N95- 23743

The Associate Principal Astronomer Telescope Operations Model

Mark Drummond* John Bresina* Keith Swanson† Will Edgington* Greg Henry‡

*Recom Technologies
†AI Research Branch, Mail Stop: 269-2
NASA Ames Research Center
Moffett Field, CA 94035-1000
Phone: 415.604.4710, *Fax:* 415.604.3594
E-mail: xfr@ptolemy.arc.nasa.gov

‡Center of Excellence in Information Systems
Tennessee State University
330 Tenth Avenue North
Nashville, TN 37203-3401
Phone: 615.251.1225, *Fax:* 615.251.1117
E-mail: henry@schwab.tnstate.edu

KEYWORDS AND PHRASES

Associate principal astronomer, automatic telescope, automatic telescope instruction set, principal astronomer, telescope operations.

ABSTRACT

This paper outlines a new telescope operations model that is intended to achieve low operating costs with high operating efficiency and high scientific productivity. The model is based on the existing *Principal Astronomer* approach used in conjunction with ATIS, a language for commanding remotely located automatic telescopes. This paper introduces the notion of an *Associate Principal Astronomer*, or APA. At the heart of the APA is automatic observation loading and scheduling software, and it is this software that is expected to help achieve efficient and productive telescope operations. The purpose of the APA system is to make it possible for astronomers to submit observation requests to and obtain resulting data from remote automatic telescopes, via the Internet, in a highly-automated way that minimizes human interaction with the system and maximizes the scientific return from observing time.

BACKGROUND

Research quality telescopes located at prime observing sites have always been a scarce resource, and astronomers have had to work with limited access to these telescopes. Typically, observing time is allocated to an individual astronomer a few times per year in short contiguous blocks of a few nights each. Furthermore, the astronomer has needed to be physically present at the telescope in order to operate his instrumentation for data acquisition. Limited access, block allocation, and local operation have restricted both the amount of data that can be gathered and the type of observational campaigns that can be accomplished.

More recently, sophisticated network and communication technologies have enabled a number of new approaches where astronomers may participate in an observation program from a remote location. These approaches range from remote verbal communications with the on-site telescope operations staff to actual remote control of a telescope with real time video feedback [4]. Such remote observations provide flexibility by allowing the observer to be physically distant from the telescope yet remain in direct control. However, even in this remote observing paradigm, the astronomer must still be involved during the execution of the observing program, and human presence at the observatory is often still required.

Fully automatic telescopes represent an extension to the remote observing paradigm, allowing an astronomer to be removed from the telescope both temporally as well as spatially. For example, Fairborn Observatory (Mt. Hopkins, Arizona) and AutoScope Corporation (Fort Collins, Colorado) have designed and built software and hardware systems for the control of modest-aperture telescopes equipped with photoelectric photometers to measure stellar brightness. These systems make it possible for a remotely located telescope to operate unattended for significant periods (up to a number of months). These telescopes execute commands provided by an astronomer in such a way that the astronomer is not required to participate in the execution of the observing program. It is in this sense that these telescopes are *fully automatic*.

While the majority of existing ground-based automatic telescopes are used for aperture photometry, automation support for spectroscopy and imaging has been increasing (primarily due to the efforts of R. Kent Honeycutt and Don Epand [3]). Genet and Hayes [6] describe automatic photoelectric telescopes in some detail.

For the sort of telescope we are considering, the language used to define observation requests is the Automatic Telescope Instruction Set, or ATIS [3]. In ATIS, a *group* is the primitive unit to be scheduled and executed. A group is a sequence of telescope commands and instrument commands defined by an astronomer to accomplish the observation of an object of interest. A group contains commands to move the telescope, to control the filters, and to gather data in a defined sequence. In the initial version, ATIS89, the only instruments accommodated were photometers, but the most recent version, ATIS93, also includes commands to obtain CCD camera images.

