IN-15
70377
p.13

moranduum 4306

# Techniques for
# bility of OPSMODEL
# ulation Software

NASA Technical Memorandum 4306

# User Modeling Techniques for Enhanced Usability of OPSMODEL Operations Simulation Software

William T. Davis
*Langley Research Center*
*Hampton, Virginia*

National Aeronautics and Space Administration
Office of Management
Scientific and Technical Information Program
**1991**

## Summary

The PC-based OPSMODEL operations software for modeling and simulation of space station crew activities supports engineering and cost analyses and operations planning. With its top-down modeling structure, the level of detail required in the modeling data base can be limited to being commensurate with the results required of any particular analysis. To perform a simulation, a resource environment consisting of locations, crew definition, equipment, and consumables is first defined. Activities to be simulated are then defined as "operations" to be performed, and they are scheduled as desired. In addition, these operations are defined within a priority structure of 1000 levels. The simulation on OPSMODEL, then, consists of the following: **user-defined, user-scheduled operations executing within an environment of user-defined resource and priority constraints**.

Techniques for using operation priority assignments to realistically model a representative daily scenario of on-orbit space station crew activities is discussed. The techniques can cover all daily activities from morning awakening and breakfast, through the workday, and into the evening activities and sleep. The large number of priority levels available allow priority assignments to be made commensurate with the level of detail (or lack of detail) necessary for the particular questions being studied by the simulation.

Also addressed are the problems encountered with realistic modeling of the day-to-day work carryover (work scheduled for a particular day but not completed due to resource or priority conflicts). Several different solutions to this problem have been examined and will be described. Recommendations for further improvements for more realistic daily scenario simulation will be discussed.

Finally, the use of conditional task execution based on counter status, a feature recently added to OPSMODEL, will be addressed.

## A Brief Description of OPSMODEL

OPSMODEL (ref. 1) is a PC-based flexible software tool developed by the Computer Sciences Corp. for NASA Langley Research Center that allows the user to realistically model and simulate the operational activities of a space station. Execution of activities can be prioritized, and interruption of currently executing lower priority activities is allowed. Its top-down modeling structure allows the user to control the level of complexity of model definition while limiting the effort expended for data base population to only what is necessary. OPSMODEL has the capability for probabilistic modeling, utilizing both commonly used statistical functions as well as unique user-defined ones. OPSMODEL has three major parts, as shown in figure 1 a relational data base, a time-event simulator, and comprehensive data output.

The data base requires three groups of input data: (1) a description of the space station physical configuration and a definition of resources (crew, equipment, and consumables) available, (2) a description of the operations/tasks to be executed in the simulation, and (3) the scheduling of these operations/tasks.

The simulator requires execution definition (start/stop times, number of repetitions, etc.) and selection of appropriate real-time monitoring options.

In addition to real-time simulation monitoring, three types of output data are available. The first data output type shows engineering performance. It consists of execution data for each operation/task performed, crew time allocations for each operation/task (including crew idle times), space station status information (including equipment and consumable use), and other summary data.

The second output data type shows cost performance data organized according to resource, work breakdown structure (WBS), and task. Resources (crew, equipment, and consumables) used in the various operations/tasks can be assigned cost factors. A task can be assigned an appropriate charge number (WBS number), and all resources used in that task will be charged to that WBS number. When a simulation run is complete, the cost factors for each resource are multiplied by simulation-time usage data, filed, and accumulated as costs in the WBS.

The third type of output is most useful to those interested in operations planning and analyses. These data are derived from a time-tagged event log that is recorded as the simulation is being run. This log can be viewed directly (via CRT or hard copy), or specific data can be examined in various graphical or tabular outputs related to particular operations/tasks, crew members, equipment, or consumables.

The data required to define the work done on, in, and by the space station are described in terms of operations and tasks. A task is the smallest element of work used in OPSMODEL. Tasks may be connected to form operations. An operation may consist of only one task or as many tasks as necessary to define the work to be done.

A useful method of understanding the workings of operations and tasks is to think of each operation as having an "activation entity." This entity starts at the beginning of an operation, works its way through the consecutive tasks, and ends at operation completion. The simulator controls the movement of this entity and thereby controls the simulation and which tasks are active at any given time.

A task is the basic building block for all operations in OPSMODEL. The OPSMODEL sees a task in terms of a task diagram (TD) as shown in figure 2. The TD has two inputs: the external input where the activation entity normally enters and the internal input where the entity reenters after having left via the internal output. The TD has three outputs: the normal output taken after the task is completed, the alternate output taken when conditional logic requires an alternate path, and the internal output used when the given work is expanded in terms of subtasks. Subtasks use the same TD template and allow more definition to the original task. Normally the entity acquires resources to be used by the task, does the task work, releases the resources, and exits. If subtasks are present, the switch shown in figure 2 is switched to the internal output, the subtasks are performed, and the entity returns to the main task via the internal input. The entity then releases the nonconsumable resources it acquired at the beginning of the task and exits. The number of levels of subtasks is not constrained.

