Managed Intelligent Deconfliction and Scheduling for Satellite Communication

Richard Stottler
Stottler Henke Associates, Inc,
1650 S. Amphlett Blvd., Suite 300
San Mateo, CA 94404
650-931-2700
Stottler@StottlerHenke.com

Robert Richards
Stottler Henke Associates, Inc,
1650 S. Amphlett Blvd., Suite 300
San Mateo, CA 94404
650-931-2700
Richards@StottlerHenke.com

Abstract— **The Air Force Satellite Control Network (AFSCN) coordinates hundreds of satellite communication requests from various users every day. Building a conflict-free schedule from a large set of satellite communication requests is a difficult problem. Human schedulers are very adept at generating high-quality solutions, usually allowing all requests to be serviced. However, this process is time-intensive and requires highly trained, experienced individuals. That is, the teams of highly trained and experienced schedulers must manually check every schedule request received. Approximately half of all requests require adjustment to remove conflicts.**

The US Air Force (USAF) had an interest in further automating this process. Stottler Henke worked with the USAF to develop the Managed Intelligent Deconfliction and Scheduling (MIDAS) solution, an artificial intelligence (AI) tool for automatically scheduling satellite contacts, by incorporating the experience, insights, and expertise of human schedulers.

MIDAS can now automate virtually all the scheduling of the satellite communication requests for the AFSCN, allowing schedulers to apply their expertise where it is really needed. MIDAS accomplishes this with a two-stage process that first shuffles tasks within their defined constraints before carefully applying a user-definable set of business rules that allow certain constraints to be relaxed when necessary. The system provides a familiar, user-friendly interface modeled on legacy Electronic Schedule Dissemination (ESD) systems to facilitate comparison and to allow users to switch from one interface to the other with relative ease. It runs on inexpensive consumer hardware and communicates with legacy systems via a well-defined plain-text file format: raw scheduling requests are imported to MIDAS, and scheduling results can be exported back to legacy tools.

MIDAS now provides Air Force planning at a level not previously possible. A viable schedule can be assembled in a matter of minutes in order to assess the impact of possible outages, events, expansion of equipment, etc., that is, MIDAS provides the ability to perform "what-if" scenarios to assess the impact of a potential event or mission change. With only a few minutes of processing, MIDAS is able to deconflict all or virtually all communications requests for a given day. MIDAS has eliminated much of the repetitive work involved in scheduling and allows schedulers to focus on other important problems.

This paper provides a history of MIDAS, an overview of its architecture and the many benefits MIDAS provides; in addition to its general applicability to non-USAF satellite communications scheduling.

978-1-5386-2014-4/18/$31.00 ©2018 IEEE

TABLE OF CONTENTS

1. INTRODUCTION

The Air Force commands and controls a variety of satellites through a global network of antennas and ground support equipment. Each constellation of satellites (e.g. the GPS satellites) is commanded from one satellite operations center (SOC), see Figure 1. Each constellation's controlling organization makes satellite communication support requests for the antennas and other ground support equipment (including limited bandwidth for each multi-antenna site as a whole) independently of the others to a central scheduling organization which must deconflict the competing requests. The most obvious constraint on this process is that there must be line of sight between the antenna and the satellite. For satellite communications, the bottleneck resource is the limited number of ground stations.

Figure 1. Satellites, Ground Station & SOC

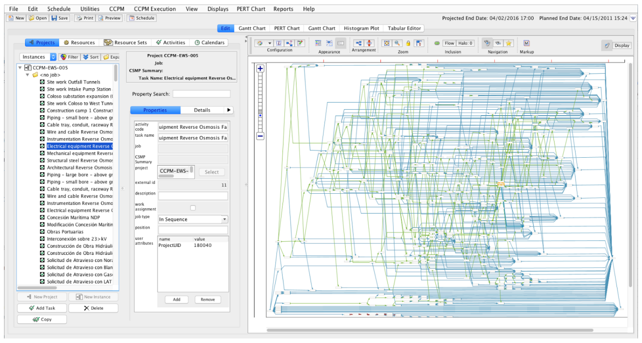
Satellite schedulers working for the Air Force Satellite Control Network (AFSCN) are responsible for scheduling hundreds of millions of dollars' worth of ground satellite equipment, squeezing out the highest communication capability possible while protecting the billions of dollars' worth of on-orbit satellites that are crucial for our nation's defense. The schedulers try to meet the original requests as closely as possible. In a typical single day, there are normally more than 600 support requests and usually more than half are in conflict with each other. Many of the conflicts are seemingly unsolvable, e.g. if there is only one antenna at a site and two requests for that antenna at the same time, the conflict is seemingly unsolvable. Yet this organization produces a conflict-free schedule daily, while meeting all requests. Meeting all (or as many as possible) support requests as closely as possible is the main objective.

