Web is to enable an agent-mediated approach to interoperability. The semantic Web is likely to be built from the ground up. That is, it is not likely to be the case that everyone will describe all of their resources in RDF, instantly achieving global interoperability. Rather, opportunities for interoperability will be need to be discovered. People working on diverse projects will identify how components of their systems can profitably interact. The resulting semantic markup will likely enable interaction with components of other systems not yet considered. One purpose of NSDL is to bring diverse projects together to take these first steps towards automatic interoperability. Below, we list two projects, one within NSDL, and one without, that we are working with to demonstrate the power of the semantic Web concept:

1. Syracuse University's *EduRef* (Lankes 2002), an NSDL service, enables learners to post questions on any subject, and then farms the questions out to experts, who provide a response. We will add value to the "ask an astronomer" questions in the following way. After an expert has finished composing her response in an HTML text box, she clicks submit. Her response is then put into the same hyperlink induction system described in section 2.2.1. Rather than generating links to encyclopedia entries, however, we will automatically annotate the expert's response with links to VTIE resources - proposals, student journal papers, images, etc. The annotated response will be presented to the expert, who will have an opportunity to edit it before it is returned to the student. The automatic annotation will not only add to the expert's response, but will also suggest possible uses of VTIE to the student who asked the initial question.
2. As part of the DARPA Agent Markup Language (DAML) program, researchers at the University of Maryland Baltimore County and the Johns Hopkins University are working together to build a search engine capable of indexing and drawing inference on RDF triples (Shah *et al*. 2002). We see such a system as essential to the scalability of VTIE. As the resource library grows, it will be harvested and indexed to provide a timely response to user queries. Ultimately, when commercial search engines become fluent in RDF, VTIE resources will become part of the searchable/retrievable Web, and will be findable even by those who are not aware of VTIE's existence.

## 2.6 Telescope Resource Management

### 2.6.1 Physical Layout

VTIE is a distributed system. There will be a large server at the National University of Ireland Maynooth (NUIM), (henceforth, the VTIE server), and each observatory will have its own smaller server. If an observatory has more than one telescope, each telescope may or may not be running its own server. The only communication the users will have with the system will be through the Web using the HTTP protocol. In our initial release, Java RMI will be used to achieve communication among distributed VTIE components. In future releases, we plan to use SOAP. Our expectation is that SOAP will permit a more loosely coupled system, and will also allow interoperability between VTIE and non-VTIE subsystems No peer-to-peer communication will exist between the software on the various telescope servers or between the software on the various observatory servers. The servers will each be running a number of software components. The majority of these will be Java based.

The VTIE server will host the Web interface for the learners and teachers as well as the scheduler and associated tools. The VTIE server will also be running a Census Agent to collect information from all the telescope servers and to format it for use by the scheduler. There will also be other software elements to facilitate communication using Java RMI described above, namely a remote interface.

The telescope (or observatory) servers will store information about that specific telescope, its capabilities and instrumentation, and will also store that telescope's schedule. Again, there will be a need for a remote interface for Java RMI communication. This layout is shown in Figure 6.