The versatility of OPSMODEL covers a wide range of modeling capability and includes such things as

1. Daily crew activities (eat, bathe, etc.)
2. Onetime events
3. Equipment failure and repair
4. Emergencies
5. Specific crew work assignments
6. Space station configuration
7. Extravehicular activity (EVA)
8. Resource definition and depletion (includes crew, equipment, and consumables)
9. Space station maintenance and support
10. Sleep
11. Probabilistic modeling
12. Space station environment (i.e., day/night cycles, communications coverage, etc.)

## Priority Scenario Development

Probabilistic modeling capabilities may be introduced into an OPSMODEL simulation by the application of probability functions to any of three appropriate parameters: (1) any task duration, (2) the time of the first execution of a task, and (3) the repetition rate of a repetitively scheduled task. When these probabilistic parameters are applied, scheduling and resource conflicts occur which are best resolved by the use of priority assignments. As initially delivered, the OPSMODEL software had the capability of assigning an operation only two levels of priority, either high or low. Preliminary exercise of the software indicated an expansion of the priority capability was needed to make effective use of the probabilistic modeling features included in OPSMODEL. A priority capability of 1000 levels was implemented.

As a part of acceptance testing, it was necessary to create comprehensively populated data bases to effectively exercise the software. To avoid unrealistic priority operation preempts (such as dinner interrupting sleep) and to more fully utilize the synergistically powerful features of probabilistic modeling and multiple priority levels on these data bases, a priority scenario technique for operations in OPSMODEL was developed. For OPSMODEL, a priority scenario is defined as a graphical method of displaying the priorities of operations (or operation types) as they relate to other operations (or operation types) and to time. The priority scenario is a useful tool when developing simulation data base models. Furthermore, the more complex and involved the simulation is, the more useful is the priority scenario. The initial 1-day operations priority scenario used for OPSMODEL is shown in figure 3. Priority levels from 0 to 999 are represented along the ordinate, and time is indicated along the abscissa. This priority scenario is used to model and simulate a typical day's activity on a space station and reflects a requirement of OPSMODEL software acceptance testing.

The usefulness of a priority scenario is not specifically from the absolute priority levels assigned, even though specific priority levels must be entered into the simulation data base. The usefulness of a priority scenario derives from its ability to explicitly show at a glance the priority level and time (of day) relationships between all operations (or types of operations).

In addition, the effort and tedium associated with populating a large data base is made significantly easier if a plan (priority scenario) is used for priority assignments.

The rationale for priority assignments may vary from one simulation data base to another, resulting in the establishment of different priority scenarios. A discussion of rationales for the priority scenario of figure 3 will illustrate the manner and process by which they are determined. The priority scenario of figure 3 was developed for a typical space station workday. Relative priorities are assigned, allowing the desired operations to proceed in a rational sequence when conflicts in time and resource allocation occur. For example, if the sequential activities of wake-up and breakfast tried to execute simultaneously (or with any undesired overlap), a higher priority for wake-up properly allows it to execute first. Similarly, a preshift work conference, in order to execute before shift work operations, should have a higher priority. Also, to assure that the astronauts eat lunch, the lunch activity would also have a higher priority than the work operations. The shift termination conference operation has a higher priority than the work, dinner, or rest and recreation (R & R) operations, since it was assumed more important than each of these.

Priority assignments can also reflect subtle differences in simulation assumptions. For instance, if the astronauts were only expected to work during their work shifts, then the priority for R & R could be higher than for work time to assure unfinished work was not accomplished during R & R time, but carried over to the next day. On the other hand, the priorities could be adjusted to allow the astronaut to return to unfinished work after dinner by setting the R & R priority to less than the work time priority.

Emergencies are assumed to always have the highest priorities and would preempt any resources needed to handle them. However, a noncritical failure may be assigned a lesser priority commensurate with the activities that it is and is not allowed to preempt.

Extravehicular activity (EVA) operations may extend to nearly 8 hours. They should have less priority than the preshift work conference but more than lunch and, in particular, more than the shift termination conference.

Sleep has a higher priority than all operations except emergencies. If, however, a particular operation was required to be executed during normal sleep time, it could be given higher priority than sleep, but less priority than emergencies.

Priority scenarios are a useful organizational tool for operations simulation. Priority scenarios can be tailored for specific simulation objectives by using alternative priority levels for appropriate operations. In addition, once developed, the priority scenario facilitates large simulation data base population.

## The Work Carryover Dilemma

Work carryover is work that has been scheduled for a particular day but not completed because of resource or priority conflicts and must be carried over to the next day for completion. The dilemma occurs because once an operation is initiated, it will continue to attempt to complete, using any unscheduled time available, even though conditional logic which allowed its initiation may change to a state which would normally preclude its initiation. In other words, once an operation starts, it never rechecks its conditional logic again. The simulation analyst must allow for this software characteristic. Several ways to address this OPSMODEL simulation dilemma will be discussed.