The AFSCN does coordinate the circa 600 hundred satellite communication requests from various users every day. Building a conflict-free schedule from a large set of satellite communication requests is a difficult problem. Fortunately, human schedulers can be very adept at generating high-quality solutions, usually allowing all requests to be serviced (although not as fully as originally requested). However, this process is time-intensive and requires highly trained, experienced individuals, and the demands placed on them only increases as demand intensifies over time.

This situation provided an opportunity for an automated scheduling tool to take some of the burden off these schedulers. Additionally, such a tool can provide the currently unavailable possibility of running "what-if" scenarios to assess the impact of a potential event or mission change rapidly enough to be useful. For example, when a vehicle emergency is declared for even one of these satellites, schedulers must quickly reshuffle the communications plan to maximally support both the distressed vehicle and the high-priority tactical missions it supports. Balancing these priorities effectively is known in the industry as "deconfliction."

Stottler Henke in conjunction with the US Air Force has developed MIDAS (Managed Intelligent Deconfliction and Scheduling) -- a tool for rapidly scheduling and deconflicting AFSCN satellite communication requests. MIDAS is based on *Aurora*, Stottler Henke's intelligent resource scheduling platform.

Aurora is designed for modification / adaptation to vastly different domains, one aspect of this is the flexibility of the user interface, see Figure 2.



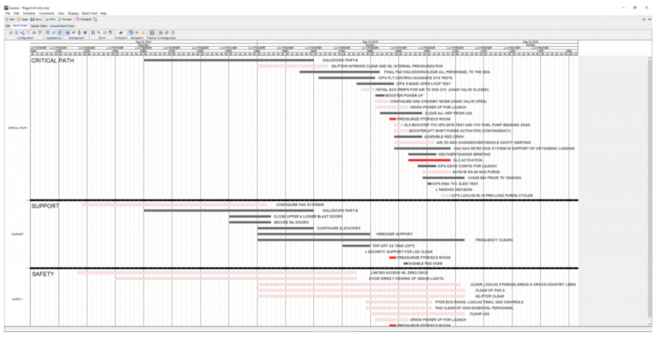
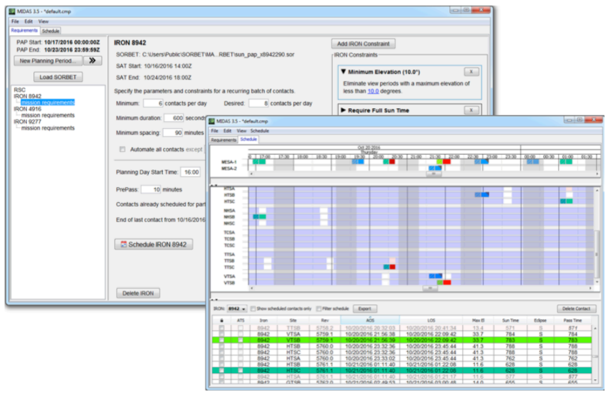


Figure 2. Aurora's Adaptable User Interface

MIDAS leveraged this aspect of Aurora to create a modified user experience. For example, Figure 3 shows one version of the interface adapted for satellite scheduling.

Before MIDAS, deconfliction could only be as fast and as good as the expert humans working at AFSCN, which was normally eight or more, now it can be done in less than hour, and usually in less than 15 minutes.

Prior to MIDAS the human schedulers manually checked every schedule request received, with the result being that approximately half of all requests required adjustment to remove conflicts. MIDAS now automates all or a vast majority of this. MIDAS accomplishes this with a two-stage process that first shuffles tasks within their defined constraints before carefully applying a user-definable set of business rules that allow certain constraints to be relaxed when necessary.