In addition to specifying the syntax and semantics for observation requests and results, the ATIS standard provides a set of *group selection rules* that are used to determine the execution order of groups during the night. The group selection rules provided by ATIS essentially implement a first-to-set-in-the-west policy: at any given point in time the telescope observes the star that will set next. It is possible to improve upon this default group selection policy by using more sophisticated scheduling techniques. Specifically, it is possible to improve the quality of the data by more precisely scheduling groups so that observations are taken at lower airmass (on average), and so that observations are obtained at astrophysically interesting times. Additionally, for a multi-user telescope, better scheduling can result in a fairer allocation of telescope time to requesting astronomers.

We were invited to be part of the International Astronomical Union ATIS93 standardization committee to assist with ATIS extensions in support of advanced scheduling. Along with other committee members, we designed a new group selection advice statement. This new statement is used to override the default ATIS group selection rules. The committee also agreed on a mechanism for communication with a telescope controller in terms of incremental ATIS93 partial input and partial output files. Together, these new features make it possible to implement a non-native (*i.e.*, external) scheduler that can effectively drive a telescope's controller to better serve the scientific objectives of participating astronomers.

Our new approach to the automatic management of remotely located telescopes is based on ATIS93. At the heart of our approach is automatic observation loading and scheduling software, and it is this software that is expected to help increase science quality and telescope productivity. Our goals are to provide software tools to assist managers of multi-user automatic telescopes and to make it possible for participating astronomers to have their observation requests scheduled on and their resulting data returned from remotely located telescopes, via the Internet, without the necessity of daily human intervention.

THE CURRENT APPROACH

Before we explain how we intend to improve telescope management and use, we need to briefly explain the current manner in which automatic ATIS-compatible telescopes are managed. This is illustrated in the left half of Figure 1 and briefly described by the following scenario.

First, an astronomer forms a set of groups consistent with the scientific goals of his or her observation campaign. For any given automatic telescope, there is a single *Principal Astronomer* or PA. The PA manages the set of requests that are loaded onto the telescope. Thus, once an astronomer has assembled a set of ATIS groups, these are sent to the appropriate PA, typically via e-mail, Internet FTP, or on floppy discs in the postal service.

The PA collects together the sets of requests from participating astronomers and attempts to ensure that the total set of groups is desirable – that the telescope load makes good utilization of observing time and is fair to all participating astronomers, that there are appropriate groups for quality control and data reduction, *etc.* Then the complete set of groups is sent to the computer controlling the telescope. Communication between the PA and the telescope controller is typically carried out using personal computers connected via the Internet or modems and phone lines. The important aspect of the communication is that the PA can be located anywhere on the planet (in principle) and need only have access to an appropriate communication link.

The telescope controller uses its built-in ATIS group selection rules to implement a form of heuristic dispatch scheduling. At any point in time, the rules recommend a single group to execute next. The groups are executed by the telescope controller for some number of nights (often months); eventually, the PA requests from the controller the results that have been collected thus far. The collected data are returned to the PA as a results file specified within the ATIS language. The results include the raw data obtained from the observations, as well as a chronological record of the groups that were executed and relevant observing parameters to help with data reduction. The PA edits the results file and sends each astronomer the pieces corresponding to his or her requested observations (again typically via e-mail, Internet FTP, or on floppy discs). In some cases the PA provides a data-reduction service, returning reduced results, not simply raw data.

THE APA MODEL

The goal of our project is to provide automation support for all aspects of ATIS-compatible telescope management. Our focus is on providing software tools to help a PA who represents a community of participating astronomers; however, the increased automation also improves the way in which the astronomers interact with a PA. The right half of Figure 1 and the following scenario illustrate a new way of doing business with ATIS-based automatic telescopes that we are in the process of making possible. More details on the APA operations model are available in Bresina *et al.* [1].