The simplest manner in which to handle work carryover is to run only 1 day's simulation at a time. Examination of the output data products will reveal how much of the day's work was not completed. This information may be input to the next day's simulation and so forth until the required number of simulation days are accomplished. This process, while doable, can quickly become very inefficient.

A method with more merit can be illustrated by the priority scenario of figure 4 (a single modification to fig. 3). In this scenario, sleep is modeled to start at its normal time, but only lasts some nominal time (in this case, 2 hours). Work carryover is artificially completed in the pseudosleep period, where its accounting can be done if necessary. The simulation can be run for multiple days without encountering unrealistic execution of some types of operations. This method works particularly well for operations not requiring completion of scheduled execution time (R & R time is an example) or operations where a daily reset for execution initiation is mandatory (meals are a good example).

The final method which has been used to solve the operation carryover dilemma was derived to automatically accommodate work time carryover. This method requires the use of several OPSMODEL software operations to redefine the single original simulation operation. These operations are graphically represented in figures 5, 6, 7, and 8, showing pertinent operation parameters which must be defined to implement work carryover. A detailed explanation of this work carryover scheme follows. For illustrative purposes, we will assume that the work cannot be completed during the work shift and must be completed during the following day, that is, the next regular work shift. The operation shown in figure 5 controls the logical parameter SHIFTxON. Its logical status is 1 during the work shift, and 0 otherwise.

OPxx (fig. 6) has most of the parameters of the original simulation operation except the required duration, which is specified in OPxy (fig. 7). Both OPxx and OPxy are scheduled to execute at the required execution time of the original operation. OPxx sets a logical function, STRTDUROPxx = 1, when it begins execution. OPxy, since it contains the actual duration parameter, does a conditional logical HOLD ON: STRTDUROPxx = 0. This is necessary to ensure that OPxy does not begin execution without OPxx.

After leaving the top level task (the duration of a top level task is always zero) in OPxx, the execution of OPxx enters an endless loop between tasks 2 and 3 (i.e., the normal output of task 2 goes to the normal input of task 3, and the normal output of task 3 goes to the normal input of task 2). At the start of both tasks 2 and 3, conditional logic allows an alternate path escape from the endless loop. In task 2, the alternate path escape is accomplished if SHIFTxON = 0, that is, if the work shift (it typically could be 9 a.m. to 5 p.m.) is over. For task 3, the alternate path escape is accomplished if the logical function DUROPxx = 0. DUROPxx is the logical function set to 1 when OPxy begins and is reset to 0 when OPxy terminates. It therefore represents the duration of execution time required as defined for the original operation.

If we assume that the work shift ends before the operation duration ends (i.e., work carryover is required), then when the state of logical parameter SHIFTxON becomes 0, the execution of OPxx takes the alternate path escape from task 2 and proceeds to exit from OPxx execution.

Two pieces of equipment, TOKENxx and TIMERxx, are used and must now be discussed. In OPSMODEL, higher priority operations preempt needed equipment, and operations not executing release equipment. The operation OPxz, scheduled to execute from the beginning of the simulation to beyond the final execution of OPxx/OPxy, requires the use of TOKENxx and TIMERxx and has a priority of 1. OPxx also requires the equipment TOKENxx and, having a priority greater than 1, takes it from OPxz at its scheduled execution time. OPxz can no longer execute and releases the equipment TIMERxx. Now, OPxy, with a priority of 0, can be allocated TIMERxx and begin the duration timing. When OPxx completes first shift execution (by alternate path escape from task 2), it releases equipment TOKENxx, which is then available for OPxz resumption. For OPxz to resume, however, it must preempt the equipment TIMERxx from OPxy, which halts its execution and thereby stops the operation duration timing.

The next scheduled execution of OPxx and OPxy occurs in a similar manner when all necessary logical conditions have been met (in this case, it occurs the next day). There is a difference, however, in that the first sequence of the OPxx/OPxy operation (the work carryover) has never completed and will complete execution first thereby delaying the start of the second sequence until the completion of the first. When the required duration of the first sequence of OPxx/OPxy is complete, OPxy causes the state of the logical function DUROPxx to change from a 1 to a 0. OPxx, sequence 1, then terminates via the alternate escape path from task 3, and its second sequence commences. If the second sequence also fails to complete during its work shift, work carryover will occur again. In this manner, the work carryover is automatically accomplished until the overall simulation is complete, always doing the earliest sequences of the scheduled operation (OPxx/OPxy) first.

If all sequences of a carryover operation are not completed at simulation end, the duration of the original scheduled operation sequences (OPxx/OPxy) not accomplished can be determined. The completion of OPxz can be adjusted to a point in time beyond the final execution of OPxx/OPxy. When OPxz completes, the operation OPxy will execute for a time period equal to the unaccomplished, but scheduled, work of the original operation. This occurs because OPSMODEL always completes scheduled operations when conditions allow.