PAP Start: 10/17/2016 00:00:00Z
PAP End: 10/23/2016 23:59:59Z

IRON 8942
SORBET: C:\Users\Public\SORBET\MA...RBET\sun_pap_x8942290.sor
SAT Start: 10/16/2016 14:00Z
SAT End: 10/24/2016 18:00Z
Specify the parameters and constraints for a recurring batch of contacts.
Minimum: 6 contacts per day Desired: 8 contacts per day
Minimum duration: 600 seconds
Minimum spacing: 90 minutes
Automate all contacts except
Planning Day Start Time: 16:00
PrePass: 10 minutes
Contacts already scheduled for part
End of last contact from 10/16/2016

Eliminate view periods with a maximum elevation of less than 10.0 degrees.

IRON: 8942

	ATS	Iron	Site	Rev	AOS	LOS	Max El	Sun Time	Eclipse	Pass Time
		8942	TTSB	5758.2	10/20/2016 20:32:03	10/20/2016 20:41:34	13.4	571	S	571
		8942	VTSA	5759.1	10/20/2016 21:56:38	10/20/2016 22:09:42	33.7	784	S	784
		8942	VTSB	5759.1	10/20/2016 21:56:39	10/20/2016 22:09:42	33.7	783	S	783
		8942	HTSB	5760.0	10/20/2016 23:32:36	10/20/2016 23:45:44	41.3	788	S	788
		8942	HTSC	5760.0	10/20/2016 23:32:36	10/20/2016 23:45:44	41.3	788	S	788
		8942	HTSA	5760.0	10/20/2016 23:33:02	10/20/2016 23:45:44	41.3	762	S	762
		8942	HTSB	5761.1	10/21/2016 01:11:40	10/21/2016 01:22:08	11.6	628	S	628
		8942	HTSC	5761.1	10/21/2016 01:11:40	10/21/2016 01:22:08	11.6	628	S	628
		8942	HTSA	5761.1	10/21/2016 01:11:40	10/21/2016 01:21:17	11.6	577	S	577
		8942	GTSB	5762.0	10/21/2016 02:49:53	10/21/2016 03:00:48	14.0	655	S	655

Figure 3. Aurora User Interface Adapted for satellite scheduling

MIDAS provides a familiar, user-friendly interface modeled on legacy Electronic Schedule Dissemination (ESD) systems to facilitate comparison and to allow users to switch from one interface to the other with relative ease. It runs on inexpensive consumer hardware and communicates with legacy systems via a well-defined plain-text file format: raw scheduling requests are imported to MIDAS, and scheduling results can be exported back to legacy tools. MIDAS is useful for rapidly deconflicting real-world scheduling requests as well as, applications to planning (what-if scenarios) and training.

Again, the Air Force needed an automatic, intelligent software system that could quickly schedule and deconflict the communications requests from separate satellite operations centers (SOCs). Without MIDAS, the scheduling and deconfliction process requires excessive manpower from highly trained and highly skilled operators, each requiring almost a full year of training with a high washout rate. There was no way to quickly assess the impact of outages, vehicle emergencies, attacks, or possible changes to the communications systems. These very difficult, complex, and challenging problems inherently lead to inconsistencies in the schedules previously produced.

2. TWO-STEP SOLUTION

The solution is a two-step process:

1. Bottleneck scheduling,
2. Business rules deconfliction.

MIDAS includes a single-pass scheduling algorithm, called *bottleneck scheduling* that minimizes inter-support conflicts while obeying all user-specified constraints.

The chronological order in which tasks are considered for scheduling is the schedule processing order. Attaining a good schedule processing order for tasks is critical to building a near-optimal schedule during bottleneck scheduling. A human expert uses heuristics when deciding the order in which to review tasks. Some schedulers, for example, tend to look at Low-Earth-Orbit (LEO) contacts before High-Earth Orbit (HEO) or Geosynchronous (GEO) contacts. MIDAS mimics this behavior. The reasoning behind this decision is that LEO satellites have comparatively short visibility windows, significantly limiting the temporal flexibility of their contacts. By getting them out of the way early, a scheduler ensures that the contact gets the resources it needs while leaving the more flexible tasks for later, when the resources have already been partially allocated. If one waited until the end of scheduling to allocate these inflexible tasks, one would be left with very few (or no) alternatives if the required location is taken. Another way to say this is that, by scheduling inflexible tasks first, we keep the maximum amount of flexibility in the schedule at each step.