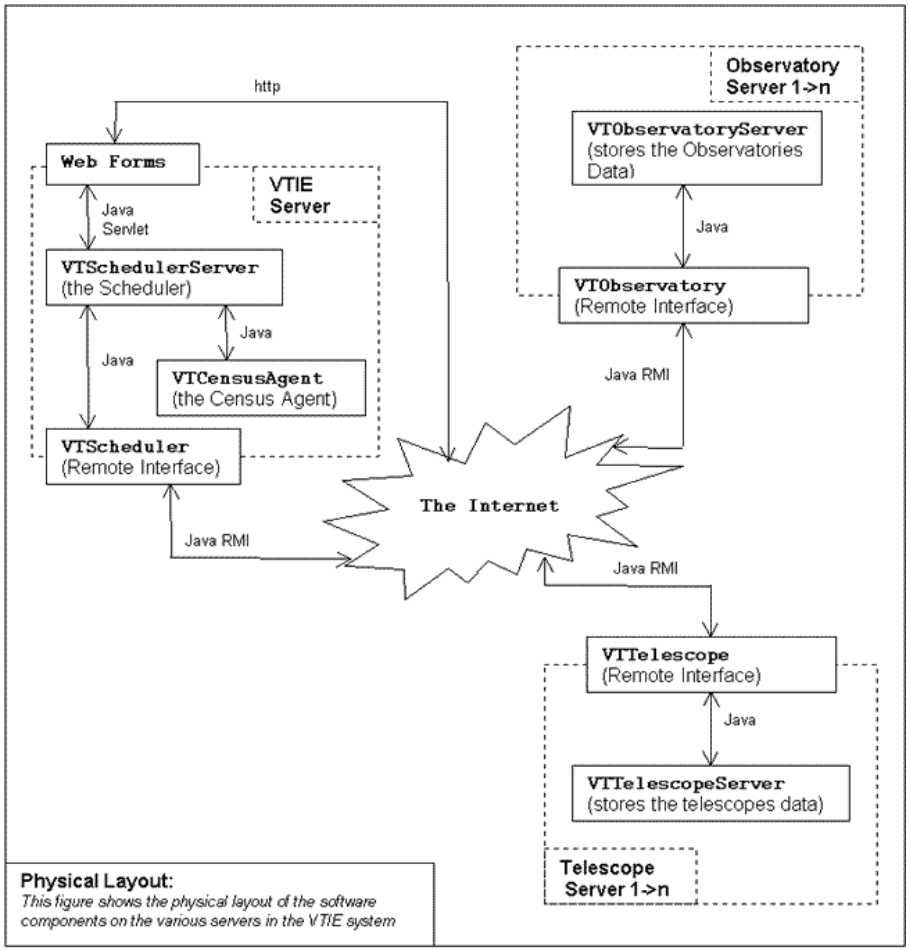


Figure 6. VTIE Physical Layout

### 2.6.2 Data Flow

The requests for an observation will originate from the Authorized Users (typically teachers) who will submit them using the Interactive Schedule Picker & Viewer. A Java Servlet is used to validate the data and then pass it on to the scheduler (VTSchedulerServer) if it is properly formed, or back to the user if it is not.

The scheduler (VTSchedulerServer) will collect all these observation requests (VTObservation) and store them to be scheduled later. The scheduling will be done in batch (period to be determined empirically).

When the scheduler is invoked it will request information on all the telescopes from the Census Agent (VTCensusAgent) which will then ask each telescope (VTTelescopeServer) for the information it contains. Each telescope will then ask its observatory (VTObservatoryServer) for the information it contains and return all this to the census agent (VTCensusAgent), which will format it correctly before returning it to the scheduler (VTSchedulerServer).

Once the scheduler (VTSchedulerServer) has all the information it requires it creates an optimized schedule. This schedule is then sent to each of the telescopes (VTTelescopeServer) where it is stored.

There will be a number of observations that are notcompleted at their allocated time due to weather conditions or technical failures. These observations (VTObservation) will be returned to the scheduler (VTSchedulerServer) by the telescope (VTTelescopeServer).

This process is depicted in Figure 7.

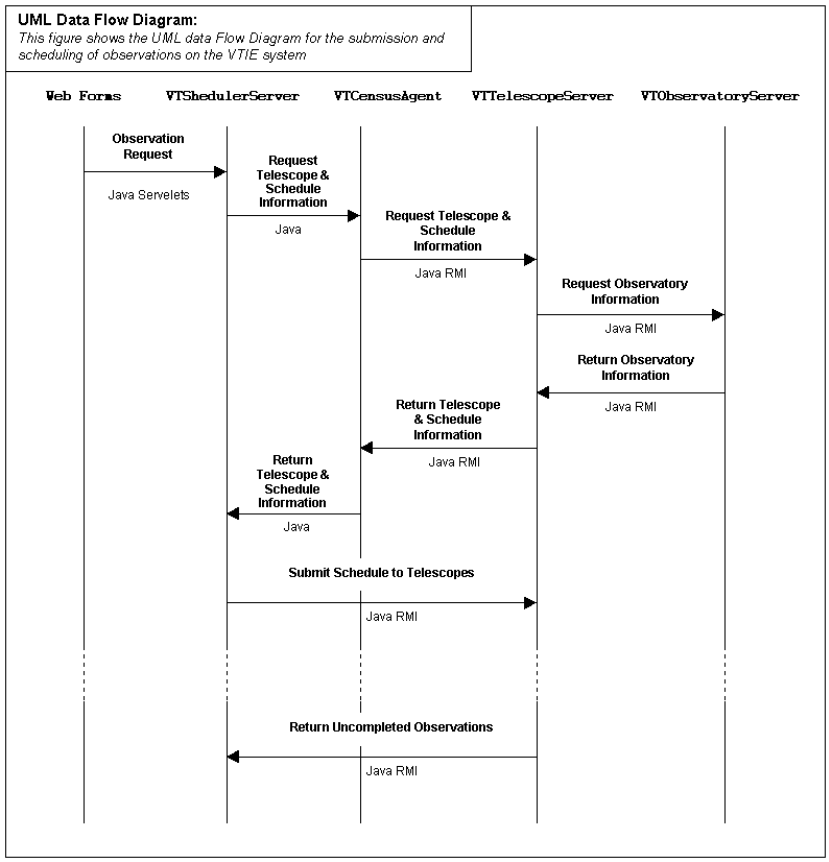


Figure 7. UML Data Flow Diagram

### 2.6.3 Class Structure

The structure of the main Java classes and interfaces used in the VTIE scheduling system is shown in Figure 8. There is a detailed description of the classes in the Appendix.