The duration of endless loop tasks 2 and 3 of OPxx deserves further comment. Task duration in OPSMODEL is specified in whole minutes. The simulation completion of one loop around the endless loop has a lower bound of 2 simulation minutes (1 minute for each task) and no pertinent (for this application) upper bound. Two otherwise identical endless loops were tested to determine the effects of different task times on a 4-hour operation duration. The first endless loop had simulation task times of 1 minute and 1 minute. The second endless loop had simulation task times of 1 minute and 1 hour. On an IBM PC-AT with math coprocessor, a wall clock execution time difference of less than 5 minutes was observed. The first case gave a more accurate simulation but at the expense of a slightly longer wall clock simulation run time, while the second case took less wall clock time at the expense of simulation time detail accuracy. The simulation analyst may adjust the endless loop task duration times as appropriate for each simulation.

The way OPSMODEL executes could be changed (a software modification) to eliminate the necessity of handling the work carryover dilemma in such a complex fashion as above. The execution of an operation in OPSMODEL would only take place after any logical conditions or criteria assigned to it are satisfied. However, as mentioned above, once an operation meets its conditions and criteria, it begins execution and never checks them again. Even if the operation is preempted, it does not check them again upon resumption. If the software could respond to changes in these conditions and criteria as an interrupt to halt execution, the simulation would be considerably simplified for work carryover instances. In addition, many other aspects of the simulation would become more realistic, such as operations based on the space station environment (i.e., daylight, communication links, etc.). The implementation of this change would significantly enhance the simulation realism and accuracy.

An alternative change would be to require an operation, once preempted, to recheck its conditions and criteria before resuming execution. Although not as comprehensive in enhancing accuracy and realism, this software modification would probably be much easier to implement.

## Conditional Operation Execution Based on Counter Status

As initially implemented in OPSMODEL, counters could be set to zero and incremented or decremented (by selected amounts) during a simulation. Counter status could then be determined by examination of the data output products at the end of the simulation. In addition, conditional logic execution based on the functional state of selected logical parameters was possible. A change was instituted into the OPSMODEL software to also enable conditional operation execution based on counter status. Conditional logic execution can be based on whether the counter status is greater than, less than, or equal to a particular count. Counters can be made to either count up or count down with a user-set increment. This capability gives greater flexibility and more realistic implementation options to the simulation analyst under a wide variety of simulation conditions. Some examples of enhanced OPSMODEL simulation capability resulting from this change are discussed below.

The most obvious use of conditional-counter-status operation execution is the generic requirement of executing task $b$ only after task $a$ has executed a user-specified number of times $n$. An example of this capability is shown graphically in figure 9. When the operation execution reaches task 2, it holds (based on the conditional logic) until the actual counter value $C$ is equal to $C_n$, the required number of times, $n$, that the specified task $a$ has executed. Task $a$ need not (but can) be a part of this operation. The counter, of course, is incremented once for each execution of task $a$. A related variation of this application would terminate an operation after the selected task/operation has occurred a specified number of times.

Not so obvious is the use of the new counter/conditional logic capability as a secondary simulation-time clock. This application is shown in figure 10. The execution of this operation and the start-up of the secondary clock can be controlled by a conditional logic hold for task 2. Once satisfied, the execution proceeds to the endless loop of tasks 3 and 4. Tasks 3 and 4 may each increment the counter. The duration time of each would usually be equal to 1, but can be selected as appropriate by the simulation analyst. Other operations are then able to use the counter status as an enabling gate for their own execution.

With the ingenuity of simulation analysts, further applications of the counter status/conditional logic capability can quite surely be found.

## Concluding Remarks

A brief description of the OPSMODEL operations simulation software was given. The desirability of, and the development of, a priority scenario technique was discussed and a representative daily priority scenario was given. A priority scenario is a graphical method of displaying a simulation priority plan. It shows the priorities of simulation operations (or types of operations) as they relate to other operations (or types of operations) and to time.

Realistic modeling of day-to-day unfinished tasks, called work carryover, was addressed. Several methods for handling this dilemma were discussed. A software change that would allow more straightforward application of the simulation modeling program OPSMODEL was discussed.

Finally, a recent software modification, the ability to use counter status as a basis for conditional logic decisions, was discussed. Examples of several methods for the application of this capability were examined.

## Reference

1. Davis, William T.; and Wright, Robert L.: OPSMODEL, an On-Orbit Operations Simulation Modeling Tool for Space Station. *Proceedings 2nd AIAA/SOLE Space Logistics Symposium,* Oct. 1988, pp. 94–103. (Available as AIAA-88-4732-CP.)