Bottleneck scheduling includes bottleneck avoidance using a similar heuristic, attempting to schedule the least flexible tasks before the most flexible tasks. Flexibility is defined using several dimensions: temporal flexibility (like the LEO-before-HEO approach), the degree of contention for resources in that time window, and the current state of tasks that have already been scheduled. These three considerations are automatically entailed in the predicted usage calculations for finding bottlenecks. The bottleneck avoidance algorithm involves a Preprocessor to derive a global perspective by determining which resources are bottlenecks (most overly contended-for) and at which times. This is explained more fully in [1] but very briefly, this involves "spreading" each request pseudo-probabilistically across all resources that it might use. (E.g. if a support request needs one of two antennas it is pseudo allocated 50% to each one and similarly the request's needed minutes are spread across the full possible time window). The Prioritizer uses this information to put requests that need the most overly-contended-for resources at the most overly contended for times at the front of the queue to be scheduled first. Then MIDAS uses the bottleneck information to make resource and time window selections to avoid the worst bottlenecks by making the assignment which most reduces the bottleneck problem. That is, in making this local decision it considers the global perspective.

Bottleneck avoidance solves about half of the conflicts but the remaining ones are typically unsolvable without relaxing some aspect of the requests.

The second step of the process, business rules deconfliction, iteratively examines each remaining conflict, and makes suggested changes to one or more support requests. For example, a specific support may request 10 minutes of preparation time before the support will actually commence. The scheduler may know that this constellation's manager will accept 5 minutes, if there is no other choice. The suggested change to that manager is to reduce his preparation time to 5 minutes. Other changes relate to moving the support out of its requested time window or to a different site or replacing ground support equipment with alternatives or even dropping certain hardware requirements all together. Some of these changes are more suggestable if the other satellite in the conflict is from the same constellation. The scheduler annotates the schedule with symbols and notes for the suggested change, appending his initials. With dozens of constellations, and each constellation having dozens of these rules of thumb, there are hundreds of undocumented rules that the expert schedulers used to resolve effectively all the remaining conflicts. Within each set of rules, there are preferences for which to use before others. Combinations and *domino* effects (e.g. solving a conflict by creating another, then solving that one) had to also be considered. This knowledge was elicited and implemented in constellation-specific, user-editable rule bases which were incorporated into Aurora's Postprocessor. The application of each rule also made the necessary note annotations and appended the software's initials. More details are found in the sections below.

3. BUSINESS RULES DECONFLICTION OVERVIEW

Schedulers frequently (tens to hundreds of times a day) encounter situations where conflicts cannot be solved without modifying the constraints of the original request – they must "bend the rules." These modifications must be reviewed and accepted by the user (satellite operations center) before they can be included in the final schedule, so it is important that the scheduler have a high degree of certainty that their suggestions will be accepted. This fact lead to the concept of business rules deconfliction – archetypal strategies that describe the different ways in which constraints can be modified – and how they can be orchestrated to solve even complex collisions between requests. There are a limited set of business rules. For example:

1. A support may be given less setup time than the user has requested.
2. The support can be moved temporally to another side
3. The support can be moved to another station altogether.

Long supports can (and often must) be broken up and handed off from one station to another. Each of these business rules takes a variety of parameters that describe when and how much the strategy can be applied based on the type of support, the user, and who they are in conflict with. Many conflicts can be solved by simply iterating through all conflicted supports and applying business rules in order until a solution is found. However, applying the rules in combination (e.g., combining a less severe rule with a more severe rule, rather than applying the severe rule alone) will often yield more acceptable results. And in some cases, a solution can only be reached by considering cascading combinations of business rules. Fortunately, only certain rules make sense in combination, so it is possible to consider the impact and application of each pair.