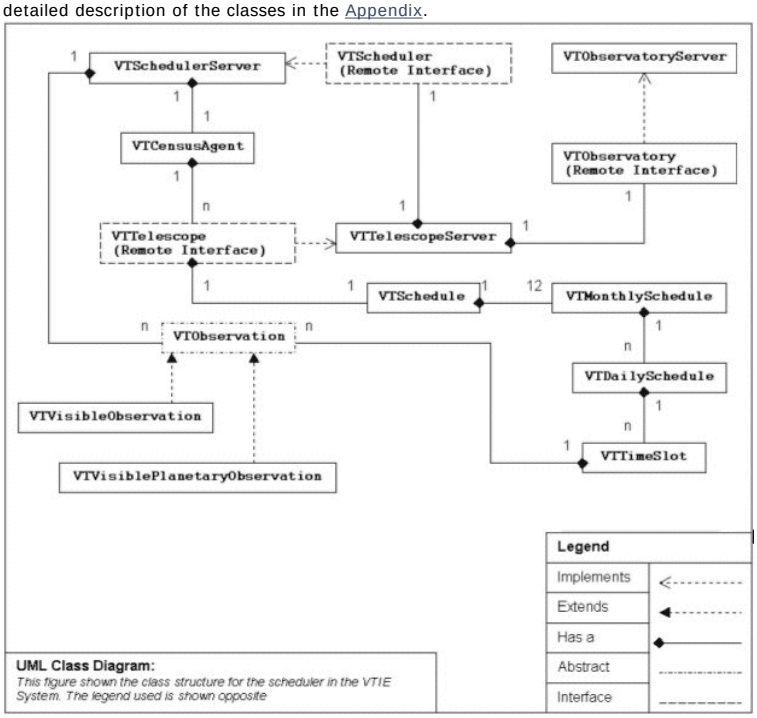


Figure 8. UML Class Diagram

As this project is concerned with providing global access to spatially distributed telescopes, each having local controllers, etc., we deemed it appropriate to develop a distributed solution requiring minimal or no change in the local operation of the telescopes. Our software would interface with the local telescope controller, and seamlessly integrate with a centralised monitoring agent (Census Agent). We use the Java RMI model to implement this distributed system as it provides powerful facilities for calling remote object methods, and the ability to treat remote objects as if they are local. Out test experiments have demonstrated that RMI is both scalable and sufficiently fast to allow the Census Agent (VTCensusAgent) to request the information from the telescopes (VTTelescopeServer) with ease.

In our system, the observing schedule is not stored centrally. Instead, each telescope contains the part of the total schedule that is relevant to it. This ensures that the demand on network resources will be decreased. When a telescope is making its observations, it will not need to access the network to obtain its schedule - it will just have to query its local Virtual Telescope

object (VTTelescopeServer). The only network traffic will be between the Census Agent (VTCensusAgent) and the telescopes, specifically, when the scheduler requests the current state of the system and again to write the updated schedule back to the telescopes after the scheduling process. The scheduler operates in a batch mode so this will not happen often.

A schedule consists of a list of 12 monthly schedules (VTMonthlySchedule), which in turn consist of a number of daily schedules (VTDailySchedule), which consist of a number of time slots (VTTimeSlot). Our motivation for this design is to allow telescopes to swap schedules with varying degrees of granularity. For example, if a telescope is in need of repair for three days, those three daily schedules could be collectively transferred to another free telescope by the scheduler.

### 2.6.4 Automated Optimizing Scheduler

Research on the scheduling component of the project is focusing on developing AI-based software and interactive constraint techniques to provide a scheduling capability for VTIE.

In the current TIE environment, requests from students are evaluated and scheduled on the telescopes by hand. With more telescopes coming online, and proportionally more student requests, the scheduling problem will become too complex and time-consuming to handle manually. In addition, changing weather and telescope conditions may require rescheduling existing telescope commitments dynamically.

To provide this capability, we are applying constraint-based scheduling techniques (see e.g. Le Pape 1994) to build a specialized scheduling system for VTIE. This system will incorporate a novel interactive constraint management system currently under development. The activities to be scheduled are those student requests that have been approved by a review panel. (The scheduling system will also be used to filter infeasible requests---those for which there is no available telescope---early in the review process.) The constraints associated with approved requests include:

- A description of the astronomical phenomenon or specific sky coordinates to observe;
- Telescope characteristics required for the observation, such as diameter and filters;
- The time of day or year that the observation needs to be taken;
- A priority associated with the requester.

The domain knowledge to be modeled includes individual telescope capabilities and available time slots, translation between sky coordinates and telescope latitude/longitude/altitude/azimuth, weather conditions, etc. The scheduling system will use this domain knowledge to identify the telescopes and times that could satisfy each request, allocate time on telescopes to particular student observations, and reschedule observations as needed due to changing telescope and weather conditions.

Our development approach is iterative, producing a series of scheduler releases with successively greater capabilities: (1) batch scheduling, (2) incremental scheduling, (3) cost-sensitive scheduling, and (4) dynamic scheduling. We now turn to a brief discussion of each of these capabilities.

**Batch scheduling**. A group of observation requests will be aggregated over a period of time and scheduled in a batch process. (The group of requests might be all of the requests over the course of some time period, e.g. one week, or the batch scheduler might be run every *n* requests, e.g. *n*=20.) This approach allows tradeoffs to be made in scheduling sets of observations. An alternative approach of first-come-first-served scheduling could cause locally preferable assignments to be made at the expense of globally optimal solutions, potentially violating VTIE program criteria, such as giving priority to certain groups or maximizing usage of the telescope resources. An automated batch approach allows the system to represent and optimize global criteria for a good schedule.

In the batch scheduler, requests that were previously scheduled will be treated as unchangeable (i.e. those slots would be unavailable for new requests). As shown in Figure 9, a greedy assignment algorithm is run first, to create an initial assignment of requests to telescope/time slots. An iterative repair method then identifies possible swaps in the schedule, that is, an unsatisfied or suboptimally scheduled request, which could be satisfied by another request's allocated time slot. If the latter request can be scheduled in a different time slot without sacrificing quality, the swap is made. This process of identifying and evaluating swaps repeats until no further improvement can be made.

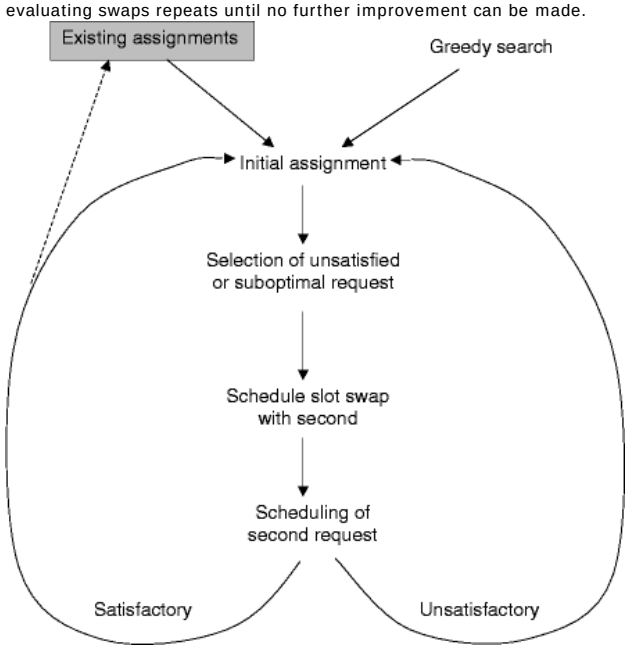


Figure 9. Batch scheduling process

**Incremental scheduling**. Extending the batch scheduler to behave incrementally is straightforward in principle. Existing requests would be rescheduled if a new request could make more effective use of that time slot. In Figure 9, the existing assignments (shaded box) feed into the initial assignment, and a feedback loop is added from the search process to possibly reassign those existing requests. Heuristic repair methods (Smith 1994) will be applied to identify schedule modifications that are likely to result in overall improvements.

The cost of rescheduling is that the existing observation's owner (i.e. the teacher or head of the requesting organization) would need to be asked whether the move is acceptable. Such costs would initially be modeled as a penalty in evaluating the swap. In the next phase of development (next section), we explicitly model these costs.

**Cost-sensitive scheduling**. We are developing a method of cost-sensitive constraint satisfaction (CSCS) that incorporates the costs of constraint checking into the constraint reasoning process. Using this method will allow the scheduler to be sensitive to the cost of evaluating alternatives (testing constraints). Constraints that are expensive or difficult to test are treated as having an additional cost that must be minimized. The goal is to develop a constraint satisfaction method that minimizes the total cost of constraint checking.

The number of constraint checks is commonly used as an evaluation metric for the time performance of constraint methods, but to our knowledge there is no previous research that allows constraints to have varying costs, or that treats the cost of constraint checks as an explicit optimization criterion. In our initial development, we plan to use a combination of variable ordering heuristics (placing the highest-cost constraints at the end of the search, attempting to minimize the total cost of constraint checks), intelligent backtracking (to avoid re-checking where possible), and constraint propagation over low-cost constraints (to reduce the domains of the variables as much as possible before checking the more costly constraints).

These cost-sensitive methods will enable two extensions to the scheduler: first, the user-interaction cost of rescheduling can be explicitly modeled; second, a tighter integration with the identification of available time slots is possible. For example, initially the telescopes could be polled for their availability over the next two months. If there were requests that could not be satisfied by the resulting time slots, another query could be issued to all or some of the telescopes. The cost of gathering this information could be modeled explicitly and incorporated into the search process.

**Dynamic scheduling**. The final extension is to enable rescheduling in the face of a dynamically changing situation. For example, the weather forecast could indicate that visibility will be unsatisfactory; a telescope could break or require unscheduled maintenance; or a scheduled observation could be preempted by a higher-priority request. In this case, as in the case of incremental scheduling, the goal is to reassign the affected requests in a way that minimally impacts other scheduled requests.

The incremental and cost-sensitive scheduling methods described will directly support a dynamic scheduling system. Incremental scheduling methods can be used to identify and evaluate alternative slots for an affected request. Cost-sensitive scheduling allows this process to minimize the ripple effect that results from rescheduling the affected requests.

### 2.6.5 Interactive Schedule Picker and Viewer

The request for observations will be collected using a point-and-click interactive picker (a sample screenshot is shown in Figure 10) that uses a calendar metaphor for the user interface. While a broad audience may be able to view availability, only Authorized Users (typically teachers) will be able to use the Picker to submit a request for an observation. The Picker, which appears in a separate window, defaults to the current day, month and year for the current locale (for example, GMT in Figure 10). Users will have the option of clicking through to the date of interest or by using the quick-jump buttons to go directly to the date of interest. This DHTML-enabled Picker communicates with Java Servlet software to present the user with up-to-date information about the current system loading. Usage is for the entire system, and not for a particular telescope, as observation requests will be automatically matched to available telescopes depending on a number of factors, including telescope availability, observation time slot, system loading, priority, etc. Unavailable times for a particular telescope may correspond to daylight hours, a non-VTIE time allocation, or other factors.


Figure 10. VTIE Interactive Telescope Inspector Window

A legend is provided at the bottom of the window to indicate the icons used in the interface. By providing users with an indication of system availability, they will offload some of the work from the Scheduler by requesting available dates up front. In addition, the Picker allows the user to browse other observation requests and identify projects of interest.

The Picker is almost entirely text-based (with the exception of the next and previous month buttons). This allows flexibility, such as dynamic generation to present the information to the user in different languages (Note 7).

A Schedule Submission Form is presented to the user once a potential date is chosen. This form (a sample screenshot is shown in Figure 11) appears as a new child window of the Picker window. Clicking on alternative days in the Picker causes an update of the Schedule Submission Form window for the new day. This form contains an interactive DTHML-element driven by a Java Servlet, which provides details of the schedule for that day. These details indicate that a time slot is available, unavailable, or will show a proposal ID. Clicking on an available slot will automatically set the Observation Start Time in the drop-down menu items below the schedule panel. Clicking on a later time will automatically set the Observation End Time. The user may alternatively select the start and end times directly using the menus. A valid proposal ID is required, and if the user has other proposals awaiting scheduling, they will automatically appear in the Proposal ID selection input menu. As soon as a proposal has been scheduled, it will be removed from this list. If, for some reason, scheduling is impossible, the user will be informed via the VTIE Messaging System. The *Submit Schedule Request* option delivers the request to a Java Servlet which

passes the schedule request to the Census Agent, which collects the requests and delivers them to the Scheduler at predetermined intervals.

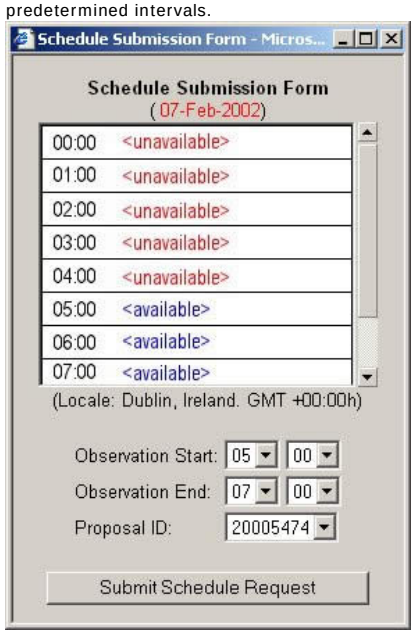


Figure 11. VTIE Schedule Proposal Submission Window

All users must log into the MyVTIE environment, therefore all information in the Schedule Submission Form will pertain to that user. If the user does not have proposals available for scheduling, then the schedule form inputs will not appear and it will become a Viewer (Figure 12), a browsing dialog giving details for the day selected in the Picker window. Proposals only become available in the Schedule Submission Form after they have been approved and have been allocated a Proposal ID.


Figure 12. VTIE Schedule Viewer Window

## 2.7 Testing

### 2.7.1 Approach

VTIE will begin testing with the initial prototype and continue with each subsequent iteration to ensure the design remains on course. Budget constraints preclude high-end options such as employing the services of usability consultants, although accurate and informative testing can be accomplished by the project team with as few as 4-6 individuals (Nielson and Tahir 2001; Krug 2000).

Pre-test questionnaires will collect basic information from outside test users to aid analysis of results and will collect individual computer use patterns. Questions will reveal how often the tester uses a computer, the primary purpose of use such as business or personal, online, gaming, or other. The level of computer use and competency of the testers will necessarily be a factor in analyzing results, i.e. one would not expect testers who rarely use computers for online activities to benefit from usability strategies based on users' prior experience and expectations from other online Web sites (Bergin 2001).

There are several proven testing strategies that we will employ in addition to pre-test field observations of our target users. Two basic testing procedures are often employed: fact-based questions, and judgment questions (Spool 1999). Fact-based test questions are straightforward. A test user will be presented with the site and asked to find a specific item or piece of information. A VTIE observer might direct the tester to find a teacher lesson plan. VTIE observers would record the time it takes a tester to locate the information while recording the steps the tester takes to find the information. If the tester finds the information, that test is considered successful. The time or number of clicks taken to find the information might qualify a successful completion. A test user who can find information after 5 minutes of searching and clicking is still technically successful but the length of time and large number of clicks would suggest a poorly designed interface confusing to the test user and in need of a serious overhaul (Bergin 2001).

Different users employ different strategies for locating information. Users fall into two categories: those who employ search engines immediately and a second group that rapidly scans a page selecting the first link they subjectively feel might be correct, referred to by Nielsen as search-dominant or link-dominant users (Nielsen 2000). Observing which of these strategies our test users prefer and are successful with will further refine our design.

### 2.7.2 Implementation

Testing of the VTIE modules described above will begin as the prototype software is released in Summer 2002. The testers will be a selection of second-level students and teachers from schools close to Maynooth. Groups will visit NUIM and be provided with basic system training in a formal setting. A more casual environment, in a modern Usability Laboratory, will be used to investigate the difficulties or successes associated with the prototype software. This laboratory will be based around 10 wireless notebook computers with high-speed access to the VTIE Server. Each computer may be monitored and all interaction will be logged for offline analysis.

Students from participating schools will be encouraged to try the software and then respond with personal feedback. As the research program increases and the software develops (MyVTIE and Authenticated Access), students will be able to access the information using their respective school computers. Teachers will also be trained on the components of VTIE for reviewing proposals and papers, and for interacting with the scheduling tools. The VTIE team will liaise with the secondary schools in the testing activities and help with any questions on usability. The liaison will report to the VTIE team any suggestions or comments that the students or teachers may have. This continuous interaction will enable the developers to create user-friendly and usable software.

## 3 Status and Summary

The NSDL project Virtual Telescopes in Education seeks to enhance science education through a Web-based interface to astronomy resources. One of VTIE's main goals is to enable learners to participate in the process of science by designing and carrying out astronomy observations. VTIE strives for maximal interoperability with other NSDL projects and Web resources by adopting a Java-based architecture and by following the Open Archives Initiative and the W3C Semantic Web activity. The first VTIE prototype will be tested with students in Summer 2002. Version 1 will be publicly available in Fall 2002. Inquiries may be directed to info@vtie.gsfc.nasa.gov.

## References


Barber, W. and Badre, A. (1998) "Culturability: The Merging of Culture and Usability". In *Proceedings of 4th Conference on Human Factors & the Web,* October 8
http://www.research.att.com/conf/hfweb/proceedings/barber/index.html

Bergin, S. (2001) "Visualization and Modeling of Highly Structured Mobile Phone Pricing Data". Computer Science Masters Thesis, National University of Ireland, Maynooth

Clark, G. (2002) "Telescopes in Education". Telescopes in Education, NASA Jet Propulsion Laboratory, http://tie.jpl.nasa.gov

Eads, S. (2000) "The Web's Still-Unfulfilled Personalization Promise". *BusinessWeek Online,* August 4
http://www.businessweek.com/bwdaily/dnflash/aug2000/nf2000084_506.htm

Hanson, B. (2001) "Building a More Perfect Digital Library, Bit by Bit". *UCAR Quarterly.* http://www.ucar.edu/communications/quarterly/fall01/library.html

Hawkins, M. (2001) "XML Document Storage: Native or Relational". *XML Asia Pacific 2001* http://www.planetmarkup.com/pdfs/MichaelHawkins.pdf

Horrocks, I., *et al*. (2001) "DAML+OIL (March 2001)" DARPA Agent Markup Language http://www.daml.org/2001/03/daml+oil-index.html

Jaggadish, H.V. (2002) "TIMBER". University of Michigan Database Group
http://www.eecs.umich.edu/db/timber/

Krug, S. (2000) *Don't Make me Think! A common sense approach to Web usability* (Indianapolis: QUE)

Lankes, D. (2002) "Integrating Expertise into the NSDL: Putting a Human Face on the Digital Library". Information Institute of Syracuse http://quartz.syr.edu/eduref/

Lassila, O. and Swick, R. (1999) "Resource Description Framework (RDF) Model and Syntax Specification". W3C Recommendation
http://www.w3.org/TR/REC-rdf-syntax/

Le Pape, C. (1994) "Scheduling as Intelligent Control of Decision-Making and Constraint Propagation." In *Intelligent Scheduling*, edited by M. Zweben and M. S. Fox (Morgan Kaufmann: San Francisco, CA), pp. 67-98

Martinez, M. (2001) "Key Design Considerations for Personalized Learning on the Web (Educational Technology and Society)". *Educational Technology and Society,* Vol. 4, No. 1, January
http://ifets.ieee.org/periodical/vol_1_2001/martinez.html

Nielsen, J. (2000) *Designing Web usability* (Indianapolis: New Riders)

Nielsen, J. (1998) "Personalization is Over-Rated (Current Issues in Web Usability)". *Alertbox*, October 4
http://www.useit.com/alertbox/981004.html

Nielsen, J. and Tahir, M. (2001) "Building Web Sites with Depth". *WebTechniques Magazine*, February
http://www.webtechniques.com/archives/2001/02/nielsen/

Shah, U., Finin, T., Mayfield, J. (2002) "Information Retrieval on the Semantic Web". University of Maryland Baltimore County Department of Computer Science and Electrical Engineering http://www.csee.umbc.edu/~finin/papers/aaai02abir/

Smith, S. F. (1994) "OPIS: A Methodology and Architecture for Reactive Scheduling". In *Intelligent Scheduling*, edited by M. Zweben and M. S. Fox (San Francisco, CA: Morgan Kaufmann), pp. 29-66

Spool, J., *et al*. (1999) *Web site usability: A designer's guide* (San Francisco: Morgan Kaufmann)

Stonebraker, M. (2001) "XML vs. Tables". High Performance Transaction Systems Workshop
http://research.microsoft.com/~jamesrh/hpts2001/presentations/XMLvsTables.ppt


## Notes

1. Currently, only formal education groups (i.e. teachers and students) have access to the Proposal Generation Tool. We intentionally use "learners" here, as we anticipate the capability in the second VTIE release to write observing proposals that will generate a query to the database for the data, rather than use of the TIE telescopes (which are restricted to formal education groups). There will be no access restrictions on the data.

2. Teacher training in TIE observational capabilities is essential, however, as teachers should know the types of questions to encourage. These capabilities will be detailed in the *See the Telescopes* function.

3. Later, when the TIE affiliates include spectrographic capabilities, the situation will become more complex, and we will re-evaluate our strategy.

4. It should be noted that proposals need not be rejected, as is possible for professional astronomers. Teachers are encouraged to provide constructive comments and suggestions for improvement until a valid proposal is constructed.

5. It is expected that at some later date, a collaborative paper-generation tool will be provided for multiple users to work simultaneously on a paper.

6. This most aggressive vision is a straw man. No-one is currently trying to implement such a vision, but it is useful to identify where in the continuum a semantic Web system lies.

7. For the prototype we propose to provide pickers in English, Spanish, Dutch and Gaelic. Initial development of the database will use a single language (English) to evaluate usability and development methodologies. The other languages will be added at a later stage.

## Appendix: Virtual Telescopes: Java Classes

**VTSchedulerServer: (Implements VTScheduler)**

An object of this class will run on the VTIE Server in NUIM. This object will be persistent (will serialise itself to disk regularly and restore from disk when restarted). The function of the VTSchedulerServer object will be to implement the Optimising scheduler currently being developed at UMBC. The VTSchedulerServer object will be visible to the software on other servers in the VTIE system through its remote interface VTScheduler.

**Properties:**

1. **theObservations** - a Vector of VTObservation Objects that need to be Scheduled
2. **theCensusAgent** - a VTCensusAgent

**Methods:**

1. **schedule** - the most important method implemented by this class is the method to do the actual scheduling
2. **addObservation** - a method to add an observation (VTObservation) to the Vector of observations. (for VTScheduler)

**VTScheduler: (Remote Interface)**

This is the remote interface for the VTSchedulerServer class. VTSchedulerServer implements the methods in this interface. An interface has no Properties.

**Methods:**

1. **addObservation** - a method to add an observation to the scheduler (VTSchedulerServer)

**VTCensusAgent:**

An object of this kind will be used by the scheduler (VTSchedulerServer) to gather information from all the telescopes (VTTelescopeServer) in the system.

**Properties:**

1. **theTelescopes** - a Vector of links to all the Telescopes (VTTelescopeServer) in the system

**Methods:**

1. **poll** - the only method in this class gathers information from all the telescopes (VTTelescope) in the System and formats it for the Scheduler (VTSchedulerServer)

**VTTelescopeServer: (Implements VTTelescope)**

An object of this class will be located on the server at each telescope and will store all the information about that telescope as well as the schedule (VTSchedule) for that telescope. This class also implements the methods of the remote interface VTTelescope.

**Properties:**

1. **theDetails** - the telescopes details
2. **theInstruments** - the instruments the telescope can use
3. **theSchedule** - the Schedule for the telescope
4. **theObservatory** - a link to the remote VTObservatoryServer object representing the observatory to which the telescope belongs.
5. **theScheduler** - a link to the scheduler (VTSchedulerServer)

**Methods:**

1. **getTelescopeDetails** - a method to return the details of the telescope (for VTTelescope)
2. **getInstrumentDetails** - a method to return the instruments the telescope has at its disposal (for VTTelescope)
3. **getFreeTimeSlots** - a method to return all the free time slots on the telescope (a time slot is the smallest unit of time used by the schedule and the scheduler. A time slot can hold a group of observations (VTObservation)) (for VTTelescope)
4. **setTimeSlot** - a method to set the contents of a time slot (for VTTelescope)
5. **canSchedule** - a method to determine if a given observation (VTObservation) can be scheduled on the telescope (for VTTelescope)
6. **returnObservations** - a method to return unfinished observations (VTObservation) to the scheduler (VTSchedulerServer)

**VTTelescope: (Remote Interface)**

This is the remote interface for the VTTelescopeServer class.

**Methods:**

1. **getTelescopeDetails** - a method to return the details of the telescope
2. **getInstrumentDetails** - a method to return the instruments the telescope has at its disposal
3. **getFreeTimeSlots** - a method to return all the free time slots on the telescope
4. **setTimeSlot** - a method to set the contents of a time slot
5. **canSchedule** - a method to determine if a given observation (VTObservation) can be scheduled

**VTObservatoryServer: (Implements VTObservatory)**

There will be an object of this class running on the server in each observatory. The object will hold all the information about the observatory and will be visible to the software on other servers via its public interface VTObservatory.

**Properties:**

1. **theCoords** - the co-ordinates of the observatory
2. **details** - any other information to be stored about the observatory

**Methods:**

1. **getCoords** - a method to return the observatories co-ordinates (for VTObservatory)
2. **getDetails** - a method to return the other information about the observatory (for VTObservatory)

**VTObservatory: (Remote Interface)**

This is the remote interface for the VTObservatoryServer class.

**Methods:**

1. **getCoords** - a method to return the observatories co-ordinates
2. **getDetails** - a method to return the other information about the observatory

**VTObservation: (abstract)**

This class is used to represent an observation request to the system. It stores all the information about the observation requested as well as who submitted it and when it was submitted. The class is abstract because there will be many different kinds of observation that can be made and they will all need different parameters. Only the common elements will be stored in the parent class. This class will have to be extended for each actual type of observation.

**Properties:**

1. **theOwner** - the UID of the person who submitted the observation request
2. **submittTime** - the Time the request was submitted
3. **timeWindow** - the time range over which the Observation is valid
4. **SNRatio** - the desired Signal to Noise Ratio of the Observation
5. **minRes** - the Minimum Resolution of the Observation
6. **minFoV** - the Minimum Field of View of the Observation

**Methods:**

1. There are accesser methods to all the data stored

**VTVisibleObservation: (Extends VTObservation)**

This class extends the abstract class VTObseravation and represents a visible observation of a sidereal object.

**Properties:**

1. **theCoords** - the co-ordinates of the observation (represented using the Right Ascension and Declination system)

**Methods:**

1. **getCoords** - a method to return the co-ordinates
2. **setCoords** - a method to set the co-ordinates

**VTVisiblePlanetaryObservation: (Extends VTObservation)**

This class extends VTObservation and represents a visible observation of a planetary body.

**Properties:**

1. **theTarget** - the name of the planetary body (the target of the observation).

**Methods:**

1. **getTarget** - a method to return the target of the observation
2. **setTatget** - a method to set the target of the observation

**VTSchedule:**

This object represents the schedule of a telescope for a year from the current day. It stores this schedule as a list of 12 Monthly Schedules (VTMonthlySchedule) which in turn store the schedule as a list of Daily Schedules (VTDailySchedule) which then store the observations (VTObservation) per time slot (VTTimeSlot). All the sub components of this class are declared as private, hence a break down of the Monthly and Daily Schedules is not needed.

**Properties:**

1. **granularity** - the length of each time slot

**Methods:**

1. **getSchedule** - a method to return the entire schedule
2. **canSchedule** - a method to determine if a given observation (VTObservation) can be fitted into this schedule

3. **getFreeTimeSlots** - a method to return all the free time slots in this scheduler
4. **getTimeSlot** - a method to return the time slot that represents a given time

**VTTimeSlot:**

This class represents a time slot in the schedule of a telescope. Each time slot holds a number of observations that the telescope should carry out during the time represented by that time slot.

**Properties:**

1. **startTime** - the time slot's starting time
2. **theDuration** - the time slot's duration
3. **theObservations** - a list of observations (VTObservation) that should be attempted during this time slot.

**Methods:**

1. **getObservations** - a method to return the observations (VTObservation) stored in the time slot
2. **addObservation** - a method to add observations (VTObservation) to the time slot
3. **removeObservation** - a method to remove observations (VTObservation) from the time slot
4. **canSchedule** - a method to see if this time slot can hold a given observation (VTObservation)