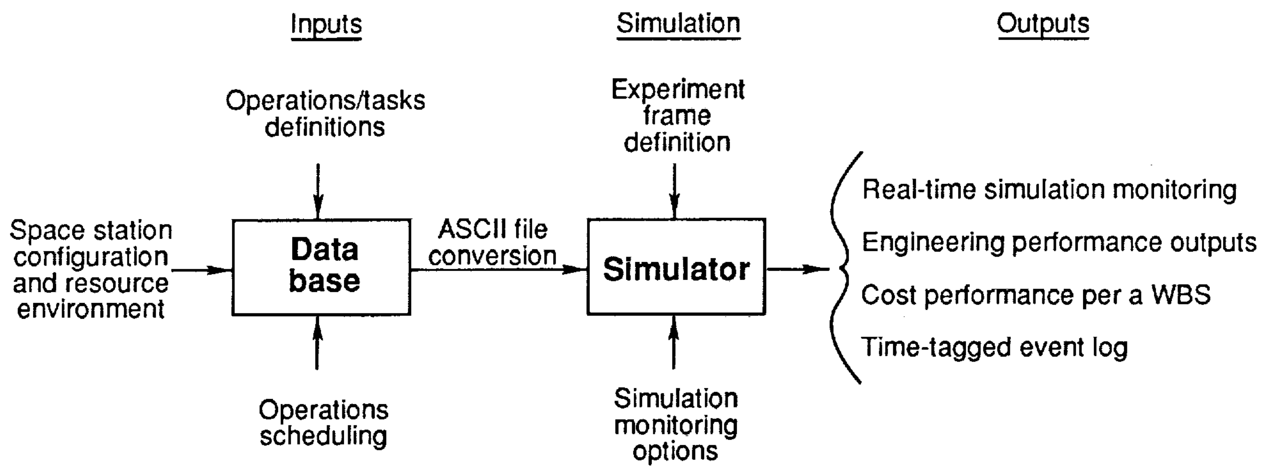

Inputs
Simulation
Outputs
Operations/tasks definitions
Experiment frame definition
Space station configuration and resource environment
Data base
ASCII file conversion
Simulator
Real-time simulation monitoring
Engineering performance outputs
Cost performance per a WBS
Time-tagged event log
Operations scheduling
Simulation monitoring options

Figure 1. OPSMODEL functional flow.

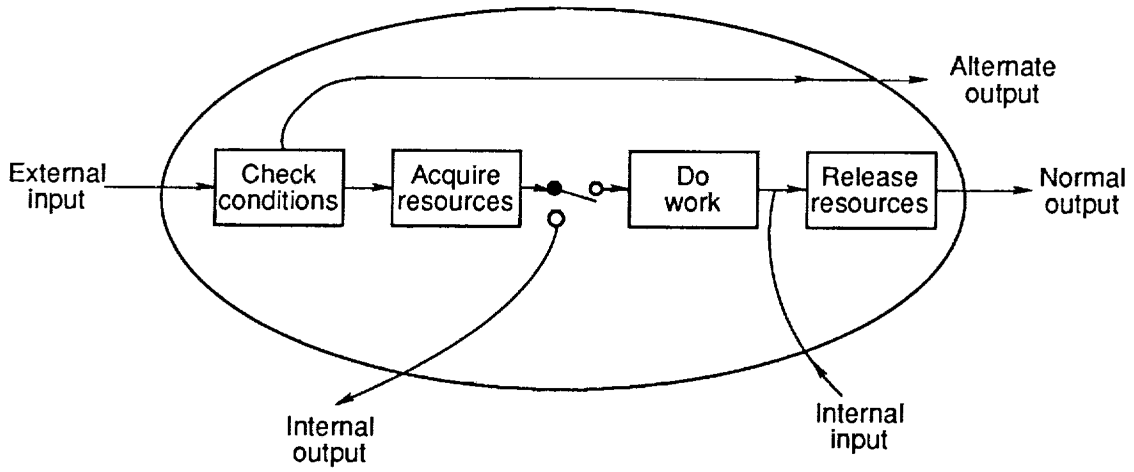


Figure 2. The OPSMODEL task diagram.