4. BUSINESS RULES

Knowledge elicitation (KE) was utilized to capture expert schedulers' experiential knowledge in order to develop the business rules, for example, which constraints can be relaxed in what situations and to what extent. The KE led to the capture of these rules in a concrete, user-definable format – as a set of "business rules." This solution kept the power in the hands of scheduling system administrators. The decisions of which constraints to relax are delicate ones, and the correct action in a given situation may alter over time as mission objectives or other factors change, so making the rules explicit and controllable has proven to be highly desirable and useful.

It was clear that what business rules should be applied and in what order varied among different groups of satellites, based on whether they were operated by the same organization or were essentially identical amongst themselves. So, for convenience the business rules are stored in order of most preferred to least preferred within the group (often called family) of satellites they correspond to. The groups range in size from 1 to several dozen.

Business rules are applied after bottleneck scheduling is complete and only to the extent necessary to solve the conflict (e.g., if a prepass can be shortened to a minimum of 5 minutes but only needs to be shortened to 7minutes to resolve the conflicts, it will shortened to 7 minutes). There are several ways to apply deconfliction strategies. The user has the option to apply business rules to a single task or to all conflicted tasks in the current time period. Rules are ordered such that those that cause the least "damage" are attempted first. Shortening a task's prepass is often one of the first steps a scheduler will take and with fairly accurate foreknowledge of what will and will not be an acceptable concession from the user. Because of this "shorten prepass" is among the first business rules attempted for many families.

Listed below is a sampling of the single business rule deconfliction strategies that MIDAS currently supports.

Shorten Prepass

- to a minimum of (minimum_duration) if (condition)
  - minimum_inter_family_duration = minimum duration if turning around from an Inter-Range Operating Number (IRON) in another family
  - minimum_intra_family_duration = minimum duration if turning around from an IRON in the same family
  - minimum_ats_duration = minimum turn around for an automated track support
  - hard_minimum = absolute minimum that will not be violated
  - condition = which minimum applies

Negative Turnaround

- allow a negative turnaround with a task in the set (irons)
    - irons = family of IRONs with which this task can have a negative-turnaround

Relax Schedule Constraints

- Remove NO# constraint x
- Relax NO(n) constraint to a minimum of NO(m)
    - n = 2..∞
    - m = 1..n-1
- The NO(n) constraint requests that n supports in a row should not be at the same station

Redundant Equipment

- Remove (secondary_equipment) if (primary_equipment) is available
    - secondary_equipment = 1 or more equipment type
    - primary_equipment = 1 or more equipment type

Handoff

- Handoff task from (source_stations) to (target_stations) with minimum block size (minimum_block) and (overlap_duration) overlap
    - source_stations = allowed stations for the prior portion of the task
    - target_stations = allowed stations for the subsequent portion of the task
    - minimum_block = the minimum duration of each resultant task after splitting
    - overlap_duration = the desired amount of overlap between the prior and subsequent task

Move Out of Window

- Allow task to move (start_change) min earlier than the beginning of its requested window
    - start_change = maximum number of min before the window start
- Allow task to move (end_change**)** min later than the end of its requested window
    - end_change = maximum number of min after the window end
- Allow task to move +/- (change) min out of its requested window
    - change = maximum number of min outside of the requested window

Shorten Task Duration

- Shorten task duration up to (minimum_duration)
    - minimum_duration = [### min OR ###% of original]
- Shorten task duration by moving start time (maximum_start_change) later
    - maximum_start_change = the number of min forward that the start can be moved
- Shorten task duration by moving end time (maximum_end_change) earlier
    - maximum_end_change = the number of min backward that the end can be moved

When applying business rules automatically to all conflicted tasks in the currently visible region, rather than applying all business rules to one task before moving on to the next task, we use an iterative algorithm to avoid highly damaging one task when a lower damage change to another task may have resolved the conflict. The algorithm, therefore,

- traverses all conflicted tasks, applying the lowest damage business rule first; then
- traverses a second time, applying the second lowest damage business rules until all conflicts are resolved or until we have tried all rules to all conflicted tasks.

There are situations when multiple business rules need to be applied to a single task. A single business rule may not be enough to resolve a conflict or by applying several business rules in concert to a small degree, we may be able to achieve a more desirable result than by only applying a single business rule to a greater extent.