From an Astronomer's Perspective

An astronomer creates an ATIS93 observation request file and sends it via electronic mail to the PA's computer. Let us refer to this computer as the *Associate Principal Astronomer*, or APA. The mailed file is automatically received and parsed to check for syntax errors. If the file adheres to the ATIS93 specification, then the APA e-mails a message back to

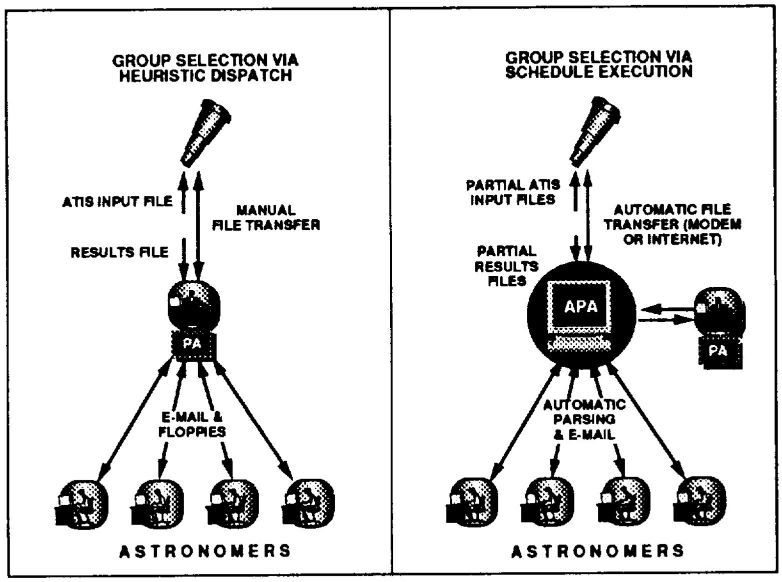

Figure 1: Operation of ATIS-compatible telescopes without (*left*) and with (*right*) the APA.

the astronomer acknowledging successful receipt of the request file; otherwise, a message is e-mailed back identifying the syntactic errors in the astronomer's file. At the end of each observing night, the APA e-mails the astronomer the results of those observation requests that were serviced that night, along with the results necessary for data reduction and data quality assessment.

From the PA's Perspective

The APA divides the overall problem of group scheduling into two subproblems: first, it assigns a group to execute on a given set of nights; second, for any group that has been selected for execution *tonight*, it assigns that group specific times through the night at which to execute. The first process is called *loading*, and its temporal scope covers many months. The second process is called *night scheduling*, and it is concerned with the seconds, minutes, and hours within a given night. After loading and night scheduling, a new combined ATIS93 file is automatically assembled by the APA. The PA can check how the controller will handle this new request file by displaying a prediction of telescope behavior for the night based on the best schedule found by the night scheduler (*i.e.*, what observations are likely to be made if the weather is ideal). If the PA is not satisfied with the prediction, then the manner in which the APA loads and schedules the observations can be modified. The next morning, the results of the night's observations are already stored at the APA. If the PA wants to assess the quality of the night's observation schedule and results, the actual telescope behavior can be displayed. Once the PA has tuned the APA to consistently produce high quality schedules, the APA takes care of routine observation loading and scheduling with only occasional supervision. A more complete description of the loader is given by Bresina [2], and a description of one of the techniques used in the night scheduler is given by Swanson, Bresina, & Drummond [9].

From the Telescope Controller's Perspective

Just before nightfall, the ATIS93 input file is automatically transferred to the telescope controller along with the observation schedule. The controller executes the schedule and, at the end of the night, transfers the ATIS93 output file back to the APA. This is the minimum amount of interaction between the telescope controller and the APA; however, the ATIS93 specification also allows for partial input and partial output files to be transmitted during the night. The partial output files enable the telescope behavior and status to be monitored during the night – either by a person (for example, to check the status of the telescope mechanics and optics) or automatically by the APA. The partial input files enable the APA to transmit new schedules and new groups during the night when necessary. For example, the APA can dynamically reschedule due to a change in the quality of observing conditions or due to an urgent observation request received during the night.

DISCUSSION

The overall goal of our project is to provide automation support for the management and use of remotely located, automatic telescopes. So far, we have focused on building the core of an Associate Principal Astronomer, or APA. This core consists of an automatic group loading and scheduling mechanism, together with a means for automatic schedule execution and dynamic rescheduling. While this core provides important functionality, there are many aspects of the PA's job that it does not address. In collaboration with other astronomers, we are currently expanding the set of functions offered by the APA to include automatic handling of ATIS request files, preliminary quick-look data reduction, and quality control measures. Experience gained with simulation tests and preliminary tests on an automatic telescope have been encouraging.

It is clear that the ATIS model is not the only one for automatic telescope management. Others have built APA-like systems [8]. The primary advantage of the APA is that it uses advanced scheduling techniques and operates with any telescope that adheres to the ATIS93 standard. Of course, NASA has a number of orbital telescopes that are operated remotely. The Hubble Space Telescope (HST), for instance, is operated in a way that is somewhat similar to our APA model. However, there is a significant amount of human infrastructure associated with the management of HST. Such infrastructure is expensive, and it cannot be replicated for every single telescope that is to be run automatically. Clearly, the human infrastructure surrounding HST performs useful tasks that our APA model ignores: for instance, helping users formulate their telescope requests and helping users make sense of the data they obtain. Our APA model leaves all such tasks firmly in the hands of telescope users (and their scientific community).

Our APA operations model requires one workstation (or a high-end personal computer), one experienced astronomer to act as the telescope PA, and one Principal Engineer / technician (PE) to fix the telescope and observatory control systems when things go wrong. A number of telescopes can be managed by a single APA, PA, and PE team.

One of us (GWH) has been working as a PA for a number of years with automatic telescopes. Together with Lou Boyd (of Fairborn Observatory) acting as PE, several telescopes have been operated automatically on Mt. Hopkins in southern Arizona to accomplish various scientific programs. The efficiency of operations for these telescopes has been estimated to achieve a dollar-cost-per-observation that is 30 to 40 times cheaper than previously possible using traditional manual telescope operations [7]. There has also been an enormous increase in observational throughput: the combined *yearly* output of the automatic telescopes managed by GWH would require a *lifetime* of effort to obtain by previous manual methods of operation.

To date, each of these automatic telescopes has been dedicated to a specific, long-term observing program. Thus, the operating schedule for each telescope has been extensively tuned by the PA (and sole user) to achieve acceptable performance. However, even small changes to the observing program make it very difficult to optimize the loading and scheduling. For multi-user telescopes, such extensive manual tuning is infeasible. In this context, our goal is to simplify and optimize the operation of single-user automatic telescopes and then to extend this simplified management structure to multi-user telescopes.

ACKNOWLEDGMENTS

Thanks to Bill Borucki, Lou Boyd, David Genet, Russell Genet, Butler Hine, George McCook, Ann Patterson-Hine, and Boris Rabin. Thanks especially to Ellen Drascher, who designed and implemented the APA's graphical user interface.

REFERENCES

[1] Bresina, J., Drummond, M., Swanson, K., and Edgington, W. 1994. Automated Management and Scheduling of Remote Automatic Telescopes. In *Astronomy from the Earth and Moon*, the proceedings of the 103rd Annual Meeting of the Astronomical Society of the Pacific, D.P. Smith (ed).

[2] Bresina, J. 1994. Telescope loading: a problem reduction approach. In *Proceedings of the Third International Symposium on Artificial Intelligence, Robotics, and Automation for Space*. Jet Propulsion Labs. Pasadena, CA.

[3] Boyd, L., D. Epand, D., Bresina, J., Drummond, M., Swanson, K., Crawford,D. Genet, D., Genet, R., Henry, G., McCook, G., Neely, W., Schmidtke, P., Smith, D., and Trublood, M. 1993. Automatic Telescope Instruction Set 1993. *International Amateur Professional Photoelectric Photometry (IAPPP) Communications*, No. 52, T. Oswalt (ed).

[4] Emerson, D.T., and Clowes, R.G. (eds.) 1993. *Observing at a Distance* (Proceedings of a Workshop on Remote Observing). World Scientific Publishing, River Edge, NJ.

[5] Genet, D. 1994. AutoScope Control System Reference Manual (Software Version 2.0). AutoScope Corporation, Ft. Collins, CO.

[6] Genet, R.M., and Hayes, D.S. 1989. *Robotic Observatories: A Handbook of Remote-Access Personal-Computer Astronomy*. AutoScope Corporation, Ft. Collins, CO.

[7] Henry, G.W. *forthcoming*. The Fairborn/TSU automatic telescope operations model. In *Proceedings of the symposium on Robotic Telescopes*, part of the 106th meeting of the Astronomical Society of the Pacific. Flagstaff, AZ.

[8] Richmond, M., Treffers, R., & Filippenko, A. 1993. The Berkeley Automatic Imaging Telescope. *Publications of the Astronomical Society of the Pacific*. No. 105 (October).

[9] Swanson, K., Bresina, J. and Drummond, M. 1994. Robust Telescope Scheduling. In *Proceedings of the Third International Symposium on Artificial Intelligence, Robotics, and Automation for Space*. Jet Propulsion Labs. Pasadena, CA.

Session PS-DS

Planning and Scheduling Workshop: Decision Support Aspects

Decision Support Aspects

N95- 23744

DTS: Building Custom, Intelligent Schedulers

Othar Hansson and Andrew Mayer
Heuristicrats Research, Inc.
1678 Shattuck Avenue, Suite 310
Berkeley, CA 94709-1631
(510) 845-5810, Fax: (510) 845-4405
{othar, mayer}@heuristicrat.com

KEY WORDS AND PHRASES

Decision theory, heuristic search, optimization, scheduling, user interface.

ABSTRACT

DTS is a *decision-theoretic scheduler*, built on top of a flexible toolkit—this paper focuses on how the toolkit might be reused in future NASA mission schedulers. The toolkit includes a user-customizable scheduling interface, and a "Just-For-You" optimization engine.

The customizable interface is built on two metaphors: objects and dynamic graphs. Objects help to structure problem specifications and related data, while dynamic graphs simplify the specification of graphical schedule editors (such as Gantt charts). The interface can be used with any "back-end" scheduler, through dynamically-loaded code, interprocess communication, or a shared database.

The "Just-For-You" optimization engine includes user-specific utility functions, automatically compiled heuristic evaluations, and a postprocessing facility for enforcing scheduling policies. The optimization engine is based on BPS, the Bayesian Problem-Solver [1,2], which introduced a similar approach to solving single-agent and adversarial graph search problems.

DTS SYSTEM OVERVIEW

The Decision-Theoretic Scheduler, DTS, is designed to support scheduling of over-subscribed, long-running projects. DTS is literally implemented as a program in a specialized language for the design of scheduling and optimization systems. This DTS Customization Language (DCL) is implemented on top of the public-domain TCL/Tk system [3].

DTS has been designed for science-planning on NASA missions. We are preparing to deploy the system as one component of a cost-reduction program within the Extreme Ultraviolet Explorer mission of the Center for Extreme Ultraviolet Astrophysics at the University of California, Berkeley [4].

We have explicitly designed DTS to be customizable by users, and thus transferrable to other missions. An easily customized scheduling system can reduce costs by eliminating the mission-specific paperwork and "workarounds" that result when a system does not address a scheduling scenario completely.

To reduce mission costs further, we have designed DTS so that such extensions can be made quickly and without corrupting existing code or functionality. For example, the current DTS interface provides much of the functionality of commercial project scheduling tools, but is implemented in under 7000 lines of DCL code. User modifications—such as an import "filter" for a pre-existing file format, or a specialized report writer—typically require only a few dozen lines of DCL code. Because DCL code is interpreted, programming errors are safely trapped.

Behind the scenes, the DTS "back-end" contains a sophisticated constraint-satisfaction search engine for use in automated scheduling. The use of decision theory permits user preferences and requirements to be modeled in a mathematically coherent way. The result is that DTS can typically find near-optimal solutions to the user's actual problem, with optimality measured in the user's terms. Many existing scheduling techniques restrict both the definition of optimality and the representation of the problem: the user is forced to use a system that provides a quasi-optimal solution to an approximation of the problem.

Our research goal in the DTS back-end has been to provide a rich representation for problems and preferences, and still find near-optimal solutions through the use of compilation, learning and decision-theoretic search.

In this paper, we describe customization in both the front-end and back-end, and then conclude with a description of future plans for applying DTS to NASA missions.

USER INTERFACE CUSTOMIZATION

The DTS interface uses objects and dynamic graphs to support customization.

All data in the system is represented within an object hierarchy. The hierarchy includes Task objects, Constraint objects, etc., as you would expect. These basic objects can be subclassed, or specialized, for the needs of an individual application: in the NASA version of DTS, an Observation object represents each Task that is an astronomical observation.

The system also includes "management" information objects such as (astronomical) Targets, (scientific) Proposals, and Principal Investigators. This information is linked to "problem" information such as tasks by the use of cross-reference attributes. For example, each Observation has an attribute named Target that is a cross-reference.

The DTS interface is centered on an object browser (Figure 2). Customization begins by defining a new object class, or redefining an existing object class. Each object class has an associated form, used to display and edit object instances in the browser. A simple default form is inferred from the "type" of each attribute (String, Date, etc.).

More complex forms require the use of DCL code. Figure 2 shows the form for a TemporalConstraint instance. This is the most complicated form in the system, but it requires only 40 lines to produce a specialized display

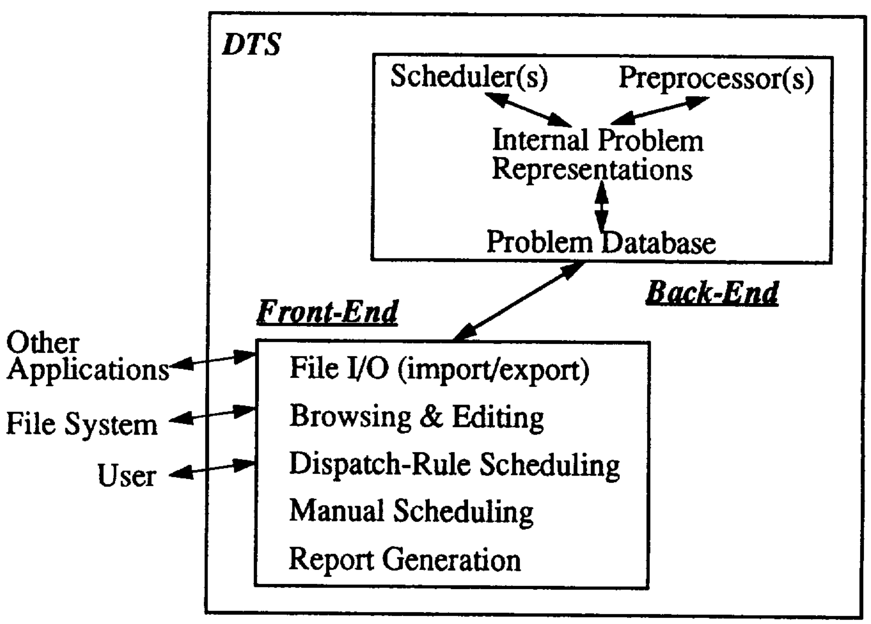

Figure 1. Overview of DTS System Architecture.

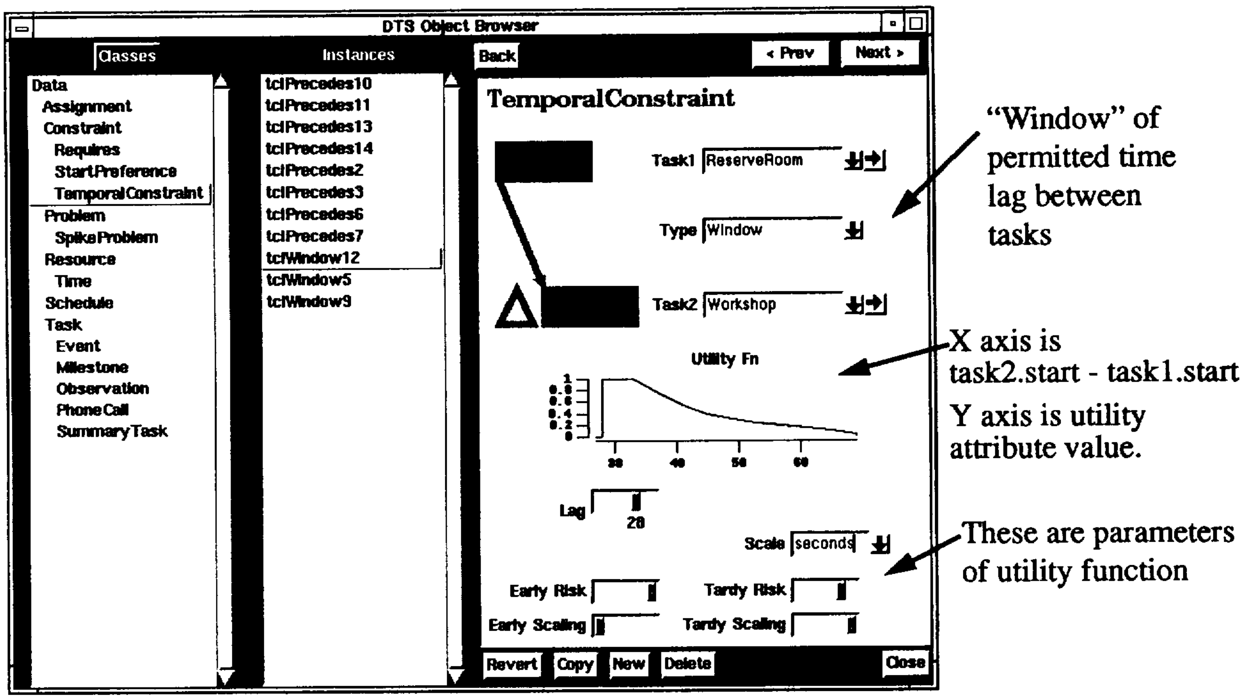

Figure 2. Example Customized Form in the Object Browser.

for a number of interrelated attributes. Like most binary constraints, the temporal constraint has two task parameters. In addition, for constraints of type "window," a utility function is defined by the parameters at the bottom of the form. These parameters are "animated" in a utility graph. Finally, each type of constraint has an associated graphical mnemonic (the upper left of the form), which reminds the user of the nature of the constraining relationship.

The second major mechanism in the DTS user interface is the dynamic graph. Dynamic graphs are editable "views" of a number of objects, built using an X-Y graphing metaphor. For example, a typical Gantt chart is an X-Y plot of tasks (Y), using their start time and duration (X). The DTS dynamic graph permits views such as Gantt charts, PERT charts, constraint matrices and resource histograms to be specified easily. These graphs are dynamic in that callbacks can be associated with user actions (e.g., mouse events), and defined to modify the underlying data appropriately.

Each of the basic views implemented thus far has required approximately 250 lines of DCL code for layout and callbacks. Application-specific views (such as augmented Gantt charts, statistical summaries, etc.) should be implementable with similar effort.

OPTIMIZER CUSTOMIZATION

The DTS back-end includes C++ routines, callable through DCL, that perform basic preprocessing and scheduling tasks. This optimization engine uses decision-theoretic search mechanisms developed by the authors in previous and ongoing work with the Bayesian Problem-Solver [1,2,5].

The use of decision theory [6,7,8] enables the engine to guide its search by user-specific utility functions, in addition to heuristic evaluation functions. Many existing schedulers use heuristic functions alone, but heuristic functions can confuse the role of schedule evaluation (utility) and search control (heuristics).

DTS collects statistics that relate heuristic evaluations to attributes of the utility function. Because these statistics relate to inputs rather than outputs of the utility function, the function itself can be modified without invalidating the statistics that have been gathered. The use of statistical estimation and probabilistic inference in DTS also permits multiple heuristic evaluations to be combined to focus the search more effectively. For example, a general-purpose constraint-satisfaction heuristic might be coupled with a domain-specific heuristic [5].

In an early phase of development, we found that the costs of state generation and heuristic evaluation were a significant bottleneck to the development of sophisticated scheduling search control. DTS thus also employs an experimental compilation mechanism that derives a specialized data structure for search tree "states" from a formal specification of the heuristic function. Hand-coding of such data structures reduces the overall cost of search significantly, and we anticipate that the automation of these data structures will permit these benefits to be achievable for users relying on domain-specific heuristics. Hansson [9] describes the compilation mechanism in more detail.

Finally, the use of DCL permits a user to code a secure "audit" or "checker" routine to validate a finished schedule before execution, or to enforce certain scheduling policies that are hard to represent within the system.

Along with other DTS features, these three mechanisms—decision-theoretic search with user-specific utility functions, data structure compilation for fast heuristic evaluation, and postprocessing for schedule validity—have been designed to ensure that DTS finds solutions to the user's *real* problem with a minimum of search cost.

CONCLUSION

We are presently customizing DTS for possible use within current and future NASA missions (including EUVE and CASSINI), and collaborating with NASA researchers to reuse the DTS interface on top of their schedulers.

We feel that the customizability of DTS can permit future NASA missions to exploit "economies of reuse" and "economies of fidelity." Economies of reuse are well-known: they result when development costs are cut by reusing flexible software.

Economies of fidelity result when a system can be made to solve a large portion of an application task, without a great degree of simplification. Many search and optimization frameworks require the user to simplify or abstract their problem into a restricted modelling language. This increases the cost of using such systems, and reduces the benefits: the solutions found are not always executable, let alone near-optimal, solutions to the real problem. On the other hand, systems like DTS, and Muscettola's HSTS [10], attempt to provide a richer framework for modeling the problem. DTS focuses on preference modeling, while HSTS focuses on constraint and state-variable modeling. We anticipate that compilation and learning techniques will permit these rich representations to be searched efficiently.

BIBLIOGRAPHY

[1] Hansson, O. and A. Mayer. "Heuristic Search as Evidential Reasoning." In *Proc. of the Fifth Workshop on Uncertainty in Artificial Intelligence*, Windsor, Ontario, August 1989.

[2] Mayer, A. *Rational Search.* Ph.D. Dissertation, Univ. of California, 1994.

[3] Ousterhout, J. *TCL and the Tk Toolkit.* Addison-Wesley, Reading, MA, 1994.

[4] Center for EUV Astrophysics. *EUVE Guest Observer Program Handbook.* Appendix G of NASA NRA 92-OSSA-5. Berkeley, Jan. 1992.

[5] Hansson, O. and A. Mayer. "Decision-Theoretic Control of Artificial Intelligence Scheduling Systems." HRI Technical Report No. 90-1/06.04/5810, Berkeley, CA, September 1991.

[6] Pearl, J. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, San Mateo, CA, 1988.

[7] Savage, L. J. *The Foundations of Statistics.* Dover, New York, 1972.

[8] von Neumann, J. and O. Morgenstern. *Theory of Games and Economic Behavior.* Princeton University Press, 1944.

[9] Hansson, O. *Bayesian Problem-Solving applied to Scheduling*. Ph.D. Dissertation, Univ. of California, 1994.

[10] Muscettola, N. *HSTS-DDL Manual*, NASA Ames Research Center, Code FIA, March 1994.