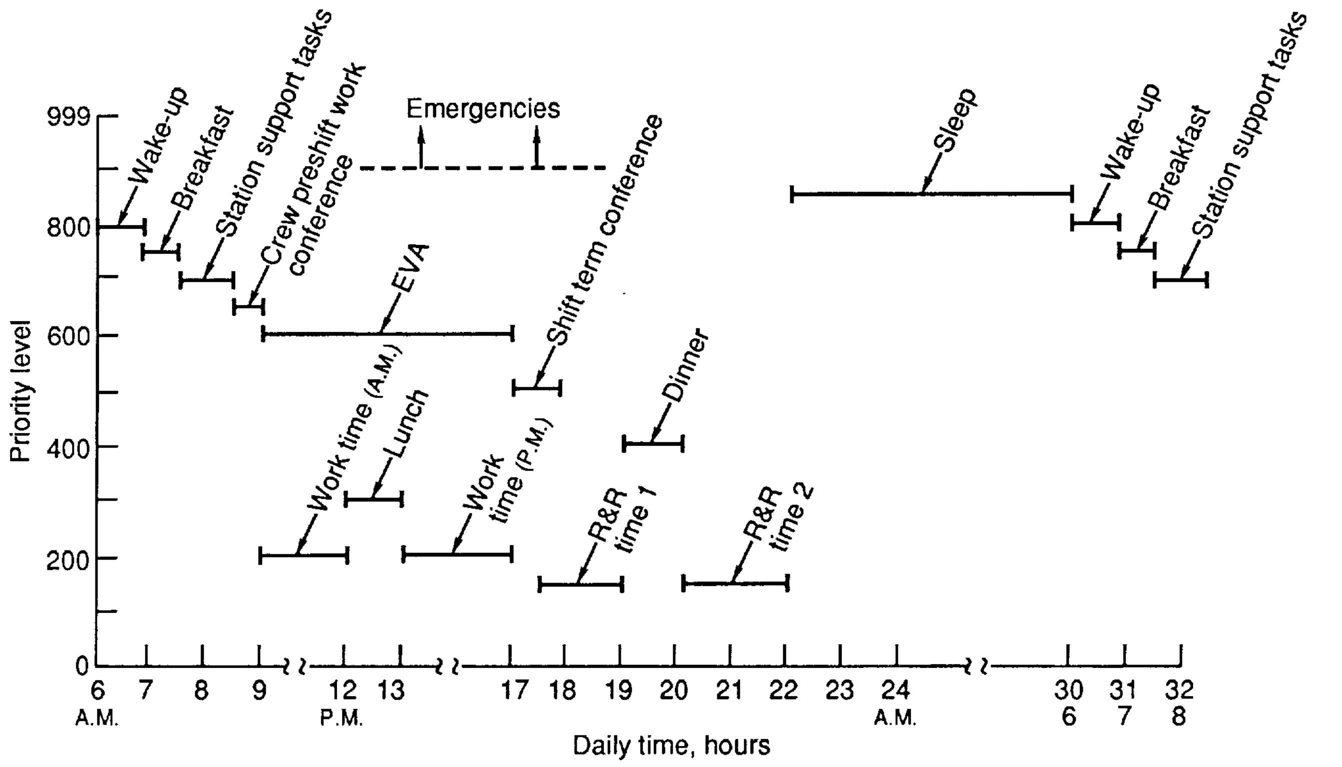


Figure 3. A possible OPSMODEL priority scenario.

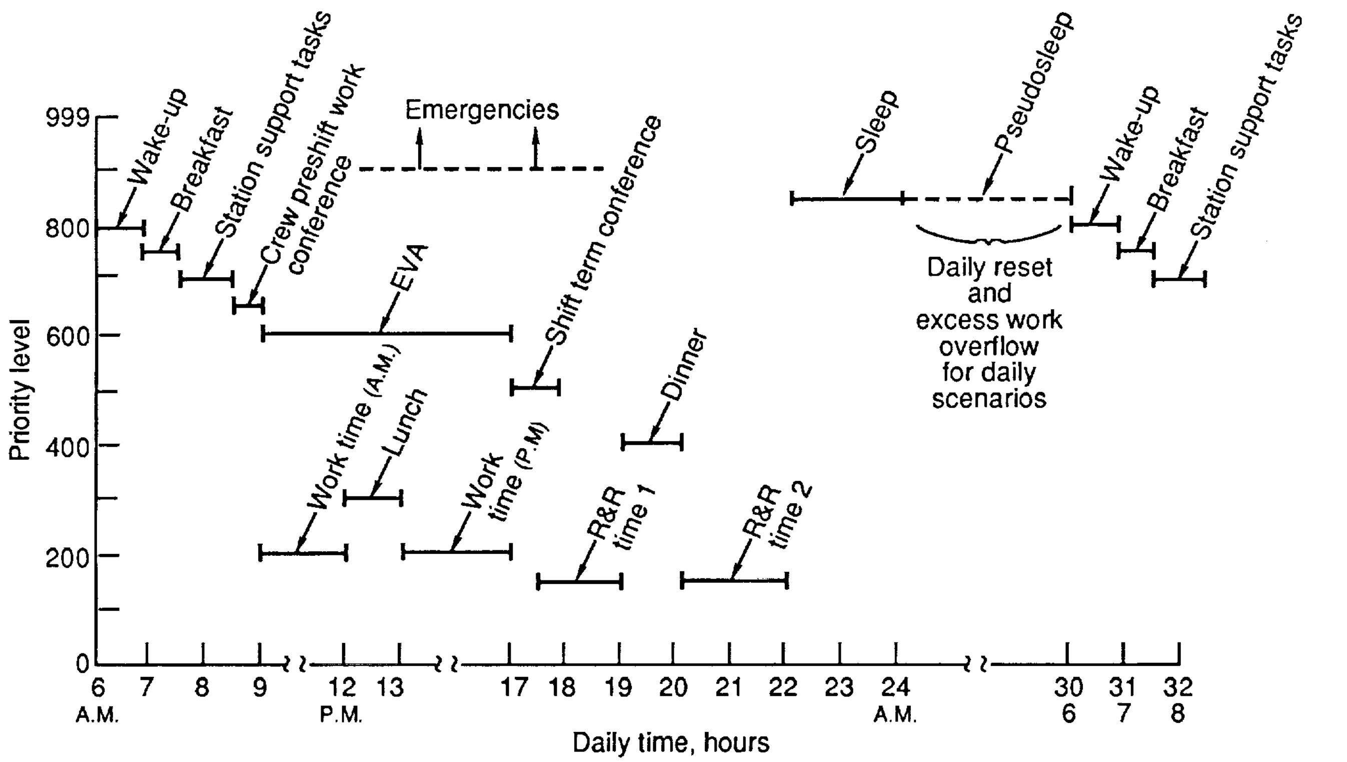
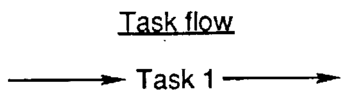