In general, two business rules are combined by applying the first rule to some maximal extent defined by the business rule, applying the second rule, and then relaxing the first rule. In practice, it is not possible to generalize the algorithm for combining any two business rules, and only certain combinations make sense. For this reason, we determined a manageable fixed set of combinations that are supported. Rule combinations are generally attempted after each of their component rules have been attempted singly.

5. BUSINESS RULES USE PROCESS

Once the business rules have been defined for each family they can be employed in multiple ways and an initial deconfliction process has been developed that uses them in several ways. As mentioned previously, the first step is to apply bottleneck scheduling to solve conflicts by shuffling the scheduling within the parameters of each support request. Then, usually the next step is to automatically try to solve conflicts by applying a single business rule to each task, separately. Next, multiple business rules are applied to each task and each conflicting pair of tasks to solve each conflict while still relaxing the constraints for the least number of tasks possible and using the least damaging combination of rules for each task.

Even after following the above process, a fair number of conflicts usually remain. For hard-to-solve conflicts, human schedulers were observed "preparing" a location in advance of a move to solve a difficult conflict. For example, it may be the case that the conflict between two tasks cannot be resolved because although either or both of the tasks can be moved (within its constraints or using a business rule), all the possible destination locations are full. In this case a human scheduler will often work on one of these possible destination areas and move the supports that are there (either within their constraints or using business rules) to make room for one of the tasks in the original hard-to-solve conflict. So, the scheduler moves the other tasks first, then moves the support into the hole they created. This is exactly equivalent to doing the same operations in the opposite order – first solving the original conflict by moving one of the tasks to a new location where there is no room for it and then resolving the new conflicts that were created.

When viewed from this perspective, a *domino* phenomenon can be seen, resolving one conflict by creating another and then solving it. Said another way, moving one support forces another support to be moved. The above is a description of a single level of domino, but any number is possible and going to two, three, or four levels of dominoes is fairly common. When using the domino method, the newly created conflicts, may be solvable with moves that are allowed by the constraints of the support, so that these should be tried first. But it is also often the case that solving the subsequent conflicts requires the use of business rules also. Typically, using the domino method with a depth setting of 2, 3, or 4 is the last step in the automated deconfliction process. Then, if there are any remaining conflicts, human schedulers resolve them.

A final very useful feature is termed by the users as "self-heal." Sometimes the reason for applying a business rule has changed. E.g., it may be the case that to resolve a conflict, one of the supports involved in the conflict was moved outside of what would be allowed by its parameters so a business rule was invoked. If later the other support involved in the original conflict is deleted or moved for other reasons such that the moved support could be moved back to being within its original parameters, then this automatically occurs. (I.e., if the reason for applying a business rule changes and is no longer valid, the effect of the business rule is reversed.)

The process of automatically employing business rules in addition to mimicking the human deconfliction process, must also follow the *human annotation process*, since it is intended to perform within the current work flow. This fulfills two functions.

1. Verify that the change is approved by the satellite operations center (SOC) before it appears in a published schedule.
2. Provide an audit trail of who made changes outside of the normal parameters. In the case of the automatic application of a business rule, the software did.

6. RESULTS

The business rules were entered for the entire set of about 2 dozen families and applied to the deconfliction task for several different 24-hour schedules. Typically, circa 600 supports needed to be scheduled each day. Although there were fairly wide variations, about half of the supports started out in conflict. From this starting point of about 50% deconflicted, bottleneck scheduling typically solved half of the conflicts leaving 25% to be deconflicted by the *business rule deconfliction* processes described herein.

- Applying single instances of business rules solved another 10% to arrive at an approximately 85% deconflicted schedule.
- Applying multiple business rules (but no dominos) generally brought that up to 90%.
- Applying the domino method using a depth of 4 brought that total up still further to around 97%.

MIDAS has proven useful for other analysis. For example, there was a seemingly compelling cost-saving case for shutting down two sites. To justify keeping them open, MIDAS was used to quickly schedule the previous month's worth of requests and show the severe mission impacts that would result from these closures.

During real or training emergencies, MIDAS allows much quicker impact determination and new optimized schedule creation.

More abstractly, MIDAS has been an impressive application demonstrating how we can replicate human thought processes in a very difficult domain.

Many of the intelligent scheduling algorithms developed as part of the MIDAS effort have been incorporated back into our commercially available Aurora intelligent scheduling tool, used by Boeing, Pfizer, General Dynamics Electric Boat, NYU, and others. For example, NASA KSC leverages Aurora to make scheduling and processing of space vehicles more efficient; from multi-year projects down to the final launch.

7. CONCLUSION

Before the MIDAS, deconfliction could only be as fast and as good as the expert humans working at AFSCN. Satellite Control Network scheduling largely consists of resolving disputes between competing support requests (and other tasks such as maintenance). About half of the conflicts can be solved by shuffling the requested supports within the constraints supplied with the requests. The other half require some degree of relaxation of the constraints. Using a representation of parameterized business rules, families of satellites, and a preference listing of the rules and parameters for each family allows the MIDAS software to automatically resolve the large majority of remaining conflicts. This greatly saves the labor required for deconfliction and allows more accurate study of future loading (and associated required resources) and more accurate response to what-if questions relating to the impact of failed resources and required emergency supports.

Over a thirty-year period, dozens of organizations have worked on this specific problem and the Air Force had previously invested tens of millions of dollars to develop various solutions, but all of them were considered operationally unacceptable (primarily because the relaxation rules had never been elicited before). MIDAS has solved this problem while demonstrating a 20-fold improvement in the time required to deconflict a 24-hour schedule; while now allowing for rapid re-scheduling under emergency conditions and the evaluation of multiple scenarios in a timeframe that makes the results useable.

8. REFERENCES

[1] Stottler, Dick and K. Mahan, "Automatic, Rapid Replanning of Satellite Operations for Space Situational Awareness (SSA)," presented at the Advanced Maui Optical and Space Surveillance Technologies Conference, 2015.

[2] Stottler, R., Mahan, K., and Jensen, R., "Bottleneck Avoidance Techniques for Automated Satellite Communication Scheduling," Proceedings of the Infotech@Aerospace 2011 Conference, Vol.1, AIAA, Reston, VA, 2011.

9. BIOGRAPHY

Richard Stottler *co-founded Stottler Henke in 1988 as a software company dedicated to providing practical solutions to difficult problems by skillfully drawing upon a large repertoire of artificial intelligence technologies. Under Dick's leadership, Stottler Henke has grown steadily and profitably into a 40-person research and software development company with distinctive expertise in intelligent tutoring systems, intelligent simulation, automated planning and scheduling, and intelligent knowledge management. Dick provides technical leadership in the design and development of intelligent tutoring systems, intelligent planning and scheduling systems, and automated design systems. He combines a strong applied research record in artificial intelligence with practical experience in rapid and efficient knowledge engineering. He has led the development of intelligent tutoring systems that encode the expertise of instructors to provide practice-based learning and automated evaluation of student performance in subject areas spanning navy tactics; army tactics, command, and control; sonar data analysis; astronaut training; helicopter cockpit operations; and battlefield emergency medicine. He also led the development of intelligent planning systems for NASA space shuttle missions and aircraft assembly and automated scheduling for the International Space Station. He also led the development of intelligent systems that encode and apply human expertise and experience to automate the design of manufacturing processes and aircraft systems to lower manufacturing costs, increase product quality, or achieve demanding performance criteria. Dick has written or presented more than two dozen papers and articles for publications such as the proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). He received his BS in engineering from Cornell University and his MS in computer science (artificial intelligence) from Stanford University.*

Robert Richards *received a Ph.D. in Mechanical Engineering from Stanford University. Dr. Richards is managing and has managed multiple projects for both commercial and government clients, including various intelligent scheduling implementations. Dr. Richards is the Principal Scientist and Manager of Stottler Henke's Pfizer project for scheduling pharmaceutical packaging plants, the end product is Aurora-ProPlan. Dr. Richards also leads the intelligent scheduling project for General Dynamics Electric Boat for improving the throughput of submarine production. Dr. Richards has also worked on and continues to work on various projects spanning a wide range of research and application area interests, including: training system development; applying automation and artificial intelligence techniques; and decision support tool development for life-critical situations. Dr. Richards has publications in all of these domains.*