Figure 4. A possible OPSMODEL priority scenario with pseudosleep.

**OPERATION NAME: SHIFTxON**

Priority = 0
Schedule = As required to define work shift (typically 9 A.M. to 5 P.M.)

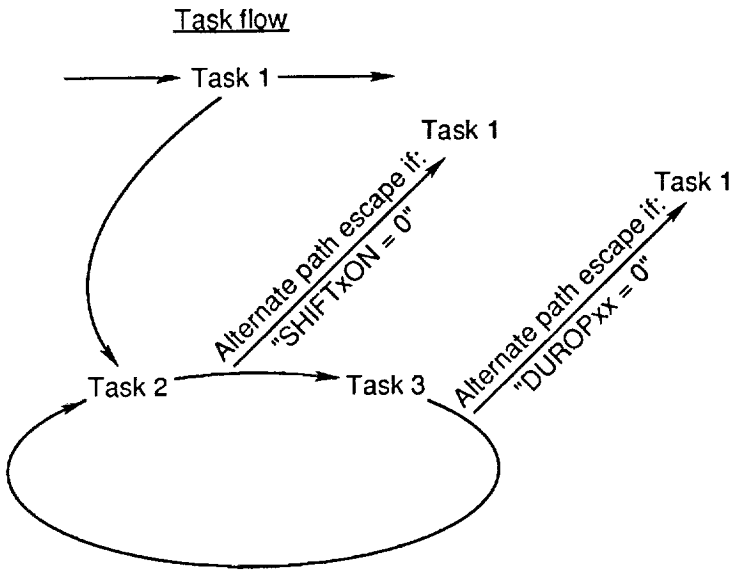


| Task | Task attributes |
|---|---|
| 1 | Duration = as required to define work shift (typically 8 hours)<br><br>At start of task:<br>set "SHIFTxON = 1"<br><br>At end of task:<br>set "SHIFTxON = 0" |

Figure 5. Shift definition operation.

**OPERATION NAME: OPxx**

Priority = As defined from priority scenario
Schedule = As required in the simulation
Location = As required in the simulation


Task flow
Task 1
Task 1
Task 1
Alternate path escape if: "SHIFTxON = 0"
Alternate path escape if: "DUROPxx = 0"
Task 2
Task 3

| Task | Task attributes |
|---|---|
| 1 | Duration = 0<br><br>Conditional logic hold if:<br>"SHIFTxON = 0"<br><br>Equipment required:<br>"TOKENxx"<br><br>At start of task:<br>set "STRTDUROPxx = 1"<br><br>At end of task:<br>set "STRTDUROPxx = 0" |
| 2 | Duration = Minimum of 1<br><br>Conditional logic escape path if:<br>"SHIFTxON = 0" |
| 3 | Duration = Minimum of 1<br><br>Conditional logic escape path if:<br>"DUROPxx = 0" |

Figure 6. OPxx description.

**OPERATION NAME: OPxy (DUROPxx)**

Priority = 0
Schedule = Same as OPxx
Location = Same as OPxx

Task flow

$$\begin{tikzcd} {} \arrow[r] & \text{Task 1} \arrow[r] & {} \end{tikzcd}$$

| Task | Task attributes |
|---|---|
| 1 | Duration = Actual required in the simulation<br><br>Conditional logic hold if: "STRTDUROPxx = 0"<br><br>Equipment required: "TIMERxx"<br><br>At start of task: set "DUROPxx = 1"<br><br>At end of task: set "DUROPxx = 0" |

Figure 7. OPxy description.

**OPERATION NAME: OPxy**

Priority = 1
Schedule = Beginning of simulation
Location = Same as OPxx

Task flow

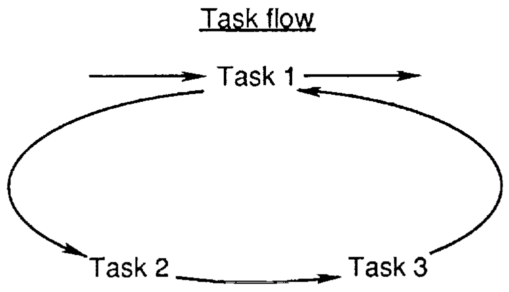
$$\begin{tikzcd} {} \arrow[r] & \text{Task 1} \arrow[r] & {} \end{tikzcd}$$

| Task | Task attributes |
|---|---|
| 1 | Duration = From the simulation start to after the final execution of OPxx/OPxy<br><br>Equipment required: "TOKENxx" and "TIMERxx" |

Figure 8. OPxz description.

**OPERATION NAME: (Counter/conditional logic hold example)**

Priority = As required
Schedule = As required

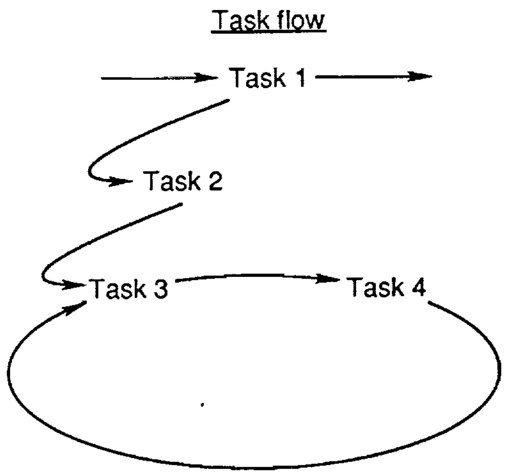


| Task | Task attributes |
|---|---|
| 1 | Duration = 0 (All tasks which have subtask have a duration = 0) |
| 2 | Duration = 0 or as required<br>Conditional logic hold until: C (counter value) = $C_n$ ($C_n$ = required counter value) |
| Other | Somewhere in the simulation, a task must increment the counter |

Figure 9. Counter/conditional logic hold example.

**OPERATION NAME: (Secondary simulation-time clock)**

Priority = As required
Schedule = As required



| Task | Task attributes |
|---|---|
| 1 | Duration = 0 |
| 2 | Duration = 0<br>Conditional logic hold when: (As required) |
| 3 | Duration = 1 (or as required)<br>Increments counted by: 1 (or as required) |
| 4 | Duration = 1 (or as required)<br>Increments counted by: 1 (or as required) |

Figure 10. Secondary simulation-time clock example.

| REPORT DOCUMENTATION PAGE | *Form Approved* *OMB No. 0704-0188* |
|---|---|

| | | |
|---|---|---|
| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE November 1991 | 3. REPORT TYPE AND DATES COVERED Technical Memorandum |

| | |
|---|---|
| 4. TITLE AND SUBTITLE User Modeling Techniques for Enhanced Usability of OPSMODEL Operations Simulation Software | 5. FUNDING NUMBERS WU 506-49-3101 |
| 6. AUTHOR(S) William T. Davis | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) NASA Langley Research Center Hampton, VA 23665-5225 | 8. PERFORMING ORGANIZATION REPORT NUMBER L-16915 |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001 | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER NASA TM-4306 |
| 11. SUPPLEMENTARY NOTES | |
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified–Unlimited Subject Category 15 | 12b. DISTRIBUTION CODE |

13. ABSTRACT *(Maximum 200 words)*

The PC-based OPSMODEL operations software for modeling and simulation of space station crew activities supports engineering and cost analyses and operations planning. Using top-down modeling, the level of detail required in the data base can be limited to being commensurate with the results required of any particular analysis. To perform a simulation, a resource environment consisting of locations, crew definition, equipment, and consumables is first defined. Activities to be simulated are then defined as "operations" and scheduled as desired. These operations are defined within a 1000-level priority structure. The simulation on OPSMODEL, then, consists of the following: user-defined, user-scheduled operations executing within an environment of user-defined resource and priority constraints. Techniques for prioritizing operations to realistically model a representative daily scenario of on-orbit space station crew activities are discussed. The large number of priority levels allows priorities to be assigned commensurate with the detail necessary for a given simulation. Several techniques for realistic modeling of day-to-day work carryover are also addressed. Recommendations for improvements for more realistic daily scenario simulation are discussed. Finally, conditional task execution based on counter status (a feature recently added to OPSMODEL) is addressed.

| | | | |
|---|---|---|---|
| 14. SUBJECT TERMS Operations, Simulation, Software, Modeling, Space Station | | | 15. NUMBER OF PAGES 11 |
| | | | 16. PRICE CODE A03 |
| 17. SECURITY CLASSIFICATION OF REPORT Unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |