Final Report
PWGSC Contract W7711-027844/A
IPME Stabilization and HAWK Development
Document Created: 9 August 2004
Last Revision: 23 February 2005

Anna Fowles-Winkler
Micro Analysis and Design, Inc.
4949 Pearl East Circle, Suite 300
Boulder, CO 80301
303-442-6947
awinkler@maad.com

Prepared For:
Defence Research and Development Canada Toronto
1133 Sheppard Avenue west
Toronto, ON

Brad Cain
416 635 2195

DRDC Document Number: CR 2004−126

Table of Contents

Abstract

This report describes the work completed by Micro Analysis and Design for DRDC-Toronto under contract W7711-027844/A, “IPME Stabilization and HAWK Development.” The contract task items were performed during the period April 2003 – April 2004. Modifications were made to the Integrated Performance Modelling Environment (IPME) and the Human Analysis Work Kit (HAWK) per customer specification.

Résumé

On décrit dans ce rapport le travail effectué par Micro Analysis and Design pour RDDC Toronto en vertu du contrat W7711-027844/A, « IPME Stabilization and HAWK Development ». Les modalités de ce contrat ont été réalisées entre avril 2003 et avril 2004. Les modifications apportées à IPME (environnement intégré de la modélisation de la performance) et à HAWK (trousse d’analyse de la performance humaine) sont celles exprimées par le client.

Executive Summary

IPME (Integrated Performance Modelling Environment) and HAWK (Human Analysis Work Kit) are software products for collecting human and system performance data in task network hierarchies such that this information can be used to predict human and system performance. These computer-based products are being developed commercially by DRDC Toronto in collaboration with the UK MOD through QinetiQ Plc., and the commercial software developer Micro Analysis and Design.

IPME uses hierarchical networks of tasks to simulate human activities. IPME has a number of component models that make it a flexible modelling tool, able to incorporate models of the environment and crew in addition to activities. IPME differs from other, general modelling approaches through both its internal human models of workload and task scheduling, as well as through its ability to easily integrate task performance moderator models. DRDC Toronto is promoting IPME for use within DND for human-systems analysis to support acquisition projects. Unfortunately, practical use of IPME has been hampered by instabilities and errors in the software. Extensive verification tests were performed on various IPME components and most of the known serious errors and bugs within IPME were corrected.

HAWK is a companion program to IPME that is being developed to support the analysis of operator activities and collection of task details, then combining this information into a model that the IPME simulation engine can execute. HAWK also provides the user with a number of human factors tools that support the modelling and assist in data analysis of the results from IPME simulations. This contract extended HAWK's capabilities by providing a new interface for creating task networks where the logical flow of tasks is presented graphically for easier interpretation, enhancing the simulation event view to provide graphical timeline-based views of simulations for model verification and debugging, and a spreadsheet data-entry tool to help populate models with data.

The contract has resulted in more capable yet stable software (IPME V3.0.11 and HAWK V3.0.8) that can be used to analyze human-systems interaction and predict human performance. Further, this work establishes a firm foundation for future development of HAWK and IPME to fulfill key roles in a proposed project to develop virtual operators that can be used for teammates and adversaries in training simulations. It is envisioned that these virtual operators will reduce the demand on CF personnel as role-players while providing desired levels of realism to meet the training objectives.

Sommaire

IPME (environnement intégré de modélisation de la performance) et HAWK (trousse d'analyse de la performance humaine) sont deux logiciels utilisés pour recueillir des données sur la performance des systèmes et des humains dans des hiérarchies de réseau de tâches et pour prédire avec ces données la performance des systèmes et des hommes. Ces produits logiciels ont été développés commercialement par RDDC Toronto en collaboration avec UK MOD via QinetiQ Plc. et avec l'éditeur de logiciels commerciaux Micro Analysis and Design.

IPME exploite les réseaux hiérarchiques de tâches pour simuler les activités humaines. IPME dispose d'un certain nombre de modèles de composant, ce qui en fait un outil de modélisation versatile en mesure d'incorporer des modèles de l'environnement et de l'équipage en plus des activités. Ce qui distingue IPME des autres approches de modélisation générale ce sont ses modèles humains internes de la planification de la charge de travail et des tâches et sa capacité à intégrer facilement les modèles modérateurs de la performance des tâches. RDDC Toronto cherche à promouvoir au sein du MDN l'utilisation de IPME pour l'analyse homme-systèmes dans le support des projets d'acquisition. Malheureusement, l'utilisation pratique de IPME a été retardée par des instabilités et des erreurs dans le logiciel. Des tests de vérifications complets ont été menés sur divers composants de IPME et la majeure partie des erreurs graves et des bogues de IPME ont été corrigés.

HAWK est un programme complémentaire de IPME qui a été développé pour aider à l'analyse des activités des opérateurs et à la collecte du détail des tâches. Ensuite, ce programme combine l'information recueillie dans un modèle que le moteur de simulation IPME peut exécuter. HAWK procure à l'utilisateur certains outils sur des facteurs humains qui assistent la modélisation et facilitent l'analyse des résultats des simulations IPME. Le présent contrat a étendu les capacités de HAWK en le dotant d'une nouvelle interface de création de réseaux de tâches dans laquelle le flux des tâches est exprimé graphiquement, ce qui en facilite l'interprétation; en rehaussant la vue des événements de simulation par une ligne de temps des vues de simulation pour faire la vérification du modèle et le débogage; et, enfin, en fournissant un outil de saisie dans un tableur qui sert à garnir les modèles avec des données.

Grâce à ce contrat, nous avons deux logiciels (IPME V3.0.11 et HAWK V3.0.8) plus puissants et plus stables pour analyser l'interaction hommes-systèmes et prédire la performance humaine. De plus, ce travail fonde solidement le développement à venir de HAWK et IPME qui serviront d'instruments clés dans un projet envisagé de mise au point d'opérateurs virtuels qui agiront comme collèges ou opposants dans des simulations de formation. On anticipe que ces opérateurs virtuels réduiront le nombre de membres des FC qui doivent participer à des jeux de rôles tout en procurant des niveaux de réalisme conformes aux objectifs de formation.

1 IPME Re-engineering

1.1 Task Network Model

1.1.1 Network Drawing

The network drawing functionality in IPME was redesigned and reimplemented, producing code that both works correctly and is maintainable. Functionality has been separated from visual attributes. Network drawing tools are implemented as separate classes (Figure 1). The Tool Palette Controller has one active tool at a time (corresponding to when the user selects a tool).

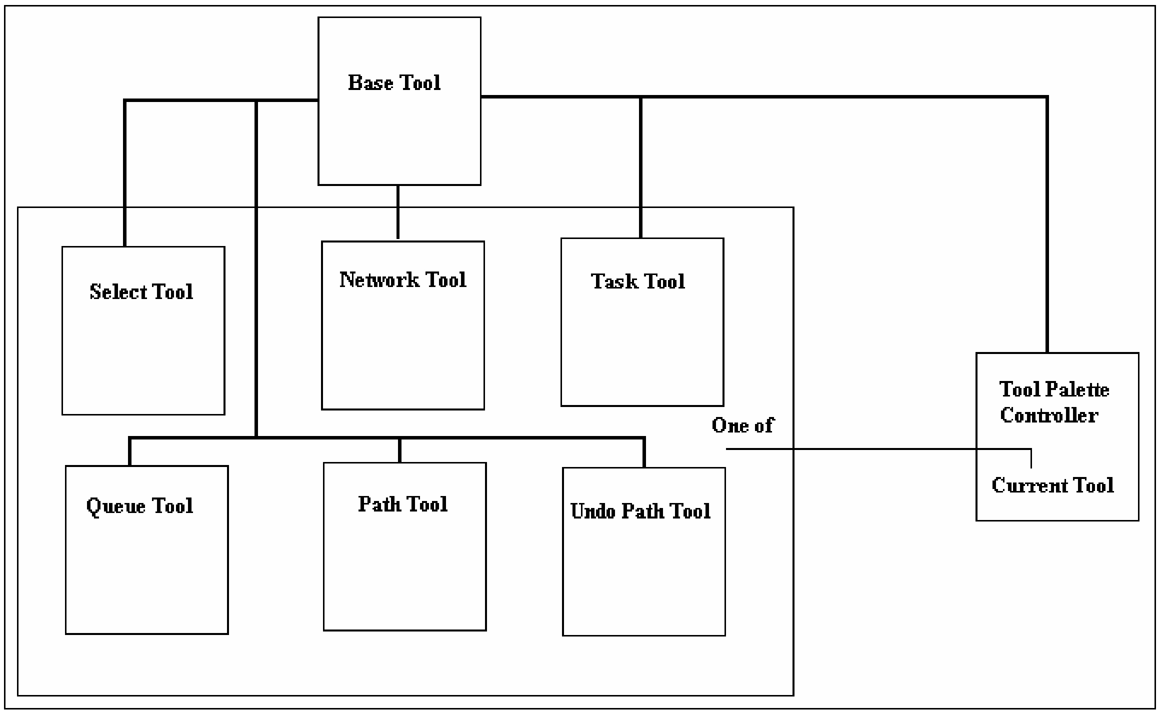


Figure 1: Network Drawing Tools

Visual Elements handle

- display information (what shape to draw, what text should be displayed)
- location information (where the element appears on the screen)
- model information (what NetObject they are associated with)
- selection state (selected or not selected)
- action information (what actions are valid for the element)

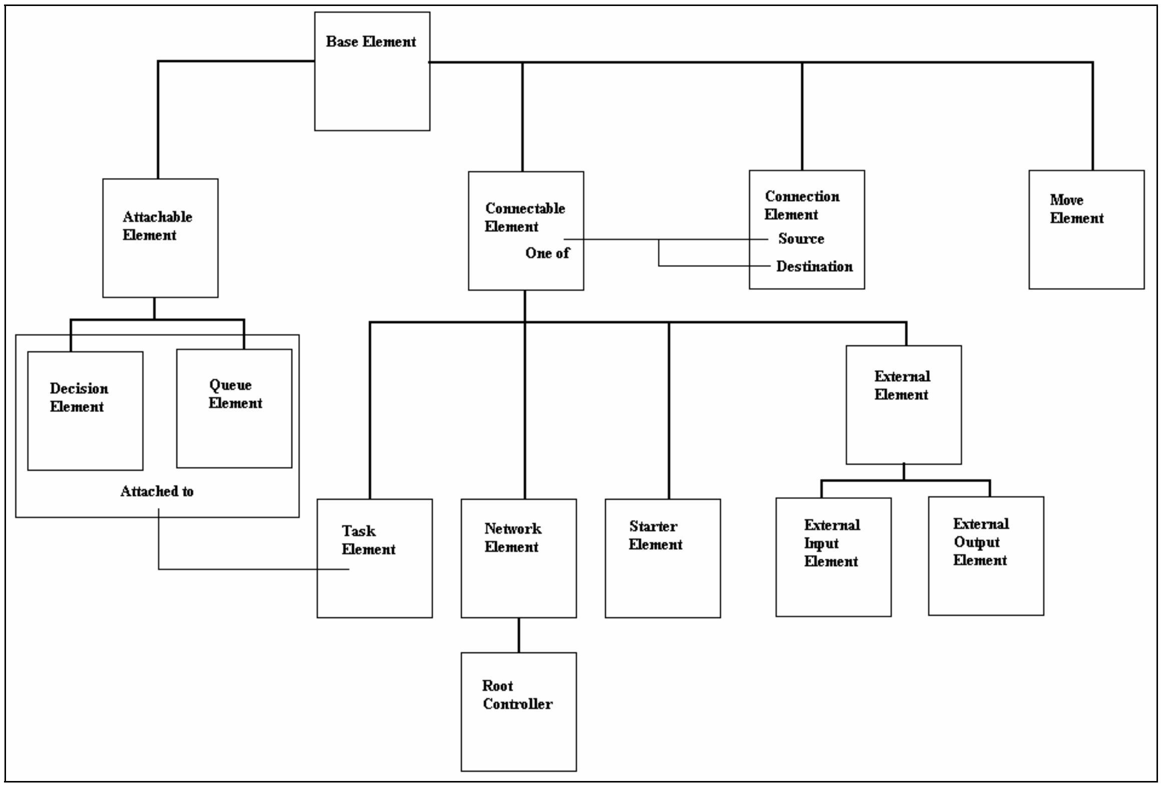


Figure 2: Visual Element Classes

Finally, actions have been broken out into separate classes. Figure 3 shows the different classes. Actions are *Commands*. Visual elements contain a list of actions they respond to. Actions are used in addition to the *Composite* structure of the visual element hierarchy because we sometimes want to limit actions taken on an element or indicate whether an action is valid for a given element type. Additionally, the action directory contains various *Mediators*. Certain actions require complicated responses. This flow is facilitated by a *Mediator*.

These significant changes to the drawing code in IPME mean that the code is more maintainable (reducing future development and maintenance costs) and less prone to errors.

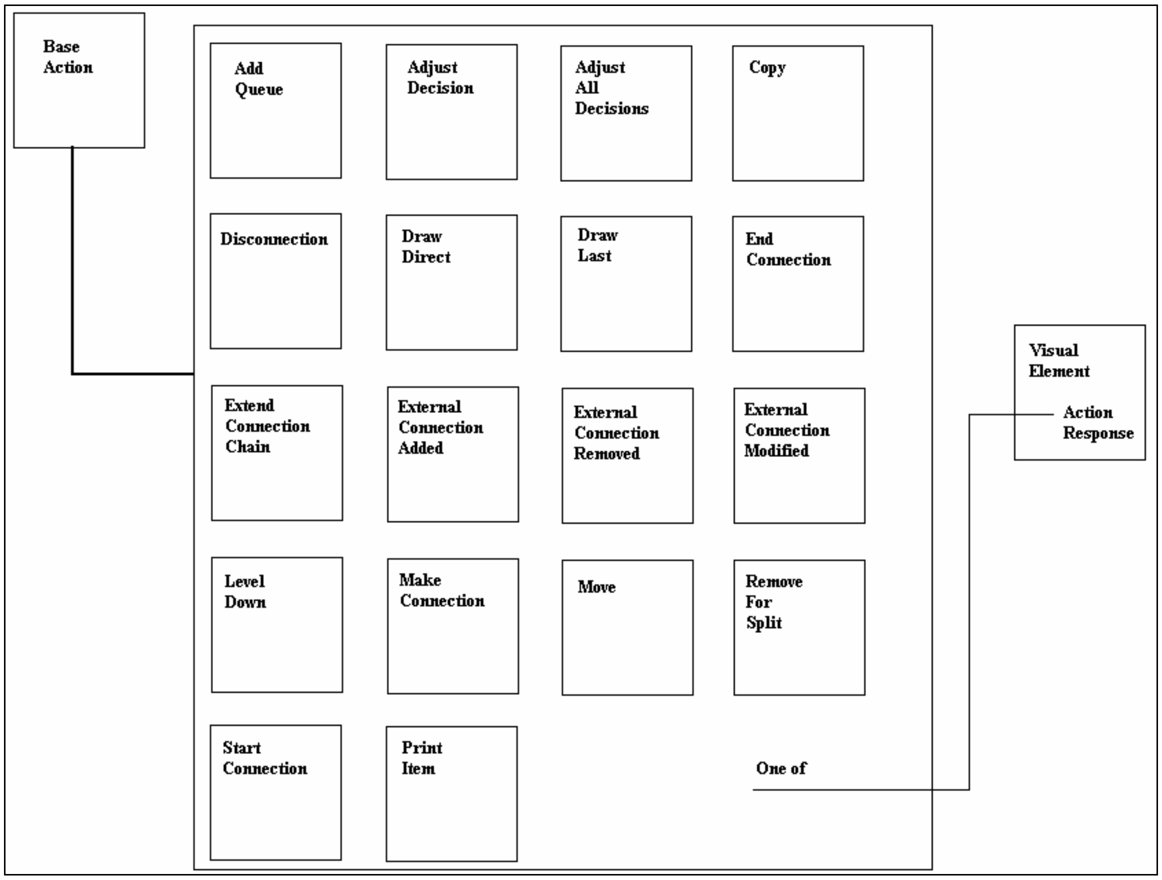


Figure 3: Action Classes

1.2.1 <u>Data integrity</u>

Data integrity in IPME was tested and reported in the Test Report for the contract. The IPME and Master databases were tested to ensure that all data would be saved to the databases.

1.2.2 <u>Internal Object Model</u>

In order to improve code maintainability, these portions of the code were reworked:

- The event queue was rewritten and improved to allow all events to be displayed.
- A class was added to support showing the correct list of operators in the task operator assignment dialog.
- A new snapshot selection dialog was added to encapsulate the dialog code.

1.2.3 <u>Task Network Model Bugs</u>

The contract listed a number of task network model problems to be fixed during this contract. All of the listed bugs were fixed and tested. Plus, additional network model problems not listed in the contract were fixed and tested. The following is the complete list of 59 network model bugs fixed during this contract (the bugs specified in the contract are shown with a yellow background):

PR	Category	State	Synopsis
2250	net_model	closed	IPME aborts with this system
2249	net_model	closed	Really long network name crashes IPME
2231	net_model	closed	File overwrite warnings are no longer displayed
2204	net_model	closed	variable changed in var cat is not updated in component models
2199	net_model	closed	difficulty default deviates from Keith's spec
2192	net_model	closed	Repeating task that starts itself crashes IPME
2189	net_model	closed	What is Data Integrity??
2184	net_model	closed	cancel on overwrite snapshot warning caused crash
2153	net_model	closed	model freezes on a queue where RC is false *this is a showstopper*
2151	net_model	closed	runtime VACP graph missing ALL labels
2110	net_model	closed	Snapshot name too short
2104	net_model	closed	Variable description is really short
2090	net_model	closed	Execution Settings Allows Runs = 0
2087	net_model	closed	IPME 3.0.4 SPR 6 - Task Network File ! Print Network Doesn't Do Anything
2085	net_model	closed	IPME 3.0.4 SPR 3: Graph snapshots and IPME hangs
2082	net_model	closed	copy network then delete causes crash
2080	net_model	closed	can't open queue stats file
2035	net_model	closed	model won't run - get divide by zero error
2017	net_model	closed	wrong operator shown in Event Queue
2016	net_model	closed	run time suspend error gives wrong task id
1948	net_model	closed	tags in queue are swapping operators
1912	net_model	closed	problems with Boolean variable type
1883	net_model	closed	change variable dimensions and select variables dialog doesn't update
1856	net_model	closed	When adding new snapshot, variable selections aren't saved.
1851	net_model	closed	More than one task dialogue for same task

1842	net_model	closed	List Variables button does not show new list selection dialog
1838	net_model	closed	VACP categories and values don't import correctly from IPME
1831	net_model	closed	Event Queue should be scrollable L to R
1807	net_model	closed	Change exit button to close on Graph Trace
1769	net_model	closed	Can't create NetModel if you have psychomotor channels selected and saved from IP defaults.
1764	net_model	closed	graph trace causes crash
1752	net_model	closed	Can't open trace file from Analyse Results
1712	net_model	closed	Set External Interface Logging Missing from Execution Settings
1710	net_model	closed	make both execute menu items behave the same
1709	net_model	closed	Enable Experiment missing from Execution Settings Dialog
1708	net_model	closed	indicate that ignore branch logic goes with other OSD option
1706	net_model	closed	Number of Crews missing on Execution Settins dialogue
1705	net_model	closed	use of output/enable on exec settings should be consistent
1679	net_model	closed	taskTimeRemaining has problems in POP Scheduler mode
1660	net_model	closed	Empty statement in release conditions, decisions
1584	net_model	closed	option to overwrite ssr files is not working
1582	net_model	closed	++group, group++ and group+=1 do not function properly in a decison
1576	net_model	closed	++tag, tag++, and tag+=1 do not work properly in decisions
1559	net_model	closed	Prob of Failure is getting executed twice
1464	net_model	closed	Aegis bug: operator variable is getting reset improperly
1455	net_model	closed	changing variable name does not update in scope { }
1443	net_model	closed	snapshots core dump IPME
1429	net_model	closed	SPR CJR26 (IPME 2.4.4): Delete snapshot, still thinks it is there when you execute.
1426	net_model	closed	SPR CJR23 (IPME2.4.4): Multiple lines on graphed snapshot don't have data points
1402	net_model	closed	Max # of objects when I don't have the max
1349	net_model	closed	change Exit button to Close on graph snapshot window
1340	net_model	closed	change "jobs" in pwr file to "Tasks"
1102	net_model	closed	IPME crashes with floating point exception.
1059	net_model	closed	cycle time distribution is hard coded and people don't know what it is
951	net_model	closed	not all tasks show up in the Event Queue
896	net_model	closed	arrow keys to scroll in snapshot vars selects vars

813	net_model	closed	Graph Trace Timeline Missing Last Run
759	net_model	closed	Can add ops to crew model when Assign To dialog is open
457	net_model	closed	consequences of failure allows "None" when prob>0 for #3

1.2.4 List Selection Dialogues

A list selection dialogues generic base class was added to the IPME code to support the many different list selection dialogues. Variable selection dialogues were updated. Scenario event selection dialogues were updated. Finally snapshot selection dialogues were updated. With the new functionality changes, it is now possible to reorder the lists and undo a mistaken “clear all”.

1.2.5 Improve Memory Handling

In the IP Scheduler, the IP queue data structure was updated from an old list structure to a C++ Standard Template Library (STL) data structure. The Global IP Parameters dialogue was updated to avoid crashes. Although not directly related to memory issues, a new execution settings dialogue was created. The POP ongoing task list was updated from an old list structure to an STL data structure.

1.2.6 Variable Catalogue

The Boolean variable type was added to both IPME and HAWK. When adding a new variable, the user may select the variable type as Boolean from the pull-down menu. The user interfaces were updated in both IPME and HAWK.

1.2.7 Access to VACP Data Values

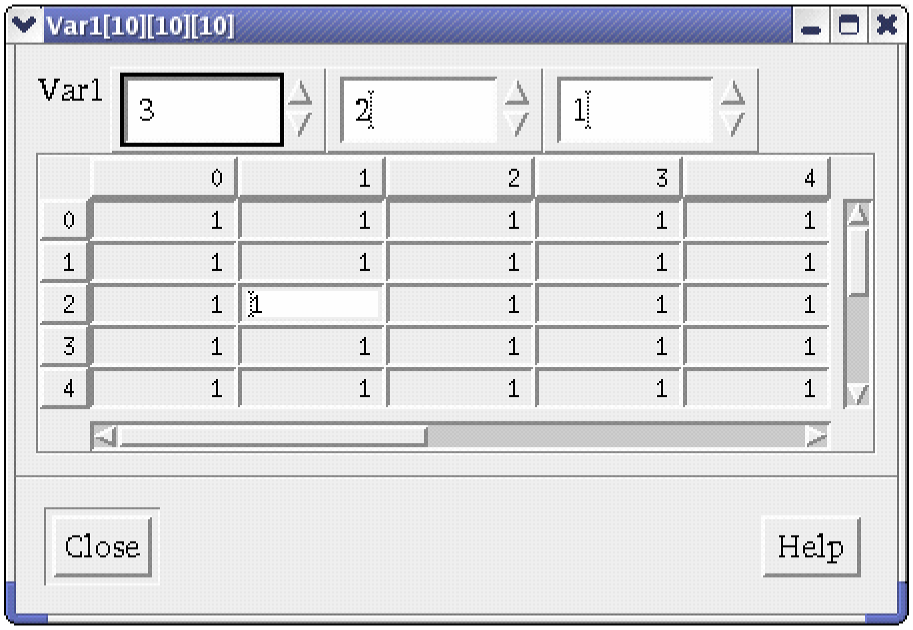
The following built-in functions were added to the IPME simulation engine parser to allow the user to collect demand ratings from these two alternative workload scales:

```
float getVACPWeight( task_id task, string vacp_category_type )
int getWIndexWeight( task_id task, string windex_category_type )
float getInstantVACPRating( string operator_name, string channel )
float getInstantWIndexRating( string operator_name )
```

The IPME Task Network User Guide describes the functions in greater detail.

1.2.8 Update Array Access and Viewing

Array Viewing was updated to allow the user to display an entire array during execution. The following picture shows a sample dialogue:



	0	1	2	3	4
0	1	1	1	1	1
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1

The user can view different dimensions for a multi-dimensional array, scroll through large arrays, and modify values in the interface.

Array selection was also updated to allow the user to choose an array variable range for snapshot variables, viewing during execution, snapshots, scenario events, and audit variables. For example, the user may specify MyVar[1-2][1-3] to choose a range of values. Additionally, an entire array may also be selected. The IPME User Guide provides more information on this new feature.

1.2.9 <u>Read/Write to File</u>

The following built-in functions were added to the IPME simulation engine parser:

    boolean openFile(string filename)
    boolean closeFile(string filename)
    string readFile(string filename)
    boolean writeFile(string filename, string output, boolean append)

These functions provide alternative means of recording data either for performance analysis or model debugging. Additionally, these functions allow the user to easily set model variables based on an external file (that might be shared by other applications)—reducing copy/paste or data entry time.

1.2.10 <u>Control Execution of Ending Effects</u>

The new Scheduling Effects tab available in the task information dialogue, in conjunction with the taskStatus() function allows the user to control task calculations and ending effects based on the scheduler and task status, including task failure.

These functions give the user a simple way to detect changes in task status, and incorporate actions based on status changes.

1.3 Improve Import/Export

XML import and export functionality was added to IPME. Two new dialogues were added: one for import, and another for export. The import and export code was added for all data in IPME. The user may now select to import or export projects, systems, or models in the XML format. This format is compatible with HAWK. Currently this feature is only available in development versions of IPME. It is anticipated that this functionality will be released as a standard feature by November 2004.

2 IP/PCT Verification

2.2 Fix identified IP/PCT Bugs

The following 114 IP/PCT bugs were fixed during this contract (bugs listed in the contract are noted in yellow).

PR	Category	State	Synopsis
2202	ip_model	closed	ip/pct ITP needs to consider tasks in the STM queue
2188	ip_model	closed	IP/PCT time pressure calculation should include STM tasks
2137	ip_model	closed	global Ip params crashes on enigma
2064	ip_model	closed	get internal error when viewing Global Ip Params
2048	ip_model	closed	ITP dpt file max value field is incorrect
2046	ip_model	closed	Task that never ends in IP/PCT mode
2025	ip_model	closed	change "vision" to "visual" in modality interference report
2022	ip_model	closed	generate IP reports screen does not resize nicely
2007	ip_model	closed	Task visual component visual area may become invalid when deleting a visual area from the global params.
1987	ip_model	closed	Scheduling and Priority info not saved to database correctly
1968	ip_model	closed	multiple cog categories - default doesn't work
1951	ip_model	closed	ip reports not using normalized check box when normalizing
1950	ip_model	closed	problem with IP reports and percentage of memory
1908	ip_model	closed	Crash when clicking Visual Area
1893	ip_model	closed	SHOWSTOPPER! Crash when editing
1888	ip_model	closed	Executing model via task network causes crash in IP/PCT mode

1876	ip_model	closed	Crash when trying to open "vision" tab
1858	ip_model	closed	Remove Generate FFD option from IPME
1843	ip_model	closed	Active tasks when task shed/del/int not shown in IP reports
1840	ip_model	closed	model is crashing in IP/PCT mode
1828	ip_model	closed	Tasks Delayed report showing values > 100%
1826	ip_model	closed	Copy from Global should enable TPMs
1801	ip_model	closed	short term to short-term
1797	ip_model	closed	TPM text boxes enabled when TPMs are not
1694	ip_model	closed	syntax error in conflict transform is not shown in check for errors
1676	ip_model	closed	resume penalty is applied to user suspend/resume in ip/pct
1666	ip_model	closed	IP/PCT scheduler uses the probability of forgetting before calculating it.
1642	ip_model	closed	clicking add area clears the workspace object column
1641	ip_model	closed	Factory Default button does not clear workspace object for vis areas
1602	ip_model	closed	Infinite loop if you have > 1 conflict transform and press Save As Default on the IP global parameters dialog.
1588	ip_model	closed	problems with the Tasks Delayed, Int, and Shed reports
1587	ip_model	closed	same tag/task pair is getting merged in STM
1586	ip_model	closed	add items to the ipr file
1567	ip_model	closed	selecting 2 tasks in compatibility tab should default to cognitive interference
1566	ip_model	closed	Reset to Defaults button does not restore some items
1565	ip_model	closed	copy from global TPM doesn't work quite right
1564	ip_model	closed	Can't save changes to a conflict transform
1561	ip_model	closed	ip_cij_visual value is always 1
1555	ip_model	closed	clciking Default button causes tabs to get wider
1554	ip_model	closed	can't delete a global TPM expression
1553	ip_model	closed	changing task category on global tab changes default settings
1552	ip_model	closed	% complete for interrupted tasks is not being logged
1551	ip_model	closed	Task interrupt in ipr does not indicate non-resumable task
1549	ip_model	closed	problems with ip_average_tp
1548	ip_model	closed	ip_critical_tp is always zero
1547	ip_model	closed	ip_task_time is always zero
1546	ip_model	closed	ip_prob_success is always zero

1544	ip_model	closed	ip_prob_detect is zero
1542	ip_model	closed	Deleting conflict transform hangs IPME
1541	ip_model	closed	Transform not saved error is weird
1539	ip_model	closed	expression in Conflict transform causes crash
1530	ip_model	closed	IPlimb gives incorrect result
1526	ip_model	closed	No default selection for alternate channel
1524	ip_model	closed	can select alternate without selecting primary
1522	ip_model	closed	tasks allowed to execute with psychomotor conflict
1521	ip_model	closed	Task Priority TPM does not get eval'd when no op assigned
1520	ip_model	closed	TPMs are getting executed too many times
1519	ip_model	closed	IPlimb function gives NULL error
1518	ip_model	closed	IPinterruptions is still not working
1514	ip_model	closed	change primary & alternate to preferred & non_preferred
1513	ip_model	closed	change ip_cij_vision to ip_cij_visual to be consistent
1504	ip_model	closed	STM size should only be editable in one place
1500	ip_model	closed	central and output give error not declared in TPMs
1499	ip_model	closed	ip_prob_detect has incorrect value
1498	ip_model	closed	vision error in crew model needs to be changed
1494	ip_model	closed	IPelapsedTime function causes error
1492	ip_model	closed	IPcategory, IPinterruptions, IPlateStart, IPlimb, IPpriorityCategory causes error
1490	ip_model	closed	error with IPattempts function
1489	ip_model	closed	error with IPpercentComplete function
1487	ip_model	closed	F2 does not work in forgetting tpm
1484	ip_model	closed	Interrupted task is being delayed
1483	ip_model	closed	tasks assigned to visual areas other than default don't go
1481	ip_model	closed	ipr file should print name of visual area, not integer id
1480	ip_model	closed	Interrupted tasks are never restarted or resumed
1478	ip_model	closed	default for Task Resumption Penalty should be 5%
1475	ip_model	closed	Task Restart Penalty should be Task Resume Penalty
1474	ip_model	closed	Task overlap time not being calculated properly
1472	ip_model	closed	Visual subtenses can be out of order
1471	ip_model	closed	Using Visual area causes crash
1470	ip_model	closed	allowed to delete Default visual area
1469	ip_model	closed	add a confirm delete message to Delete Visual Area

1468	ip_model	closed	fix error message on Misc matrix tab
1467	ip_model	closed	task components global tab has wrong default setting
1466	ip_model	closed	create and name 50 vision areas crashes ipme
1463	ip_model	closed	Allow Shed if Late for all categories of tasks
1454	ip_model	closed	error on cognitive IP defaults tab
1440	ip_model	closed	tasks delayed report is too slow
1392	ip_model	closed	Search and check conflict transforms
1247	ip_model	closed	IP/PCT execution screen should display Mode like it does in POP mode
1123	ip_model	closed	all priority 8 tasks have negative time pressure
1082	ip_model	closed	decision order impacts IP/PCT execution
1039	ip_model	closed	why is there negative time pressure in ipr file?
1035	ip_model	closed	Short Term Memory report - still has problems
952	ip_model	closed	repeating tasks not shedding correctly
897	ip_model	closed	Which ipr file are the Perl Scripts using if more than one exists
680	ip_model	closed	click Reset to defaults mult times in vision tab in crew model core dumps
678	ip_model	closed	inconsistent formatting on IP TPM tabs in task dialog
677	ip_model	closed	selecting workspace object in vis area tab corrupts work space names
675	ip_model	closed	no way to cancel on global ip param dialog
672	ip_model	closed	does not save workspace object name for visual areas when close ipme and reopen
670	ip_model	closed	deleting area from visual area tab w/object assigned object name is not cleared
666	ip_model	closed	remove the "sec" label from sliding window on global IP param
665	ip_model	closed	Prospective memory is "Short term memory" in IP report
664	ip_model	closed	reset to default creates duplicate tasks in compatiblity tab
663	ip_model	closed	reorder global task tabs to be consistent with task level
662	ip_model	closed	adding tasks changes following task default in global param
657	ip_model	closed	array var in abs/rel end field without [ ] not caught in error check
654	ip_model	closed	cont task difficulty=1 sheds itself
653	ip_model	closed	Brad Cain: Cont task d=0
638	ip_model	closed	abs/rel end field allows negative numbers
633	ip_model	closed	changing var in var catalogue does not update rel end field in IP mode

632	ip_model	closed	add stressor variable name to global IP paramater box
631	ip_model	closed	Enter expression into global TPM shows up wrong at task level
493	ip_model	closed	General/Specific identifiers on Conflict transforms does not work

Bug 1264 was deferred due to insufficient time, but will be fixed under the normal bug-fixing strategy for IPME.

The remaining IP/PCT bugs are problems with the IP functions. MA&D has discussed this issue with the Scientific Authority. MA&D needs to first confirm the desired functionality for the IP functions, and will then implement the correct functions.

The other remaining IP/PCT problems mentioned in the test report will be fixed under the normal bug-fixing strategy for IPME.

2.3 Verify IP/PCT Performance

The IP/PCT implementation was verified according to the provided specification. Test models were included with the final contract delivery. The following models were created:

- compatible_tasks_sys.dat
- IPlateStart_sys.dat
- ip_task_time_sys.dat
- IPlimb_sys2.dat
- ip_task_time_sys_modified.dat
- interrupt_logging_sys.dat
- IPlimb_sys.dat
- IPtimePressure_sys2.dat
- Int_Resume_Restart_sys.dat
- IPPCT_DelayTasks_sys.dat
- IPvisualArea_sys2.dat
- IPattempts_sys2.dat
- IPpercentcomplete_sys2.dat
- psychomotor_alternate_false_sys.dat
- ip_average_tp_sys.dat
- IPpriorityCategory_sys2.dat
- psychomotor_alternate_sys.dat
- IPcategory_crash_sys.dat
- IPpriorityCategory_sys.dat
- psychomotor_sys.dat
- ip_cij_sys.dat
- ip_prob_detect_sys.dat
- stressors_crash_sys.dat
- ip_crit_tp_sys.dat
- ip_prob_success_sys2.dat

- taskDelayed_prob_sys.dat
- IPelapsedTime_sys2.dat
- ip_prob_success_sys.dat
- taskInterrupted_prob_sys.dat
- IPinterrupt_crash_sys.dat
- IPsegmentTime_sys2.dat
- transforms_quotes_sys.dat
- IPinterruptions_sys2.dat
- IPsetCategory_nochange_sys.dat
- UserResumePenalty_sys.dat
- IPinterruptions_sys.dat
- IPsetCategory_sys2.dat
- IPlateStart_sys2.dat
- IPsetPriorityCategory_sys2.dat

The delivered test report describes the tests that were performed. As a result of the verification, a number of problems were found with the IP/PCT implementation, some of which have been fixed. The following table lists the 23 remaining known IP/PCT problems that will be fixed as part of the standard IPME bug-fixing strategy.

PR	Category	State	Severity	Synopsis
2048	ip_model	open	critical	ITP dpt file max value field is incorrect
2031	ip_model	open	serious	creating 50 visual areas causes the Global IP window to be HUGE!
2023	ip_model	open	serious	ip_task_time doesn't seem to actually affect the task time
2007	ip_model	open	serious	Task visual component visual area may become invalid when deleting a visual area from the global params.
1986	ip_model	open	serious	enabling anything on Global IP task components causes screen to resize
1972	ip_model	open	critical	IPsetCategory causes error
1971	ip_model	open	critical	get error with IP category function
1968	ip_model	open	serious	multiple cog categories - default doesn't work
1666	ip_model	open	serious	IP/PCT scheduler uses the probability of forgetting before calculating it.
1587	ip_model	open	serious	same tag/task pair is getting merged in STM
1568	ip_model	open	serious	IPsetCategory doesn't seem to actually change anything
1550	ip_model	open	serious	disabling channels globally should disable channels at task level
1540	ip_model	open	serious	either or "either" is ok with F2 check

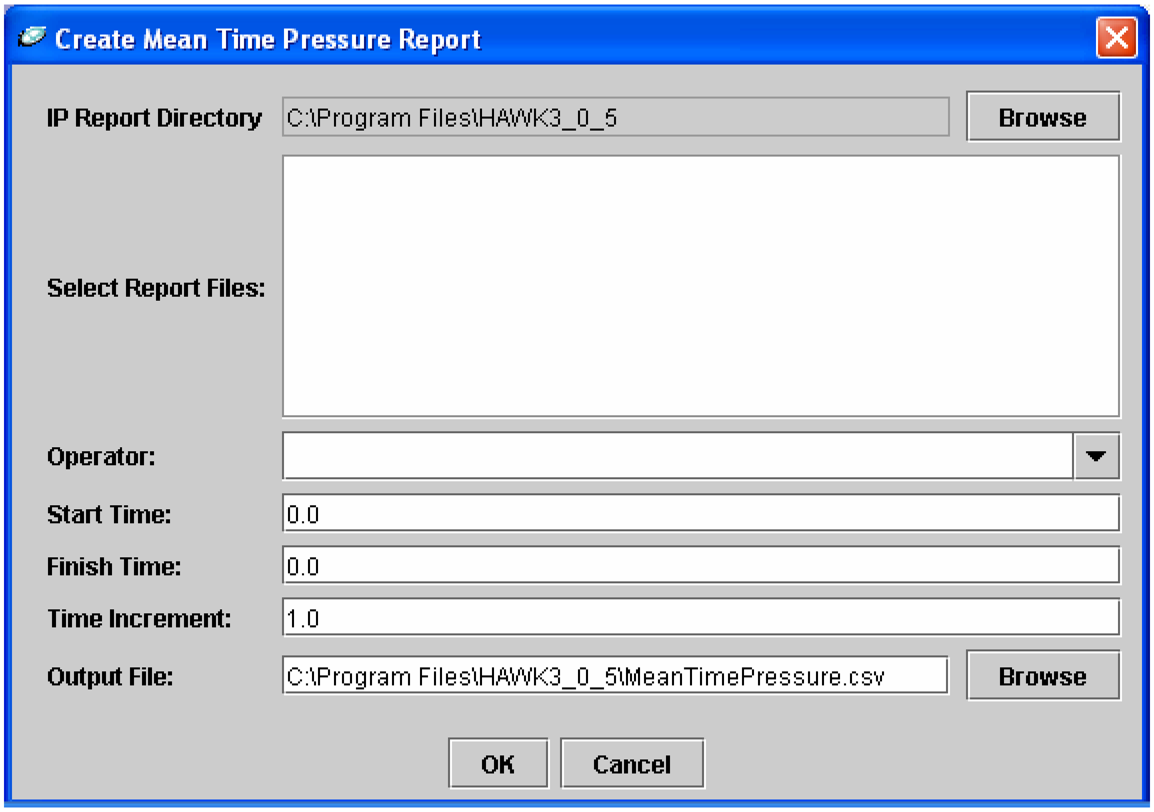
1538	ip_model	open	critical	Use of number or task_id var in IP functions causes error
1536	ip_model	open	serious	IPelapsedTime value is always zero
1535	ip_model	open	serious	IPtimePressure and IPvisualArea give error
1534	ip_model	open	serious	IPsegmentTime gives negative numbers
1533	ip_model	open	serious	IPsetPriorityCategory change is not preserved
1521	ip_model	open	serious	Task Priority TPM does not get eval'd when no op assigned
1496	ip_model	open	serious	IPsetCategory doesn't accept values
1491	ip_model	open	serious	IPattempts value does not change
1264	ip_model	open	serious	resizing Global IP Model Param windows only applies to one tab
638	ip_model	open	critical	abs/rel end field allows negative numbers

2.4 Ensure IP reports are correct

After discussion with Brad Cain, the Mean Time Pressure report has been implemented correctly. The newest HAWK release (3.0.6) includes the Mean Time Pressure report changes.

The HAWK IP report interface allows the user to select the directory where the IP report is located and the report files (allowing multiple IP reports to be selected). Based on the selected files, a list of possible Operators is provided for the user to select. The user may also specify the destination directory for the report. Because the reports were reimplemented in Java, they are significantly faster than the reports in use when the contract was written. MA&D will write a design for how to improve the current IP report implementation in HAWK. MA&D will also determine if it is possible to know how much of a file has been processed to provide a simple implementation for queuing reports and reporting on their generation status. These tasks will be completed by 30 September 2004.

The following picture shows a representative IP report dialogue (all IP report dialogues are the same):


Create Mean Time Pressure Report

IP Report Directory: C:\Program Files\HAWK3_0_5

Select Report Files:

Operator:

Start Time: 0.0

Finish Time: 0.0

Time Increment: 1.0

Output File: C:\Program Files\HAWK3_0_5\MeanTimePressure.csv

3 OSD Tool Updates and 4 Graphical Debugging Tool

Contract tasks (3) and (4) were addressed as a single task, and have resulted in the new Execution Viewer (EV) application. The EV application allows a user to see either an OSD (the traditional viewing method showing all possible paths through the network in a single view) or a graphical trace of an IPME simulation execution. A graphical trace reflects scheduler and branching effects as a single simulation run through the network, with the additional ability to show all paths through the network. The Execution Viewer was created to help the user verify task networks perform as desired. There were initially a few changes that needed to be made to the original OSD Viewer:

1. The OSD execution memory was reduced significantly to allow users to view large OSD files. This was an internal architecture change.
2. The XML file format was updated to add more information about tasks at run time. Specifically, information about how a task was started, stopped, and any other scheduling information was added. Older OSD formats are still supported.
3. The OSD Viewer was renamed to the Execution Viewer to indicate that the application displays both OSDs and graphical traces.

The EV displays tasks that have been started using start() functions and tasks that have started but were stopped via the family of stopTask functions available in IPME.

IPME was updated to allow the user to ignore branching logic. When the user chooses to ignore branching logic, IPME will treat all branches as multiples with a return condition evaluating to true. This means that every path will execute all the time.

Otherwise, actual branching logic will be applied. Additionally, IPME now generates the EV file in any scheduling mode (in the past the OSD file was only generated in IPME mode).

A graphical trace mode was added to the EV. Actual task execution times are used for this mode. Also, loopbacks are disabled in graphical trace mode—each instance of an executing task is displayed instead. The EV now saves display and directory settings.

5 Network Drawing Tool in HAWK

A network drawing tool was added to HAWK to help the analyst collect data and build task networks more conveniently. The tool uses the JGo drawing tool kit, and reproduces all of the functionality of the IPME task network drawing tool, but in a more robust and flexible design. The network drawing tool uses a palette with drawing tools. The user can drag and drop tools to the canvas. Each network is displayed in a separate drawing (canvas) window, including the top level network model. The following tasks were implemented:

- Add tasks, networks, decisions, queues, and connections between tasks and networks
- Task and network resize
- Undo, redo
- Cut, copy, paste, delete
- Show paths to and from tasks and networks
- Print network to a graphical format
- Pan over entire network
- Zoom

5.2 Report on Replacing the IPME Drawing Tool with the HAWK Drawing Tool

There are three possible techniques for replacing the IPME drawing tool. The first is to launch a Java application from IPME, the second is to make HAWK use IPME's simulation engine and the third is to replace IPME with HAWK.

5.2.1 Launching a Java Window

Of the three options, launching a Java application from IPME will require less work. However, the look-and-feel will be different from the other IPME dialogs and the transition from a C++ application to a Java application might seem slow.

There are two techniques for communicating between the Java application and IPME. One technique is to use JNI (Java Native Interfaces); the other technique is to use a client-server relationship via sockets or some other protocol.

With the JNI approach, IPME will invoke a JVM (Java Virtual Machine) that will run the Java application. Past experimentation invoking a JVM from C++ in Linux has shown that there are bugs in the JVM. Therefore, this approach is risky because it is unknown if it would actually work. Another issue with JNI is that the calls are slow, so running an animated simulation would take a performance hit.

With the client-server approach, IPME will use the operating system to launch the Java application. This way the user does not have to launch two separate applications. The socket calls will most likely be faster than JNI. The client-server approach is therefore recommended.

This approach requires that IPME supports XML import/export.

5.2.2 Connect HAWK to the IPME Simulation Engine

This approach would require IPME's simulation engine to be separated from the interface. HAWK would then connect to an IPME engine on a local or remote machine. The IPME engine would run only on Linux, so HAWK applications running on Windows would need to connect to a remote Linux computer. This approach is similar to the stand-alone IPME concept, documented in the future statement of work.

Separating the simulation engine from IPME is risky because the code is intertwined with interface calls, and potentially unknown code. This approach also requires that IPME supports XML import/export.

5.2.3 Update HAWK to Replace IPME

This option would update the HAWK application to support IPME functionality, providing a replacement for the Linux IPME. A simulation engine, including schedulers, would need to be added to HAWK. The HAWK task network drawing tool would be used. Additional user interfaces would also need to be added. While work would be required to implement this option, the end result produces a single code base for development and maintenance, versus the dual code bases we have currently (IPME and HAWK). This approach also provides multiple-platform execution: both Linux and Windows. This approach is the one that we recommend for future development, and has been specified in the recent statement of work.

6 IP/PCT Scheduler Update

The IP/PCT Scheduler implementation in IPME was modified to allow the user to select multiple cognitive categories. The user interface was updated, and the scheduler was updated. This change will be available as a standard feature in IPME by end of August 2004. HAWK will be updated to correctly handle the multiple cognitive categories by the end of September 2004. With this new functionality, if the user selects multiple cognitive categories for a task, all task interference coefficient pairs are evaluated, and the largest is selected for the interference calculation.

7 (Amendment Task) Spreadsheet Interface in HAWK

A spreadsheet view of the task network was added to HAWK. This view is called a grid editor because although it looks similar to a spreadsheet, it does not encompass all of the typical functionality that a spreadsheet would have (such as cell calculations and formulas). The grid editor displays tasks and networks in a table. Tasks are displayed in one tab, and networks are displayed in a second tab. The most common task attributes are displayed in the grid editor for editing. The user can choose to show or hide columns. Additionally, column width and row height may be modified. The font selection for the entire table can be changed. The grid editor supports cut/copy/paste and undo/redo.

As an addition to the amendment task, support for editing VACP and W/Index values in the task information operator assignment dialogue was added. This functionality change required a user interface modification, and ensuring that data was saved and loaded correctly both from the database and from XML files.

The user can now edit multiple task network models by using the spreadsheet interface. For example, if a project contains netmodel1 and netmodel2, by right-clicking on the models, and selecting ‘Grid Editor’, two separate windows display with the task network data. The user can then select tasks to copy and paste between spreadsheets.

It is also possible to have multiple HAWK applications running on a single machine. One HAWK session could be editing netmodel1, and the second session could be editing netmodel2. It is then possible to cut and paste data between the HAWK sessions.

Appendix A:
Canadian IPME Phase 6

Test Report

7021.006
POP: 1 April 2003-28 Feb 2004

Micro Analysis & Design, Inc.
4949 Pearl East Circle, Suite 300
Boulder, CO 80301
303-442-6947

2 Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.17	Anna Fowles-Winkler	Renumbered lists. Updated information for tests that had fail but now pass.	11/10/04
1.16	Shannon LaBay	Reran IP/PCT tests and updated information	10/1/04
1.15	Christy Lorenzen	Added additional information for IP/PCT Scheduler tests	03/29/04
1.14	Shannon LaBay	Added data integrity tests for IPME database	03/23/04
1.13	Christy Lorenzen	Added tests for Multiple Cognitive Categories	03/18/04
1.12	Shannon LaBay	Added tests for Read/Write functions and Boolean variable	03/16/04
1.11	Shannon LaBay	Added tests for Array Viewing	03/11/04
1.10	Unknown	Unknown	02/24/04
1.9	Christy Lorenzen	Added additional information for IP/PCT Scheduler tests	02/24/04
1.8	Shannon LaBay	Added tests for Import/Export	02/20/04
1.7	Shannon LaBay	Added tests for HAWK, OSD, and Task Network Drawing	02/19/04
1.6	Shannon LaBay	Completed Execution Viewer tests	02/11/04
1.5	Shannon LaBay	Added tests for External ins/outs and Data Integrity	02/7/04
1.4	Shannon LaBay	Added tests for Task Network Drawing and HAWK items	01/30/04
1.3	Shannon LaBay	Added tests for Task Network Drawing and listed PRs	01/16/04
1.2	Christy Lorenzen	Added test for PR 1602	10/08/03
1.1	Christy Lorenzen	Initial document	05/27/03

3 Contents

4 Introduction

Goals

The goals for the testing phase of this contract are to verify each item listed in the contract.

Assumptions

We will assume that the portions of IPME which are not affected by this contract work correctly where those portions of the software may have an impact on other sections of the software which were affected by this contract.

Risks and Assets

The pertinent risk factors involve personal obligations of the team members that may significantly limit their time available for testing and addressing those issues discovered during testing.

Terms

STM:	Short Term Memory
TPM:	Task Performance Modifier
IP/PCT:	Information Processing/Perceptual Control Theory

References

Implementing a Model of Human Information Processing in a Task Network Simulation Environment (Hendy, Keith C. and Farrell, Philip S.E.)

5 Features To Be Tested

IPME Task Network Model

Task Network Drawing

The following tests are all performed in the task network model. To create a task network:

1 Click “Open Project” on the main IPME window.
2 Click “New”.
3 Enter a name for the project and click “OK”.
4 Click “Close”.
5 Click “Select System”.
6 Click “Add”.
7 Enter a name for the system.
8 Click “OK”.
9 Click on the “Task Network” button.
10 Click “Add” on the task network selection dialogue.
11 Enter a name for the task network model.
12 Click “OK”.
13 Click “Modify” to open the task network model.

Draw Network Objects

Place each network object to verify that all drawing objects work properly. Verify that objects are drawn correctly and appear in the appropriate area.

1. Networks
2. Tasks
3. Decisions
4. Queues
5. Paths

Testing Procedure

1. Select the network object from the palette.
2. Click in the network to place the object.
3. Select the pointer object from the palette.
4. Close the task network model.
5. Reopen the task network model.
6. Verify that all objects were drawn correctly.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1747, PR1664, PR1684, PR1683, PR1682, PR1626, PR1624, PR1623, PR1622, PR1632

External Ins/External Outs

Connections to a Root Task

Test the creation of decisions from a task connected to a network. Also verify that external ins are drawn properly.

Testing Procedure

1. Add a task and a network.
2. Connect the task to the network.
3. Double click the network.
4. Verify the existence of the external in with the connected task's ID.
5. Add a task.
6. Connect the task to the external in.
7. Move back to the parent level.
8. Verify that no decision was created.
9. Double click on the network.
10. Add a task.
11. Connect the task to the external in.
12. Move back to the parent level.
13. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Connections to an Internal Task

Test the creation of decisions from a network connected to multiple tasks. Also verify that external outs are drawn properly.

Testing Procedure

1. Add a network and a task.
2. Connect the network to the task.
3. Double click the network.
4. Verify the existence of the external out with the connected task's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Add a task.
9. Connect the network to the newly created task.
10. Double click on the network.
11. Verify that a second external out was created.
12. Connect the task to the second external out.

13. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Internal to Internal Connections

Test the creation of decisions from a network connected to a network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected network's ID.
10. Add a task.
11. Connect it to the external in.
12. Move back to the parent level.
13. Double click on the first network.
14. Verify that the external out now contains the new task ID.
15. Move back to the parent level.
16. Double click the second network.
17. Add a task.
18. Connect it to the external in.
19. Move back to the parent level.
20. Double click the first network.
21. Verify that there are two external outs with the correct task IDs.
22. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Two Level Internal Connections

Variant 1

Test the creation of decisions from a network with a sub-network to another network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click on the sub-network.
8. Verify the existence of the external out with the connected network's ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task's ID.
14. Add a task.
15. Connect it to the external in.
16. Move back to the parent level.
17. Double click on the first network.
18. Verify that the external out now contains the new task ID.
19. Double click the sub-network.
20. Verify that there is no decision on the task.
21. Move up 2 levels to the parent level.
22. Double click the second network.
23. Add a task.
24. Connect it to the external in.
25. Move back to the parent level.
26. Double click the first network.
27. Verify that there are two external outs with the correct task IDs.
28. Double click the sub-network.
29. Verify that the task now has a decision.
30. Verify that there are two external outs with the correct task IDs.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 2

Test the creation of decisions from a network to a network with a sub-network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected task's ID.
10. Add a network.
11. Connect the network to the external in.
12. Double click on the sub-network.
13. Add a task.
14. Connect the task to the external in.
15. Move up 2 levels to the parent level.
16. Double click on the first network.
17. Verify that the external out now contains the new task ID.
18. Verify that there is no decision on the task.
19. Move back to the parent level.
20. Double click on the second network.
21. Double click the sub-network.
22. Add a task.
23. Connect the new task to the external in.
24. Move up 2 levels to the parent level.
25. Double click the first network.
26. Verify that there are two external outs with the correct task IDs.
27. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 3

Test the creation of decisions from a network with a subnetwork to a network with a sub-network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network’s ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click on the sub-network.
8. Verify the existence of the external out with the connected network’s ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task’s ID.
14. Add a network.
15. Connect the network to the external in.
16. Double click on the sub-network.
17. Add a task.
18. Connect the task to the external in.
19. Move up 2 levels to the parent level.
20. Double click on the first network.
21. Verify that the external out now contains the new task ID.
22. Double click the sub-network.
23. Verify that there is no decision on the task.
24. Move up 2 levels to the parent level.
25. Double click on the second network.
26. Double click the sub-network.
27. Add a task.
28. Connect the new task to the external in.
29. Move up 2 levels to the parent level.
30. Double click the first network.
31. Verify that there are two external outs with the correct task IDs.
32. Double click the sub-network.
33. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 4

Test the creation of decisions from a network to a network with a sub-network and a task. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move up to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected task's ID.
10. Add a network.
11. Connect the network to the external in.
12. Double click on the sub-network.
13. Add a task.
14. Connect the task to the external in.
15. Move up 2 levels to the parent level.
16. Double click on the first network.
17. Verify that the external out now contains the new task ID.
18. Verify that there is no decision on the task.
19. Move up to the parent level.
20. Double click on the second network.
21. Add a task.
22. Connect the new task to the external in.
23. Move up to the parent level.
24. Double click the first network.
25. Verify that there are two external outs with the correct ID numbers.
26. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 5

Test the creation of decisions from a network with a sub-network to a network with a sub-network and a task. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.

3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click the sub-network.
8. Verify the existence of the external out with the connected network's ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task's ID.
14. Add a network.
15. Connect the network to the external in.
16. Double click on the sub-network.
17. Add a task.
18. Connect the task to the external in.
19. Move up 2 levels to the parent level.
20. Double click on the first network.
21. Verify that the external out now contains the new task ID.
22. Double click on the sub-network.
23. Verify that there is no decision on the task.
24. Move up to the parent level.
25. Double click on the second network.
26. Add a task.
27. Connect the new task to the external in.
28. Move up to the parent level.
29. Double click the first network.
30. Verify that there are two external outs with the correct ID numbers.
31. Double click on the sub-network.
32. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Basic Destination Split and Merge

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.

2. Connect the networks.
3. Double click the second network.
4. Add two tasks.
5. Connect both tasks to the external in.
6. Move up to the parent level.
7. Double click the first task.
8. Add a task.
9. Connect the new task to one external out.
10. Move up to the parent level.
11. Double click the second task.
12. Verify that there are now two external ins.
13. Verify that each task is connected to an external in.
14. Move up to the parent level.
15. Double click on the first task.
16. Connect the task to the other external out.
17. Move up to the parent level.
18. Double click on the second network.
19. Verify that there is only one external in.
20. Verify that both tasks are connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Multi-level Split and Merge

Variant 1

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the second network.
4. Add a network.
5. Connect the sub-network to the external in.
6. Double click the sub-network.
7. Add two tasks.
8. Connect both tasks to the external in.
9. Move up to the parent level.
10. Double click the first task.

11. Add a task.
12. Connect the new task to one external out.
13. Move up to the parent level.
14. Double click the second task.
15. Verify that there is one external in.
16. Double click the sub-network.
17. Verify that each task is connected to an external in.
18. Move up to the parent level.
19. Double click on the first task.
20. Connect the task to the other external out.
21. Move up to the parent level.
22. Double click on the second network.
23. Verify that there is only one external in.
24. Double click the sub-network.
25. Verify that there is only one external in.
26. Verify that both tasks are connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 2

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the second network.
4. Add a network.
5. Connect the sub-network to the external in.
6. Add a task.
7. Connect the task to the external in.
8. Double click the sub-network
9. Add a task.
10. Connect the task to the external in.
11. Move up to the parent level.
12. Double click the first task.
13. Add a task.
14. Connect the new task to one external out.
15. Move up to the parent level.

16. Double click the second task.
17. Verify that there are two external ins.
18. Verify that the task is connected to one external in and the network is connected to the other.
19. Double click the sub-network.
20. Verify that there is one external in.
21. Move up to the parent level.
22. Double click on the first task.
23. Connect the task to the other external out.
24. Move up to the parent level.
25. Double click on the second network.
26. Verify that there is only one external in.
27. Verify that the network and the task are both connected to it.
28. Double click the sub-network.
29. Verify that there is only one external in.
30. Verify that the task is connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Basic Source Split and Merge

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add two tasks.
5. Connect both tasks to the external out.
6. Move up to the parent level.
7. Double click on the second network.
8. Add a task.
9. Connect the task to one of the external ins.
10. Move up to the parent level.
11. Double click on the first network.
12. Verify that there are two external outs.
13. Verify that each task is connected to one of the external outs.
14. Move up to the parent level.
15. Double click on the second network.
16. Connect the task to the other external in.

17. Move up to the parent level.
18. Double click on the first network.
19. Verify that only one external out is present and that both tasks are connected to it.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1633

Multi-level Source Split and Merge

Variant 1

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add a network.
5. Connect the network to the external out.
6. Double click on the sub-network.
7. Add two tasks.
8. Connect both tasks to the external out.
9. Move up to the parent level.
10. Double click on the second network.
11. Verify that there are two external ins.
12. Add a task.
13. Connect the task to one of the external ins.
14. Move up to the parent level.
15. Double click on the first network.
16. Verify that only one external out is present.
17. Double click on the sub-network
18. Verify that there are two external outs.
19. Verify that each task is connected to one external out.
20. Move up to the parent level.
21. Double click on the second network.
22. Connect the task to the other external in.
23. Move up to the parent level.
24. Double click on the first network.
25. Verify that there is only one external out.
26. Double click on the sub-network.

27. Verify that there is only one external out.
28. Verify that both tasks are connected to the external out.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Variant 2

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add a network.
5. Connect the network to the external out.
6. Add a task.
7. Connect the task to the external out.
8. Double click on the sub-network.
9. Add a task.
10. Connect the task to the external out.
11. Move up to the parent level.
12. Double click on the second network.
13. Verify that there are two external ins.
14. Add a task.
15. Connect the task to one of the external ins.
16. Move up to the parent level.
17. Double click on the first network.
18. Verify that two external outs are present.
19. Verify that the task is connected to one external out and the network is connected to the other.
20. Double click on the sub-network
21. Verify that there is one external out.
22. Move up to the parent level.
23. Double click on the second network.
24. Connect the task to the other external in.
25. Move up to the parent level.
26. Double click on the first network.
27. Verify that there is only one external out.
28. Verify that both the network and the task are connected to the external out.
29. Double click on the sub-network.

30. Verify that there is only one external out.
31. Verify that the task is connected to it.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Double-Click on Network Objects.

Test double-click on all network objects. Verify that the resulting action is correct.

1. Networks
2. Tasks
3. External Ins/Outs
4. Decisions
5. Queues

Testing Procedure

1. Select the network object from the palette.
2. Click in the network to place the object.
3. Select the pointer object from the palette.
4. Double click on the object.
5. Verify that the appropriate action occurs. For networks, verify that the sub-network is displayed. For externals, verify that the parent network is displayed. For all other objects, verify that the proper dialogue is displayed.
6. Close the dialogue.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Right Click on Network Objects.

Test right click on all network objects. Verify that the resulting action is correct.

6. Networks
7. Tasks
8. External Ins/Outs
9. Decisions
10. Queues

Testing Procedure

1. Select the network object from the palette.
2. Click in the network to place the object.
3. Select the pointer object from the palette.

4. Right click on the object.
5. Verify that the appropriate action occurs. For externals, verify that the data in the information dialogue is correct.
6. Close the dialogue.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1663

Cut/Paste Network Objects

Perform Cut/Paste procedures on all network objects. Cut/Paste will be performed on each network object, then groups of 3 or more like objects, and then on mixed groups made up of combinations of these objects. Paste will be performed at the same level, and at different levels of the task network model.

11. Networks
12. Tasks
13. External Ins/Outs
14. Decisions
15. Queues
16. Paths

Testing Procedure

6. Select the network object from the palette.
7. Click in the network to place the object.
8. Select the pointer object from the palette.
9. Click on the object to select it.
10. Choose Network Diagram | Cut Object.
11. Choose Paste Object.
12. Verify that all network objects have been pasted appropriately, deleted from their original position, all content has been retained, and all network objects have been renumbered appropriately.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1816, PR1702, PR1698, PR1695, PR1689, PR1665

Copy/Paste Network Objects

Perform Copy/Paste procedures on all network objects. Copy/Paste will be performed on each network object, then groups of 3 or more like objects, and then on mixed groups made up of combinations of these objects. Paste will be performed at the same level, and at different levels of the task network model.

17. Networks
18. Tasks

19. External Ins/Outs
20. Decisions
21. Queues
22. Paths

Testing Procedure

1. Select the network object from the palette.
2. Click in the network to place the object.
3. Select the pointer object from the palette.
4. Click on the object to select it.
5. Choose Network Diagram | Copy Object.
6. Choose Paste Object.
7. Verify that all network objects have been pasted appropriately, deleted from their original position, all content has been retained, and all network objects have been renumbered appropriately.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1816

Delete Network Objects

Perform Delete procedures on all network objects. Delete will be performed on each network object, then groups of 3 or more like objects, and then on mixed groups made up of combinations of these objects.

23. Networks
24. Tasks
25. External Ins/Outs
26. Decisions
27. Queues

Testing Procedure

1. Select the network object from the palette.
2. Click in the network to place the object.
3. Select the pointer object from the palette.
4. Click on the object to select it.
5. Choose Network Diagram | Delete Object.
6. Verify that the object was removed.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Zoom In/Zoom Out

Perform zoom in and zoom out on the network diagram.

Testing Procedure

1. Select “Zoom in” from the palette menu.
2. Verify that network objects are larger.
3. Select “Zoom out” from the palette menu.
4. Verify that network objects return to the default size.
5. Select “Zoom out” from the palette menu.
6. Verify that network objects are smaller.

Model Used

DRDC_Test/Palette_Func.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Show Paths To/From

Perform show paths to and show paths from on networks and tasks.

Testing Procedure

1. Click on a task or network.
2. Select Network Diagram | Show Paths To.
3. Verify that the paths going to the object are highlighted.
4. Select Network Diagram | Show Paths From.
5. Verify that the paths leading out from the object are highlighted.

Model Used

DRDC_Test/Palette_Func.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Map

Test map feature to ensure that it is functioning properly.

Testing Procedure

1. Double-click into network.
2. Click the “Map” button on the tool palette.
3. Verify that the map is correct.
4. Click on the “GoTo” button.

5. Verify that the parent level of the network is shown.

Model Used

DRDC_Test/Palette_Func.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

GoTo

Test "Go To" feature to ensure that it is functioning properly.

Testing Procedure

1. Click the "GoTo" button on the tool palette.
2. From the list of tasks, enter 4.1 next to "Go to ID:".
3. Click OK.
4. Verify that the subnetwork is displayed with task 4.1 selected.
5. Click the "GoTo" button on the tool palette.
6. From the list of networks, enter 4 next to "Go to ID:".
7. Verify that the parent level is displayed and network 4 is selected.

Model Used

DRDC_Test/Palette_Func.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

HFTD Data

HFTD Data should be tested to make sure that tasks are placed correctly in the task network model.

Testing Procedure

1. On the main IPME window, select File | Import → Import HFTD.
2. Import existing HFTD data.
3. In the task network, select File | HFTD Data from the file menu.
4. Select a task from the list.
5. Click "Add".
6. Verify that the task was added to the task network.
7. Double click on the task.
8. Verify that data was loaded properly.

Data Used

testmodels2/development/hftd/NEWTEST2.txt

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1810

Read and Write to File Functions

boolean openFile(string filename)

Testing Procedure

1. Create a system.
2. Open the task network model.
3. Add a task.
4. Enter 5; for the Mean Time.
5. In the Beginning Effects of the task enter:
```
openFile(“Test_IO.txt”);
```
6. In the Ending Effects of the task enter:
```
{
Boolean isOpen;
isOpen = false;
isOpen = openFile(“Test_IO.txt”);
debug(1,isOpen,1,1);
}
```
7. Execute the model.
8. Verify that the value of isOpen is true in the debug statement.
9. After model execution, verify the existence of “Test_IO.txt” in the output directory.

Model Used

testmodesl2/development/DRDC_Test/Test_IOFunc_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1928

boolean closeFile(string filename)

Testing Procedure

1. Create a system.
2. Open the task network model.
3. Add 2 tasks.
4. Enter 5; for the Mean Time of both tasks.

5. In the Beginning Effects of the first task enter:
   openFile(“Test_IO.txt”);
6. In the Ending Effects of the task enter:
   closeFile(“Test_IO.txt”);
7. In the Beginning Effects of the second task, enter:
   closeFile(“Test_IO.txt”);
13. Execute the model.
14. Verify that an error occurs when the second closeFile function is executed.
15. After model execution, verify the existence of “Test_IO.txt” in the output directory.

Model Used

testmodesl2/development/DRDC_Test/Test_IOFunc_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1929

boolean writeFile(string filename, string output, boolean append)

Testing Procedure

1. Create a system.
2. Open the task network model.
3. Add a task.
4. Enter 5; for the Mean Time.
5. In the Beginning Effects of the first task enter:
   writeFile(“Test_IO.txt”, “Testing1”, true);
6. In the Ending Effects of the task enter:
   writeFile(“Test_IO.txt”,”Testing2”,true);
7. Execute the model.
8. After model execution, verify the existence of “Test_IO.txt” in the output directory.
9. View the file.
10. Verify that the file contains Testing1 and Testing2 on two different lines.
11. Repeat the test changing the Ending Effects of the task to:
    writeFile(“Test_IO.txt”,”Testing2”,false);
12. View the file.
13. Verify that the file contains only Testing2.

Model Used

testmodesl2/development/DRDC_Test/Test_IOFunc_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

string readFile(string filename)

After performing the previous test:

Testing Procedure

1. In the Beginning Effects of the first task enter:

```
{
string ReadVar;
ReadVar = readFile("Test_IO.txt");
debug(1,ReadVar,1,1);
}
```

2. Execute the model.
3. Verify that ReadVar equals “Testing2”.

Model Used

testmodesl2/development/DRDC_Test/Test_IOFunc_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Boolean Variable Type

The Boolean variable type should be selectable in the Variable Catalogue.

Set Initial Value

Testing Procedure

1. Add a variable name Var1 to the Variable Catalogue.
2. Select Boolean as the variable type.
3. Set the initial value to “true.”
4. Run the model, displaying Var1 during execution.
5. Verify that the value for the Boolean variable is “true.”
6. Repeat test using initial value of “false.”

Model Used

testmodesl2/development/DRDC_Test/Test_Bool.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Translate Integer Values

Testing Procedure

1. Add a variable name Var1 to the Variable Catalogue.
2. Select Boolean as the variable type.
3. Set the initial value to 1.
4. Run the model, displaying Var1 during execution.
5. Verify that the value for the Boolean variable is "true."
6. Repeat test using initial value of zero.
7. Verify that the value for the Boolean variable is "false."

Model Used

testmodesl2/development/DRDC_Test/Test_Bool.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1910

List Selection Dialogues

Verify selections, saving, and importing/exporting

Verify that variables and events can be selected, Saved to local IPME database, and exported/imported using the XML files

The Client Exchange Variables, Snapshot Variables, Display Variables, Take Snapshots, Active Scenario Events, and Audit Variables windows have been restructured for better usability. Verify that all previous functionality still exists.

Dialog Opened From:	*Select Variables*	*Select Array Var Ranges*	*Save/Load DB*	*Export/Import XML*
Client Exchange Variables	Pass	Pass	Pass	Pass: PR 1879
Snapshot Variables	Pass	Pass	Pass	Pass: PR 1880
Execution Settings:				
Display Variables	Pass	Pass	Pass	Pass: PR 1880
Take Snapshots	Pass	N/A	Pass	Pass: PR 1881
Execute Scenario Events	Pass	N/A	Pass	Pass: PR 1882
Audit Variables	Pass	Pass	Pass	Pass: PR 1880

Model Used

testmodesl2/test/xml/testArrayRange_Selexport.syx

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR 1879, 1880, 1881, 1882, 1856, 1866, 1877

Array Range Selection

Verify that the array ranges selected for the snapshot and audit variables are decomposed into their elemental references for file output (.ssr and .audit). For example, the range MyVar[1-2][1-3] ought to yield the following single elements/columns in the output files: MyVar[1][1], MyVar[1][2], MyVar[1][3], MyVar[2][1], MyVar[2][2], and MyVar[2][3].

Testing Procedure

1. Create a new system.
2. Open the task network model.
3. Add one task.
4. Add Var1 of type Array of Integers.
5. Make it two dimensions.
6. Enter 10 and 20 for the highest indeces.
7. Add a snapshot that collects the value of Var1[1-2][1-3] at the end of run.
8. Execute the model collecting the snapshot.
9. View the snapshot results.

Model Used

Default

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Array Dimension changes

Verify that array dimension changes are handled correctly. The changes should be reflected in the array range selection, add/edit array range selection, and the variable selection dialogs.

Testing Procedure

1. Create new system.
2. Open task net.
3. Add Var1, Array of Integers, 3-dimension array with 10, 20, 30 as the highest indeces.
4. Add Snap1 which collects the default range of Var1.
5. Change Var1 to a 2-dimension array.
6. Notice the message that says it has been removed from the snapshot.
7. Open the snapshot again and click List Variables.
8. Add Var1 to the selected list and double click.

Model Used

Default

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR 1883

Add/Edit Array Range only accepts valid values

The Add/Edit Array Range dialog should only accept valid values. Negative numbers, values outside the bounds, and non-numeric (text) entries should all be detected, communicated to the user, and prevented.

Testing Procedure

1. Create new system.
2. Open task net.
3. Add Var1, Array of Integers, 3-dimension array with 10, 20, 30 as the highest indeces.
4. Add Snap1.
5. Select Var1 for collection.
6. Double click Var1.
7. For the first index, attempt to enter a range of -10- -5.
8. For the first index, attempt to enter a range of zero – five.
9. For the first index, attempt to enter a range of 0-15.
10. For the first index, attempt to enter a range of 15-0.
11. Click OK.
12. Verify that those ranges are not allowed.

Model Used

Default

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Test new dialog buttons

The OK buttons should save changes. The Cancel button should disregard changes. The Add New Range button should launch the Add/Edit Array Range dialog. The Change Selected Range button should launch the Add/Edit Array Range dialog populated with the existing range selection. The Delete Selected Range button should prompt the user to confirm the delete action. Confirmation should delete the selected range. Canceling the confirmation should preserve the range. The spinner up/down arrows should increment/decrement by one until the minimum/maximum values are reached. Also check that deleting all existing ranges is treated as the default range.

Testing Procedure

1. Create new system.
2. Open task net.
3. Add Var1, Array of Integers, 3-dimension array with 10, 20, 30 as the highest indeces.
4. Add Snap1.

5. Select Var1 for collection.
6. Double click Var1.
7. Click the Add New Range button.
8. Enter values [0-5][5-10][10-15].
9. Click OK.
10. Verify new range saved.
11. Click Add New Range button.
12. Enter values.
13. Click Cancel.
14. Verify new range not saved.
15. Select Range [0-5][5-10][10-15].
16. Click Change Selected Range.
17. Verify that the values are populated with [0-5][-5-10][10-15].
18. Change to [0-10][5-10][10-15].
19. Click OK.
20. Verify change was saved.
21. Click Delete Selected Range.
22. Click Cancel.
23. Click Delete Selected Range.
24. Click OK.
25. Verify range was deleted.
26. Click Cancel.
27. Double click Var1.
28. Verify that only the default range exists.
29. Delete the default range.
30. Click OK.
31. Double click Var1.
32. Verify that no ranges are listed.
33. Click OK.
34. Click OK.
35. Reopen the Snapshot variables list.
36. Double click Var1.
37. Verify that the default range has been placed there since all ranges were deleted.

Model Used

Default

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Viewing Arrays While Executing a Simulation

Array values should display correctly as the model is running. Controls to scroll through the array should be working properly.

Testing Procedure

1. Create new system.
2. Create a task network model.
3. Modify the task network model.
4. Select Variable Catalogue from the Define menu.
5. Add Var1, Array of Integers, 3-dimension array with 10, 20, 30 as the highest indeces.
6. Close the dialogue.
7. Add a task to the task network.
8. Enter a mean time of 10;
9. In the Beginning Effects enter:

```
{
int z;
for (z = 0; z < 30; z += 1)
        Var1[10][20][z] = 5;
}
```

10. Close the task dialogue.
11. Add a second task with a mean time of 10;.
12. Select Execute | Execution Settings.
13. Enable Display Variables.
14. Click Choose Variables.
15. Select Var1[10][20][30].
16. Close the dialogues.
17. Select Execute | Execute Simulation.
18. Click Show Array.
19. In the Array Viewing Dialogue, inside the first box enter 10. Inside the second box enter 20.
20. Step through the model.
21. Verify that after the first task executes, all values in the 20th row are 5.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Updating Array Values While Executing a Simulation

Array values should be editable as the model is running. Controls to scroll through the array should be working properly.

Testing Procedure

1. Create new system.

2. Create a task network model.
3. Modify the task network model.
4. Select Variable Catalogue from the Define menu.
5. Add Var1, Array of Integers, 3-dimension array with 10, 20, 30 as the highest indeces.
6. Close the dialogue.
7. Add a task to the task network.
8. Enter a mean time of 10;
9. Close the task dialogue.
10. Add a second task with a mean time of 10;.
11. In the beginning effects of the second task enter:
10. debug(1,Var1[5][5][5], Var1[5][5][6],1);
12. Select Execute | Execution Settings.
13. Enable Display Variables.
14. Click Choose Variables.
15. Select Var1[10][20][30].
16. Close the dialogues.
17. Select Execute | Execute Simulation.
18. Click Show Array.
19. At the top of the Array Viewing Dialogue, inside the first box enter 5. Inside the second box enter 5. Inside the third box enter 5.
20. In row 5, column 5, enter the number 4.
21. In row 5, column 6, enter the number 8.
22. Click on another cell to update the values.
23. Step through the model.
24. Verify that the debug statement shows the values of 4 and 8.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduling Effects tab

Mode Independent

OK Button

This button closes the form and saves any changes you have made to the form.

Testing Procedure

1. Open the scheduler effects tab.
2. Enter “5;” in the field.
3. Click OK.

4. Reopen the scheduler effects tab.
5. Verify that the information has been saved.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Cancel Button

This button closes the form without saving any changes you have made to the form.

Testing Procedure

1. Open the scheduler effects tab.
2. Enter “5;” in the field.
3. Click OK.
4. Reopen the scheduler effects tab.
5. Verify that the information has not been saved.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Help Button

This button launches the related help topic.

Testing Procedure

1. Open the scheduler effects tab
2. Click on the Help button.
3. Verify that the correct help topic is displayed.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduler Effects Field – F1

F1 launches the related help topic.

Testing Procedure

1. Open the scheduler effects tab.
2. Place the cursor in the field.
3. Press F1.
4. Verify that the correct help topic is displayed.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduler Effects Field – F2

F2 performs a syntax check.

Testing Procedure

1. Open the scheduler effects tab.
2. Place the cursor in the field.
3. Enter “undefined_variable;”
4. Press F2.
5. Verify that a syntax check is performed and that it alerts you to an undefined variable.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduler Effects Field – F3

F3 displays a list of available operators, variables, and functions.

Testing Procedure

1. Open the scheduler effects tab.
2. Place the cursor in the field.
3. Press F3.
4. Verify that the variable list, operator list, and function list window is displayed.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduler Effects Field – F4

F4 launches the text import browser.

Testing Procedure

1. Open the scheduler effects tab.
2. Place the cursor in the field.
3. Press F4
4. Verify that the text import browser is displayed.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Import/Export

The scheduler effects should be exported and imported. Test this for the system and task network.

Testing Procedure

1. Create System1.
2. Add a task.
3. Open that task and view the scheduler effects tab.
4. Enter “testvalue;” in the field.
5. Click OK to close the task.
6. Choose File | Export | System.
7. Select System1.
8. Name the export file System1.dat.
9. Choose File | Export | Task Network Model.
10. Choose System1_Net1.
11. Name the export file System1_Net1.dat
12. Choose File | Import | System.
13. Browse to and select System1.dat.
14. Rename the system on import to System1_imported.
15. Rename the models on import to System1_<modelname>_imported
16. Open the task network model.

17. View the task contents on the scheduler effects tab.
18. Verify that the info was preserved.
19. Choose File | Import and Assign | Task Network model.
20. Browse to and select System1_Net1.dat
21. Rename the model on import to System1_Net1_imported2.
22. Open the task network model.
23. View the task contents on the scheduler effects tab.
24. Verify that the info was preserved.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Save to/Load from Master Database

The scheduler effects should be saved to the master database with a network fragment. The scheduler effects should also be loaded with the network fragment from the master database.

Testing Procedure

1. Create a new system.
2. Open the task network model.
3. Add a network.
4. Go down one level into that network.
5. Add a task.
6. Open the task and view the scheduler effects tab.
7. Enter “test;” in the field.
8. Click OK to close the task.
9. Go up one level.
10. Select the network.
11. Choose Network Diagram | Save Network to Master DB.
12. Enter “Network1” as the name.
13. Click OK.
14. Enter your username and password.
15. Click OK.
16. Delete the network.
17. Click OK on the warning message.
18. Choose Network Diagram | Load from Master DB.
19. Select Net1 and click OK.
20. Go down one level into the network.
21. Open the task.

22. Verify that the information on the scheduler tab was preserved.

Model Used

Default

Test Platform

Sirius – RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduling Effects and taskTimeRemaining Test Grid

	IPME		POP		IP/PCT		POPIP	
	field	task Time Remaining	field	task Time Remaining	field	task Time Remaining	field	task Time Remaining
Active	Beginning Effects	N/A	Beginning Effects	N/A	Beginning Effects	N/A	Beginning Effects	Correct
Started	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct
Completed	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct
Failed	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct
User Stopped	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct	Ending Effects	Correct
User Suspended	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct
User Resumed	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct
Interrupted	N/A	N/A	Scheduling Effects	PR 1679	Scheduling Effects	Correct	Scheduling Effects	Correct
Resumed	N/A	N/A	Scheduling Effects Ending Effects	Correct	Scheduling Effects	Correct	Scheduling Effects	Correct
Delayed	N/A	N/A	Ending Effects	PR 1679	Scheduling Effects	Correct	Scheduling Effects	Correct
Shed	N/A	N/A	N/A	N/A	Scheduling Effects	Correct	Scheduling Effects Ending Effects	Correct Correct
Restarted	N/A	N/A	N/A	N/A	Scheduling Effects	Correct	Scheduling Effects	Correct

This test grid was filled with the field where each taskStatus was seen. For example, the only field which will ever display a taskStatus of Active is the Beginning Effects field. The models were constructed so that in each mode, every possible task status was exhibited. This required changing operator assignments, IP components parameters, IP scheduling parameters, Probability of Failure, POP Workload Channel demands, and other task parameters.

The taskTimeRemaining function was also verified for each taskStatus in each simulation mode. The taskTimeRemaining function was verified for every possible taskStatus in the fields where it is valid (Ending Effects and Scheduling Effects only, it is not valid in the Beginning Effects). The taskTimeRemaining function was found to properly take into account the effects of TDMs (with the exception or PR 1679), IP/PCT task overlap penalties, IP/PCT Task Resume Penalty, and POPIP Task Resume Penalty.

Models Used

testmodels2/test/Safework/IPME_mode_taskStat_sys.dat
testmodels2/test/Safework/POP_mode_taskStat_sys.dat
testmodels2/test/Safework/IPPCT_mode_taskStat_sys.dat
testmodels2/test/Safework/POPIP_mode_taskStat_sys.dat

XML Import/Export

XML Import/Export in IPME

XML Import/Export should be tested for each type of file: project, system, environment, crew, task network, and PSF.

Testing Procedure

1. Select File | Export → Export XML from the Main IPME window.
2. In the dialogue, select the project name.
3. Enter a name for the file at the prompt.
4. Click OK.
5. Delete IPME database.
6. Select File | Import → Import XML from the Main IPME window.
7. Select "Project" from the list of options.
8. Enter the name of the file (or use the Browse button to find the file).
9. Click OK.
10. Verify that all data was imported correctly.
11. Verify that there is no data loss.
12. Repeat test at the system, environment, crew, task network, and PSF levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1874

XML Export from IPME Import into HAWK

The XML Import file format should be tested for each type of import: project, system, environment, crew, task network, PSF.

Basic Export/Import

Testing Procedure

1. Select File | Export → Export XML from the Main IPME window.
2. In the dialogue, select the project name.
3. Enter a name for the file at the prompt.
4. Click OK.
5. Open HAWK.
6. Select File | Import → Project.
7. Select the name of the file from the file browser.
8. Click Open.
9. Verify that all data was imported correctly.
10. Verify that there is no data loss.
11. Repeat test at the system, environment, crew, task network, and PSF levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

Re-Export from HAWK into IPME

Testing Procedure

Continue from the previous test:

1. In HAWK, select File | Export → Project.
2. Enter a name for the file at the prompt.
3. Click Save.
4. Open IPME.

7. Select File | Import → Import XML from the Main IPME window.
8. Select "Project" from the list of options.
9. Enter the name of the file (or use the Browse button to find the file).
10. Click OK.
11. Verify that all data was imported correctly.
12. Verify that there is no data loss.
13. Repeat test at the system, environment, crew, task network, and PSF levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

Save to Remote Database

Testing Procedure

1. Select File | Export → Export XML from the Main IPME window.
2. In the dialogue, select the project name.
3. Enter a name for the file at the prompt.
4. Click OK.
5. Open HAWK.
6. Select File | Import → Project.
7. Select the name of the file from the file browser.
8. Click Open.
9. Select Database | Remote → Connect to Database.
10. Enter username and password to connect to the IPME database.
11. Select Database | Remote → Save Project to Remote Database.
12. Open IPME.
13. Click "Open Project".
14. Select the project name from the list of options.
15. Click "Close".
16. Verify that there is no data loss in the project.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

Data Integrity

IPME Database

Projects, Systems, and Component Models

Using the MySQL Control Center, verify that the correct operations are performed in the database for projects, systems, and component models.

Testing Procedure

1. Perform save, load, cancel, rename, copy and delete procedures.

2. Verify that the correct operations are performed.
    28. Save – data should be added to the database
    29. Load – data should be displayed from the database
    30. Cancel – data should not be added to the database
    31. Rename - database name should be updated
    32. Copy – a copy of the data should be added to the database
    33. Delete – data should be removed from the database

	Save	Cancel	Load	Copy	Delete	Rename
Project	Pass	Pass	Pass	Pass	Pass	Pass
System	Pass	Pass	Pass	Pass	Pass	Pass
Task Network Model	Pass	Pass	Pass	Pass	Pass	N/A
Environment Model	Pass	Pass	Pass	Pass	Pass	N/A
Crew Model	Pass	Pass	Pass	Pass	Pass	N/A
PSF Model	Pass	Pass	Pass	Pass	Pass	N/A

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

Environment Model

Using the MySQL Control Center, verify that the correct operations are performed in the database for the environment variables of the environment model.

Testing Procedure

1. Perform save, load, cancel, copy and delete procedures.
2. Verify that the correct operations are performed.
3. Save – data should be added to the database
4. Load – data should be displayed from the database
5. Cancel – data should not be added to the database
6. Copy – a copy of the data should be added to the database
7. Delete – data should be removed from the database

	Save	Cancel	Load	Copy	Delete
Threat					
Target_Bearing	Pass	Pass	Pass	Pass	N/A
Target_Elevation	Pass	Pass	Pass	Pass	N/A
Target_Location	Pass	Pass	Pass	Pass	N/A
Target_Obscuration	Pass	Pass	Pass	Pass	N/A
Target_Range	Pass	Pass	Pass	Pass	N/A
Target_Signature	Pass	Pass	Pass	Pass	N/A
Target_Speed	Pass	Pass	Pass	Pass	N/A

Target_Value	Pass	Pass	Pass	Pass	N/A
Threat_Severity	Pass	Pass	Pass	Pass	N/A
User-Defined	Pass	Pass	Pass	Pass	Pass
Physical					
Ambient_Noise	Pass	Pass	Pass	Pass	N/A
Contamination_Level	Pass	Pass	Pass	Pass	N/A
Contamination_Type	Pass	Pass	Pass	Pass	N/A
Digability	Pass	Pass	Pass	Pass	N/A
Dry_Bulb_Temperature	Pass	Pass	Pass	Pass	N/A
Env_Type	Pass	Pass	Pass	Pass	N/A
Footing	Pass	Pass	Pass	Pass	N/A
Humidity	Pass	Pass	Pass	Pass	N/A
Pressure	Pass	Pass	Pass	Pass	N/A
Illumination	Pass	Pass	Pass	Pass	N/A
Sea_State	Pass	Pass	Pass	Pass	N/A
Temperature	Pass	Pass	Pass	Pass	N/A
Terrain	Pass	Pass	Pass	Pass	N/A
Terrain_Direction	Pass	Pass	Pass	Pass	N/A
Terrain_Slope	Pass	Pass	Pass	Pass	N/A
Thermal_Radiation	Pass	Pass	Pass	Pass	N/A
Time_Of_Day	Pass	Pass	Pass	Pass	N/A
Turbulence	Pass	Pass	Pass	Pass	N/A
Weather	Pass	Pass	Pass	Pass	N/A
Wind_Direction	Pass	Pass	Pass	Pass	N/A
Wind_Strength	Pass	Pass	Pass	Pass	N/A
Wind_Speed	Pass	Pass	Pass	Pass	N/A
User-Defined	Pass	Pass	Pass	Pass	Pass
Mission					
Adequacy_of_Procedures	Pass	Pass	Pass	Pass	N/A
Communications_Density	Pass	Pass	Pass	Pass	N/A
Intelligence	Pass	Pass	Pass	Pass	N/A
Platform_Reliability	Pass	Pass	Pass	Pass	N/A
Surveillance_Reliability	Pass	Pass	Pass	Pass	N/A
Time_Stress	Pass	Pass	Pass	Pass	N/A
Weapons_Reliability	Pass	Pass	Pass	Pass	N/A
User-Defined	Pass	Pass	Pass	Pass	Pass
Crew					
Clarity_of_Role	Pass	Pass	Pass	Pass	N/A
Cooperation	Pass	Pass	Pass	Pass	N/A
Leadership_Style	Pass	Pass	Pass	Pass	N/A
Supervision	Pass	Pass	Pass	Pass	N/A

Team_Experience	Pass	Pass	Pass	Pass	N/A
Team_Morale	Pass	Pass	Pass	Pass	N/A
Team_Training	Pass	Pass	Pass	Pass	N/A
User-Defined	Pass	Pass	Pass	Pass	Pass

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

Crew Model

Using the MySQL Control Center, verify that the correct operations are performed in the database for the operator component of the crew model.

Testing Procedure

1. Perform save, load, cancel, copy and delete procedures.
2. Verify that the correct operations are performed.
3. Save – data should be added to the database
4. Load – data should be displayed from the database
5. Cancel – data should not be added to the database
6. Copy – a copy of the data should be added to the database
7. Delete – data should be removed from the database

	Save	Cancel	Load	Copy	Delete
Operator	Pass	Pass	Pass	Pass	Pass
Operator Name	Pass	Pass	Pass	N/A	N/A
Location --> x	Pass	Pass	Pass	N/A	N/A
Location --> y	Pass	Pass	Pass	N/A	N/A
Properties --> Eyes	Pass	Pass	Pass	Pass	N/A
Properties --> Eyes --> Attribute	Pass	Pass	Pass	Pass	Pass
Properties --> Left Foot	Pass	Pass	Pass	Pass	N/A
Properties --> Left Foot --> Attribute	Pass	Pass	Pass	Pass	Pass
Properties --> Left Hand	Pass	Pass	Pass	Pass	N/A
Properties --> Left Hand --> Attribute	Pass	Pass	Pass	Pass	Pass
Properties --> Right Foot	Pass	Pass	Pass	Pass	N/A
Properties --> Right Foot --> Attribute	Pass	Pass	Pass	Pass	Pass
Properties --> Right Hand	Pass	Pass	Pass	Pass	N/A
Properties --> Right Hand --> Attribute	Pass	Pass	Pass	Pass	Pass
Properties --> User-defined	Pass	Pass	Pass	Pass	Pass
Properties --> User-defined --> Attribute	Pass	Pass	Pass	Pass	Pass
Traits --> User-defined	Pass	Pass	Pass	Pass	Pass

Traits --> User-defined --> Attribute	Pass	Pass	Pass	Pass	Pass
States --> User-defined	Pass	Pass	Pass	Pass	Pass
States --> User-defined --> Attribute	Pass	Pass	Pass	Pass	Pass
Anthropometry --> Sex	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> Anthropography	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> BB	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> HW	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> HT	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> SIH	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> BKL	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> BW	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A
Anthropometry --> PBF	Pass/PR1989	Pass	Pass/PR1989	N/A	N/A

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

Performance Shaping Functions

Using the MySQL Control Center, verify that the correct operations are performed in the database for PSFs.

Testing Procedure

1. Perform save, load, cancel, copy and delete procedures.
2. Verify that the correct operations are performed.
3. Save – data should be added to the database
4. Load – data should be displayed from the database
5. Cancel – data should not be added to the database
6. Copy – a copy of the data should be added to the database
7. Delete – data should be removed from the database

	Save	Cancel	Load	Copy	Delete
PSF	Pass	Pass	Pass	Pass	Pass
PSF Name	Pass	Fail/PR1988	Pass	Pass	Pass
PSF Type	Pass	Fail/PR1988	Pass	Pass	Pass
Taxon Assignments	Pass	Pass	Pass	Pass	Pass
Expression	Pass	Pass	Pass	Pass	Pass

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

Task Network Model

Network Objects

Using the MySQL Control Center, verify that the correct operations are performed in the database for each of the network objects.

Testing Procedure

1. Perform draw, copy, cut, paste, and delete on all network objects.
2. Verify that the correct operations are performed.
3. Draw – object should be added to the database
4. Copy – object should remain in the database
5. Cut – object should be removed from the database
6. Paste – object should be added to the database
7. Delete – object should be removed from the database

34.

	Draw	Cut	Copy	Paste	Delete
Task	Pass	Pass	Pass	Pass	Pass
Queue	Pass	Pass/PR1557	Pass	Pass	Pass
Network	Pass	Pass	Pass	Pass	Pass
Decision	Pass	Pass	Pass	Pass	Pass
External Ins	Pass	Pass	Pass	Pass	Pass
External Outs	Pass	Pass	Pass	Pass	Pass

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

Network Object Data

Using the MySQL Control Center, verify that the correct operations are performed in the database for all of the data contained in the network objects.

Testing Procedure

1. Perform draw, copy, cut, paste, and delete on all network objects.
2. Verify that the correct operations are performed.
3. Save – data should be added to the database
4. Load – data should be displayed from the database
5. Cancel – data should not be added to the database

35.

	Save	Cancel	Load

Networks			
Network Name	Pass	Pass	Pass
Notes	Pass	Pass	Pass
Queues			
Queue Name	Pass	Pass	Pass
Queue Order	Pass	Pass	Pass
Entering Effect	Pass	Pass	Pass
Departing Effect	Pass	Pass	Pass
Release Condition	Pass	Pass	Pass
Sort Order	Pass	Pass	Pass
Decisions			
Decision Type	Pass	Pass	Pass
Expression1	Pass	Pass	Pass
Expression2	Pass	Pass	Pass
Expression3	Pass	Pass	Pass
Tasks - IPME			
Task Name	Pass	Pass	Pass
Task Notes	Pass	Pass	Pass
OSD Task Type	Pass	Fail/PR1990	Pass
Critical/Machine/Normalized	Pass	Fail/PR1990	Pass
Time Distribution	Pass	Pass	Pass
Mean Time	Pass	Pass	Pass
Standard Deviation	Pass	Pass	Pass
Definition of Failure	Pass	Pass	Pass
Release Condition	Pass	Pass	Pass
Beginning Effects	Pass	Pass	Pass
Ending Effects	Pass	Pass	Pass
Probability of Failure	Pass	Pass	Pass
Assign To --> Operator Assignment	Pass	Pass	Pass
Assign To --> Taxon Percentages	Pass	Pass	Pass
Assign To --> Visual	Pass	Pass	Pass
Assign To --> Auditory	Pass	Pass	Pass
Assign To --> Cognitive	Pass	Pass	Pass
Assign To --> Psychomotor	Pass	Pass	Pass
Assign To --> W/Index Channels	Pass	Pass	Pass
Consequences of Failure -- > No Effect	Pass	Pass	Pass
Consequences of Failure -- > Time Degraded	Pass	Pass	Pass
Consequences of Failure -- > Task Degraded	Pass	Pass	Pass
Consequences of Failure -- > Percentage Time Degradation	Pass	Pass	Pass
Consequences of Failure -- > Percentage Failure Degradation	Pass	Pass	Pass
Consequences of Failure -- > Different Task Follows	Pass	Pass	Pass
Consequences of Failure -- > Task to Follow	Pass	Pass	Pass
Consequences of Failure -- > Repeat Unsuccessful Task	Pass	Pass	Pass

Consequences of Failure -- > Terminate Model	Pass	Pass	Pass
Repeating --> Task Type	Pass	Pass	Pass
Repeating --> Cycle Time	Pass	Pass	Pass
Repeating --> Cycle Standard Deviation	Pass	Pass	Pass
Repeating --> Active Time	Pass	Pass	Pass
Repeating --> Active Standard Deviation	Pass	Pass	Pass
Repeating --> Difficulty	Pass	Pass	Pass
Repeating --> Task Start	Pass	Pass	Pass
Repeating --> Expression	Pass	Pass	Pass
Repeating --> Task End	Pass	Pass	Pass
Repeating --> Expression	Pass	Pass	Pass
Scheduling Effects	Pass	Pass	Pass
Tasks - POP			
Workload Percentages --> Input Demand	Pass	Pass	Pass
Workload Percentages --> Input Demand --> Visual	Pass	Pass	Pass
Workload Percentages --> Input Demand --> Auditory	Pass	Pass	Pass
Workload Percentages --> Central Demand	Pass	Pass	Pass
Workload Percentages --> Central Demand --> Spatial	Pass	Pass	Pass
Workload Percentages --> Central Demand --> Verbal/Numeric	Pass	Pass	Pass
Workload Percentages --> Central Demand --> Other	Pass	Pass	Pass
Workload Percentages --> Output Demand --> Manual	Pass	Pass	Pass
Workload Percentages --> Output Demand --> Interference Channels	Pass	Pass	Pass
Workload Percentages --> Output Demand --> Vocal	Pass	Pass	Pass
Time Pressure --> Internally Paced	Pass	Pass	Pass
Time Pressure --> Externally Paced	Pass	Pass	Pass
Priority	Pass	Pass	Pass
Tasks - IP/PCT			
Visual --> Enabled	Pass	Pass	Pass
Visual --> Externally Cued	Pass	Pass	Pass
Visual --> Category	Pass	Pass	Pass
Visual --> Visual Area	Pass	Pass	Pass
Auditory --> Enabled	Pass	Pass	Pass
Auditory --> Externally Cued	Pass	Pass	Pass
Auditory --> Category	Pass	Pass	Pass
Cognitive --> Enabled	Pass	Pass	Pass
Cognitive --> Category	Pass	Pass	Pass
Psychomotor --> Enable Preferred	Pass	Pass	Pass
Psychomotor --> Enable Non-Preferred	Pass	Pass	Pass
Psychomotor --> Preferred Channels	Pass	Pass	Pass
Psychomotor --> Non-Preferred Channels	Pass	Pass	Pass
Miscellaneous --> Enabled	Pass	Pass	Pass
Miscellaneous --> Category	Pass	Pass	Pass
Task Data Confirmed	Pass	Pass	Pass

Scheduling and Priority --> Category	Pass	Pass	Pass
Scheduling and Priority --> Multiplier	Pass	Pass	Pass
Scheduling and Priority --> Action	Pass	Pass	Pass
Crtical To --> Mission Performance	Pass	Pass	Pass
Crtical To --> Mission Awareness	Pass	Pass	Pass
Crtical To --> Situation Awareness	Pass	Pass	Pass
Consequences of Shedding--> Continue Branch	Pass	Pass	Pass
Consequences of Shedding--> Terminate Branch	Pass	Pass	Pass
Consequences of Shedding--> Start Other Task	Pass	Pass	Pass
Consequences of Shedding--> Task	Pass	Pass	Pass
TPMs --> Adaptation to Time Stress (Speed/Accuracy)	Pass	Pass	Pass
TPMs --> Adaptation to Time Stress (Visual Detection)	Pass	Pass	Pass
TPMs --> Forgetting	Pass	Pass	Pass
TPMs --> Task History	Pass	Pass	Pass
TPMs --> Task Priority	Pass	Pass	Pass
TPMs --> Aptitude	Pass	Pass	Pass
TPMs --> Experience	Pass	Pass	Pass
TPMs --> Non-preferred Limb	Pass	Pass	Pass
TPMs --> Stressor 1	Pass	Pass	Pass
TPMs --> Stressor 2	Pass	Pass	Pass
TPMs --> Stressor 3	Pass	Pass	Pass
TPMs --> Stressor 4	Pass	Pass	Pass
TPMs --> Stressor 5	Pass	Pass	Pass

36.

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

User-Defined General Information

Using the MySQL Control Center, verify that the correct operations are performed in the database for user-defined information.

Testing Procedure

1. Perform save, load, and cancel procedures and copy and delete procedures where applicable.
2. Verify that the correct operations are performed.
3. Save – data should be added to the database
4. Load – data should be displayed from the database
5. Cancel – data should not be added to the database
6. Copy – a copy of the data should be added to the database
7. Delete – data should be removed from the database

	Save	Cancel	Load	Copy	Delete
Variables	Pass	Pass	Pass	Pass	Pass
Functions	Pass	Pass	Pass	Pass	Pass
Scenario Events --> Event	Pass	Pass	Pass	Pass	Pass
Scenario Events --> Time --> Discrete	Pass	Pass	Pass	Pass	Pass
Scenario Events --> Time --> Repeating	Pass	Pass	Pass	Pass	Pass
Snapshots --> End of Run	Pass	Fail/PR1985	Pass	Pass	Pass
Snapshots --> Discrete	Pass	Fail/PR1985	Pass	Pass	Pass
Snapshots --> Repeating	Pass	Fail/PR1985	Pass	Pass	Pass
Snapshots --> Task	Pass	Fail/PR1985	Pass	Pass	Pass
Snapshots --> Queue	Pass	Fail/PR1985	Pass	Pass	Pass
IP Parameters --> General	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Visual	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Auditory	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Cognitive	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Miscellaneous	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Task Components	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Task Scheduling	Pass/PR1987	Pass/PR675	Pass/PR1987	N/A	N/A
IP Parameters --> TPMs	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Visual Areas	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Conflict Transforms	Pass	Pass/PR675	Pass	N/A	N/A
IP Parameters --> Compatibility	Pass	Pass/PR675	Pass	N/A	N/A

Test Platform

RedHat 7.2

Pass/Fail

See above table.

Related PR #s

See above table.

IP/PCT Scheduler

Task restart penalty

- Should have the possibility of modification at the task level
- default = 1.05
- Actually applies to resuming not restarting

Pass/Fail

Pass

Related PR #s

PR 1475, 1478, 1676

Concurrent processing penalty

When two tasks execute simultaneously, are assigned to the same operator, and have some level of interference greater than zero.

Pass/Fail

Pass

Related PR #s

PR 1474, 1476

Interference

VISION

The following elements must be present in the vision channel interference implementation.

- Placing visual tasks into correct subtense categories (Category 1-4)
- Probability of detection matrix is used before visual interference matrix
- Externally Cued Detection
- Visual tasks that make it to the temp queue will be added to active queue

The above items could not be verified due to PRs 1561.

Pass/Fail

Pass

Related PR #s

PR 1483, 1498, 1499, 1513, 1544, 1561, 1642, 1876, 1908, 1472, 1471, 1470, 1469, 1466

Home area

- Are operator's eyes returning properly when a
    - Task completes
    - Task interrupted
    - Task shed
- Is it selecting the highest priority visual task for the new eye position?
- Central tasks take precedence over peripheral

The above items could not be verified due to PRs 1561.

- Are at least 50 visual areas allowed? Yes, maximum is 50.
- Can the user change the home area via syntax during model execution? No
- What is the x,y,z unit for declaring visual areas? Unknown.

Pass/Fail

Pass

Related PR #s

PR 1466, 1469, 1470, 1471, 1641, 1561, 1471, 1544

COGNITIVE

- All operator tasks must be assigned a cognitive category.

This is not required, but the Factory Default settings include enabling the Cognitive Channel for each task.

Pass/Fail

Pass

Related PR #s

PR 1454

PSYCHOMOTOR

- Tasks involving different digits of the same hand will interfere.
    - Selection of digit will also select hand (and vice versa).
    - Selection of foot will also select leg (and vice versa).
    - Multiple selection allowed
        - Check this: Task 1- Left leg, right leg
        - Task 2- left foot
        - Do they interfere? Yes. Should they? Yes.
- If the preferred channel is in use, IPME will attempt to perform the task with the non-preferred channel. Yes.

Pass/Fail

Fail: PR 2229 (problem with non-preferred channels)

Related PR #s

PR 1514, 1522, 1523, 1524, 1526, 2229

MISC (pg. 38)

- Up to 15 category fields should be editable: Yes.
- Names of category fields should be editable: Yes
- Name of the table for a specific application should be editable: No.

Pass/Fail

Pass

Related PR #s

PR 1468

(pg 38) Conflict transforms

- Applied after initial cij calculation

  Verified using /development/IPmodels/ConflictTrans_ipcijcognitive_sys.dat
- Information which should be available
    - Task id's of all tasks in temp queue

      This can be accomplished by using the case parameter for IP functions.

      This could not be tested due to PR 1538.
    - Task identifiers for pairwise consideration

      This can be accomplished by using the case parameter for IP functions.

      This could not be tested due to PR 1538.
    - Domain categories of all tasks in temp queue

      This can be accomplished by using the IPcategory function.

      This could not be tested due to PR 1538.
    - Current visual area

      This can be accomplished by using the IPvisualArea function

      This could not be tested due to PRs1538 and1535.

  - Tasks status
    This can be accomplished by using the IPstatus function.
    This could not be tested due to PR 1538
    .
- Specific transformations take precedence over general ones. If an applicable conflict transform exists, then general ones will not be applied (pg 41).
- Conflict transforms can be saved as defaults.
  PR 1602
- Conflict transforms are imported/exported with a system.
  PR 1602

Testing Procedure

1. Create a new system.
2. Select IP/PCT Simulation Mode.
3. Open the task network model.
4. Select Define | IP Model Parameters.
5. Click on the Conflict Transform tab.
6. Click New Transform.
7. Click Save to List.
8. Repeat steps 6-7 to add a second conflict transform.
9. Click Save As Default.
10. Click Factory Defaults.
11. Click Reset to Defaults. The two conflict transforms should be restored.
12. Export the system.
13. Import the system.
14. Rename the system and the component models.
15. Verify that only the two default conflict transforms are present.
16. Click Reset to Defaults.
17. Verify that only two default conflict transforms are present.
18. Click New Transform.
19. Enter “2;” in the Effects field.
20. Verify the Save To List button is enabled.
21. Click Reset to Defaults.
22. Verify that unsaved changes were removed.
23. Verify that the Save To List button is disabled.

Pass/Fail

Fail: PR 1535 (IP functions not working correctly), 1538 (use of task ID in IP functions)

Related PR #s

PR 1535, 1538, 1602, 1694, 1566, 1564, 1542, 1541, 1539, 1503

COMBINING INTERFERENCE (pg 42)

- Max value of all active 5 domains is the cij value used to describe interference

- Check what happens when domain with max interference is disabled at global level? At task level? Passes.

Pass/Fail

Pass

Related PR #s

N/A

ENABLING/DISABLING INTERFERENCE (pg 43)

- User should be able to enable/disable all interference effects in each domain independently globally
- When a domain's interference is disabled globally, the associated setting and dialogs at the task level should be grayed out.

Pass/Fail

Fail: PR 1550

Related PR #s

PR 1550

SCHEDULING (pg 45)

- Predecessor shedding of cont/repeating tasks
    - If a predecessor exists, successive cycles will be shed and not placed in STM.
    - Predecessor shedding can occur if the predecessor is in the active task list or in STM.
    - Continuous and Repeating tasks have now been implemented as a shorthand notation for two tasks which are serially connected and where the second tasks is always followed by the first task.

Pass/Fail

Pass

Related PR #s

PR 952

Rule 1 (pg 45)

- "Tasks transferred from the active task list which are interruptible may be halted" In IPME terms, this means that the active task will be interrupted and placed in the Short Term Memory.
- An interrupted task which is not resumable will be placed in STM with time remaining reset.
- Task interruptions will be logged.
- Non-resumable tasks will be flagged when interrupted.
- % processing complete will be logged at time of interruption.

Pass/Fail

Pass

Related PR #s

PR 1552, 1551, 1484, 1480

Rule 2 (pg 46)

- Task delays will be logged
- Tasks are serviced in order of highest priority
- Uninterruptible tasks will be highest priority once they have started

Rule 3 (pg 46)

- The precedence relationship of uninterruptible tasks must be considered
- A pair will attempt to use alternate limb before rejecting it for activation if the primary limb was in conflict. This will happen in this order:
    - Task 1 – Primary -> reject
    - Task 1 – alternate ->accept or reject
    - Task 2, where Task 2 is lower priority than Task 1

Pass/Fail

Fail: PR 2229 (non-preferred channels are not working)

Related PR #s

PR 1552, 2229

Rule 4 Tie Breakers (pg 46)

- Originally scheduled start time
- Least processing time remaining
- According to least interference
- Random

Rule 6

- Externally cued vision tasks in the temp queue (passed the Probability of detection matrix) will always be added to the active queue.

  Could not be verified due to PR 1561.

Rule 7 STM (pg 47)

- STM defaults to 3
  Verified
- STM size is editable
- STM does not include active tasks
- Order of shedding from STM
    - Sheddable tasks: lowest -> highest priority
    - Then non-sheddable tasks: lowest -> highest
    - Ties broken by:
        - Most processing time remaining
        - According to most interference
        - Random
- Tasks shed will be logged
- Tasks partially serviced when shed will have their processed time (% complete) logged
- Predecessor shedding will be logged

Pass/Fail

Pass

Related PR #s

PR 1504, 1587, 1479

TASK SHED

- Three options:

- Branch terminates
- Another task starts (deterministic, conditional, or probabilistic)
- Mission failure
- Following tasks should not be started before the shed task was due to finish
  - Check which path
  - Check which Task to follow

TABLE 11 PR 1479

Verify everything in Table 11.

Compatible task pairs (pg 47)

- This method of altering ip_cij* values will override all other changes to ip_cij* variables.
- Selecting two tasks should default to their cognitive interference value.

Verify this order:

1. Matrices
2. Stressors
3. TPMs
4. Conflict transforms
5. Compatible task pairs

This order could not be verified due to PRs where use of the TPMs causes errors.

Pass/Fail

Fail: PR 1538 (IP function error)

Related PR #s

PR 1538, 1567

Restart/Resume

- Restarted and resumed tasks do not show correct reason on their start line in the IP Results (ipr).
- Are resumable tasks marked as "resume" in IP Results (ipr)? No.
- Are restartable (non-resumable) tasks marked as "restarted" in IP Results (ipr)?
- Should we allow an interrupted task to also be delayed? Keith: No. Brad: Yes. PR 1484.

Pass/Fail

Pass

Related PR #s

PR 1480, 1551, 1484

VISUAL DETECTION TASKS (pg 48) PR 1481

- Probability of detection matrix determines the likelihood that task will enter temp queue.
- Central tasks take precedence over peripheral.
- An attempt to add the detection task to the temp queue will be made only after all other internally cued task selections have been made – to determine the point of goal-directed vision.
  - What does this mean?
  - What does "internally cued task selections have been made" mean?
- Held in system queue until externally cued vision detection tasks enter temp queue
- Attempts will be made to add task to temp queue during the time the stimulus is present.

- If the task fails to enter temp queue, this is logged as “failure to detect”
- If the event signaling presence of the stimulus leaves system memory before entering temp queue, this is a “missed detection”
    - How do “failure to detect” and “missed detection” differ?
- Externally cued vision detection tasks will be “serviced immediately” once they enter the temp queue
    - Does this mean they will always be made active?
    - Or does this mean they get to ops full attention?
    - What exactly does “serviced immediately” mean?

Pass/Fail

Pass

Related PR #s

PR 1544, 1483

Ext Cued Auditory Tasks (48-49)

- Externally cued auditory tasks not processed at the time they occur will be shed.
- Externally cued auditory tasks never enter the system queue.
- Externally cued auditory tasks enter the temp queue immediately.
- Externally cued speech should be processed immediately or forcibly shed and logged as a missed communication.
- Externally cued auditory tasks in category 2 (incidental speech) or category 4 (attended speech) will be shed on the first attempt to transfer from temp queue to STM. This is also true for category 3 (auditory pattern).
    - So does this mean that Auditory Category 1 and Category 5 can never be externally cued?

Pass/Fail

Unknown

Related PR #s

Unknown

CONTINUOUS/REPEATING TASKS

- For difficulty: $0 < D \leq 1$
- Need to test Task Start /Task End event
- Verify that description of repeating task timing info in manuals matches description in paragraph 1 on pg 50.

Pass/Fail

Pass

Related PR #s

PRs 653, 952

INSTANTANEOUS TIME PRESSURE

Task ITP

- For a task: $\frac{\text{processing time remaining}}{\text{Time to latest processing time}}$

- If a task's ITP > 1 (not = 1) then need to examine forcibly shed checkbox
- If the box is checked, shed the task, log it as forcibly shed in IP Results (ipr) file, also set task status to forcibly shed.
- If box is not checked, retain task in STM queue and log it as late in IP Results (ipr) file. Set task status to late.
- If late_start (task_id) = = TRUE then by definition task_status (task_id) = late or forcibly_shed
- (pg 51) If a task is started at or after the latest processing time, then a new latest processing time will be calculated such that Task ITP = 1.
- Late_start (task_id) will not be reset to false in the above case.
- Verify formula for Task ITP of continuous tasks.
- Verify formula for Task ITP of repeating tasks.

Pass/Fail

Fail: PR 1535

Related PR #s

1549, 1548, 1537, 1535, 1492, 1463

PEAK ITP

- For an operator: max of all operator tasks in STM and active queues

Pass/Fail

Pass

Related PR #s

N/A

TASK PRIORITY (51-53)

- Interruptibility should be disabled for all tasks with Task ITP ≥ Serial TP (default 0.8)
- Sheddable tasks do not contribute to peak ITP.
- Tasks with the highest task ITP get serviced first.
- Priority values > 1 are allowed
- Externally cued visual detection tasks do not contribute to ITP or occupy STM.
- Externally cued auditory tasks should be designated Priority category 1 in Table 7.
    - Should this be enforced?
    - Forcibly shed if not processed immediately
- Verify Table 7

The above items could not be verified due to existing PRs.

CRITICAL TP (pg 55)

- TP crit default should be 0.7 Pass
- This is called TP (overload) on the interface. Pass
- The task performance modifier should be calculated from the moving average of peak ITP (pg 57).
- Moving average window default value of 1 minute. Pass.
- Should be modifiable to allow a window width of zero. Pass
    - What does a window width of zero mean?
- Should we allow a resolution > the window time?
    - Example: resolution=80 window=30

Unsuccessful Completion (pg 58)

- Says that there should be a flag for unsuccessful completion. This is similar to task failure, so is this intended to reuse the task failure mechanism?

  This has been accomplished with the taskStatus( ) function.

  Pass/Fail

  Pass

  Related PR #s

  N/A

TASK SHEDDING (pg 62)

- Shed tasks will be logged, noting the criticality of the info: mission performance, situation awareness, etc.

  Pass/Fail

  Pass

  Related PR #s

  N/A

Temporary Sheddable Selections

- If a sheddable task changes temporarily to non-sheddable (through conflict transform or TPM) then it should contribute to ITP.

  This could not be verified due to existing PRs relating to TPMs.

TPMs (pg 63)

There were many problems relating to the TPMs that prevented them from being properly verified.

Pass/Fail

Fail: PR 1540 (either or "either" is OK with F2 check)

Related PR #s

PR 1826, 1797, 1565, 1554, 1521, 1520, 1500, 1487, 1540

Operators

- +, -, *, ÷
- logarithmic
- trigonometric
- polynomial
- if – then – else
- = !=
- <, <=, >, >=

These operators should be available in TPM expressions as a minimum. Yes.

Requires access to specific variables (pg 64) /IP fuctions

- # of interruptions
  - IPinterruptions(id)

  Pass/Fail

Pass

Related PR #s

PR 1492, 1518

- Mean instantaneous time pressure (for an operator)
  - operatorname.MeanTimePressure

Pass/Fail

Pass

Related PR #s

N/A

- Time since task segment commenced
  - IPsegmentTime(id)

Pass/Fail

Fail: PR 1534 (IPsegmentTime error)

Related PR #s

PR 1534

- List of all active tasks
  - Collected in the ipr file at each time step

Pass/Fail

Pass

Related PR #s

N/A

- List of all tasks in STM
  - Collected in the ipr file at each time step

Pass/Fail

Pass

Related PR #s

N/A

- # of unsuccessful scheduling attempts
  - IPattempts(id)

Pass/Fail

Pass

Related PR #s

PR 1490

- % processing complete
  - IPpercentComplete(id)

Pass/Fail

Pass

Related PR #s

PR 1489

  - taskStatus( )

Pass/Fail

Pass

Related PR #s

  - N/A

- Elapsed time since first scheduled start time of a task
    - IPelapsedTime(id)

Pass/Fail

Fail: PR 1536 (IPelapsedTime error)

Related PR #s

PR 1536, 1494

- Task categories:
    - T1) vision interference
        - ip_cij_visual
    - T3) auditory interference
        - ip_cij_auditory
    - T4) cognitive interference
        - ip_cij_cognitive
    - T5) psychomotor interference
        - ip_cij_psychomotor
    - T6) misc interference
        - ip_cij_misc

Pass/Fail

Fail: PR 1972 (IPsetCategory error), 1971 (IPcategory error), 1496 (IPsetCateegory error), 1568 ((IPsetCategory error)

Related PR #s

PR 1492, 1496, 1561, 1568, 1972, 1971

- T7) scheduling/priority category
    - IPpriorityCategory(id, value)
    - IPsetPriorityCategory(id, value)

Pass/Fail

Fail: PR 1533 (IPsetPriorityCategory error)

Related PR #s

PR 1492, 1533

Other IP functions not specifically requested

- IPcategory(case | id, domain, value)
- IPlateStart(id)
- IPlimb(id, value)
- IPsetCategory(id, domain, value)
- IPtask(case, idlist)
- IPtimePressure(id)
- IPvisualArea(id)

Pass/Fail

Fail PR 1972, 1971, 1538, 1535 (all are IP function errors)

Related PR #s

PR 1972, 1971, 1538, 1537, 1535, 1530, 1519, 1492

TPM characteristics

- If a TPM is allowed to modify a variable with a defined range, the software will check the variable for a valid value
    - Interference values {0-1}
    - Probabilities {0-1}

- Default all TPMs to disabled
    - Enable at operator or task level
    - If "on" at operator level must also be "on" at task level
- Can Crew, Env, PSF, MeasSuite, Snapshots, and Display Vars access special IP/PCT variables? No.
- TPM expressions should be modifiable at the task level
- Use of the * to denote task id in global TPM will result in using local task id at task level
    - This has been replaced with $

These items could not be tested due to existing PRs related to TPMs.

TIME PRESSURE/SPEED/ACCURACY TPM

- Where TP is used to establish the value of a TPM, use the operator's mean TP

These items could not be tested due to existing PRs related to TPMs.

FORGETTING TPM/MEMORY (pg 65)

- How is the forgetting field actually implemented?
    - What does the # in this field do? Documented.
    - Is it the memory decay a constant? Yes.
        - If yes, then verify formula
            - $1\text{-}1.3591e^{-+/\sigma}$
    - Should be able to access ip_prob_success variable
        - How does ip_prob_success relate to task failure mechanism?
    - It should be possible for analyst to program other memory relationships. Yes, could be done through Scheduling Effect field.

Pass/Fail

Pass

Related PR #s

PR 1666, 1546, 1487

PREDECESSOR TASKS/TASK HISTORY TPM (pg 64)

- Needs to be able to modify ip_task_time
- Needs to be able to modify ip_prob_success
- Uses IP functions like IP status and IPlateStart

Pass/Fail

Pass

Related PR #s

PR 1547, 1546, 1492

TASK PRIORITY TPM (pg 66)

- Uses IP functions to modify task priorities of this task or other tasks as specified by the task_id in the IP function calls
- Allow use of wild card "$"
- Previously set values for interruptibility and sheddability will be retained, even if they disagree with new priority category
- Priority change will only impact task deadline
    - What is the variable that will be modified here?

Pass/Fail

Pass

Related PR #s

1521, 1492

SPEED/ACCURACY TPM (time pressure)(pg 64)

- Needs to be able to use ip_average_tp variable to represent mean peak ITP
    - What variable is being modified here?

These items could not be tested due to existing PRs related to TPMs.

Pass/Fail

Fail: 1538 (IP function error)

Related PR #s

PR 1538

EXPERIENCE & APTITUDE TPM

- Looks at values of operator variables
- Modifies ip_task_time
- Modifies cognitive category of this task
    - Only allowable changes: Category 1 –> Category 4 or Category 5; Category 4 –> Category 5
    - What is the syntax to change the cognitive category?
        - IPsetCategory

- Why are the Experience &Aptitude TPMs separated on the user interface?

These items could not be tested due to existing PRs related to TPMs.

Pass/Fail

Fail: PR 1538 (IP function error)

Related PR #s

1538, 1547

NON-PREFERRED CHANNEL/NON-PREFERRED LIMB TPM (pg 68)

- Assesses a time penalty for using alternate limb when primary limb is in use
- Default penalty 10%
- Uses IP functions like IPlimb
- Modifies ip_task_time variable value

****Where is ip_critical_tp used?

TPMs applied at "Begin"	Applied at "Start"
- Aptitude	- Memory/Forgetting
- Experience	- Task Priority
- Non-pref. limb	- Visual Detection
- Stressors	
- Task History	
- Speed/Accuracy	

- TPM		Acts On
- Aptitude	ip_task_time	ip_prob_success task cognitive.category.*
- Experience	ip_task_time	ip_prob_success task
- Stressors	ip_task_time	ip_prob_success task
- Task History	ip_task_time	ip_prob_success task
- Speed/Accuracy	ip_task_time	ip_prob_success task
- Forgetting/Memory	Probability. of shedding from STM*	
- Task Priority	task priority category*	
- Visual Detect	ip_prob_detect	

*What are the actual variables for use in the syntax?

IPsetCategory

IPsetPriorityCategory

- TPMs should only apply if an operator is assigned to the task
- Should allow global enable/disable of each TPM prior to execution
    - i.e., one checkbox should disable all Aptitude TPMs for all tasks

These items could not be tested due to existing PRs related to TPMs.

Pass/Fail

Fail: PR 1496 (IP function error), 1533 (IP function error), 2229 (non-preferred channels are not working correctly)

Related PR #s

PR 1544, 1496, 1520, 1533, 1548, 2229

TPM LOGGING (pg 71)

- Anytime a TPM is active, this should be recorded
    - Triggering conditions
    - Parameters

These items could not be tested due to existing PRs related to TPMs.

Pass/Fail

Fail: PR 1496, 1533 (IP function errors)

Related PR #s

PR 1544, 1496, 1520, 1533, 1548

STRESSORS (pg 68)

- May impact task completion times (ip_task_time)

- Minimum of 5 stressors allowed
  - Names assigned at operator level
  - Values assigned at task level
- TPMs should be allowed to use values from stressors
  - Stressors should be evaluated before TPMs
  - How do you access stressor values in TPMs?
- May also modify ip_prob_success
- What is the input?
  - Is it environment variables?

These items could not be verified due to existing PRs related to TPMs.

COMBINING STRESSORS (pg 69)

- Stressors will be added together
- Resultant value will be multiplied by ip_task_time
- (pg 70) “Probabilities will be multiplicative”
  - Probabilities of what?
- Each time the TPMs and stressors are called, the original values of all variables (ip_prob_success, ip_task_time, etc.) will be used to avoid compounding the results
- TPMs and generally called whenever a task first becomes available
  - Is this the temp queue?
  - In some cases, the TPMs are applied each time the allocation of attention module is called.
    - What are these special cases?

These items could not be verified due to existing PRs related to TPMs.

DATA ENTRY

- Verify all defaults pgs 73-79 against software
- Only appropriate data fields should be enabled
  - i.e. if vision not selected globally, no vision fields should be enabled at task level
  - Any previous entered data should be preserved
- Resume penalty should be at task-level
  - Default resume penalty should be set globally
- Each vision task must be assigned a location.
- What happens if the task is assigned to operator “none”?
  - Is there STM shedding?
  - Is there ITP?
  - Is there structural (physical) interference?

- The visual detect TPM is supposed to only be enabled for “visual detection tasks”
  - Is a “visual detection task”

    A) scheduling/Priority Category 9?

    B) vision domain enabled?

    C) both a & b

- User should be allowed to replace existing data with defaults at anytime

Some of these items could not be verified due to existing PRs related to TPMs.

Pass/Fail

Fail: PR 1550 (disabling channels globally should disable channels locally)

Related PR #s

PR 1467, 1550, 1553, 1554, 1555, 1641, 1676, 1475, 1454, 1565, 1479

COMPATIBLE TASK PAIRS (pg 79)

- The temporary coefficient is the new cognitive interference coefficient.
- All other interference coefficients are set to zero.
- Should default to existing cognitive cij value.

Pass/Fail

Pass

Related PR #s

1567

GLOBAL DISABLING (pg 79)

- V, A, P, M should be enabled/disabled for an operator.
  - Fail – can only be enabled/disable at the system level.
- Cognitive must always be enabled at the operator level.

  It was decided that this option was superceded by debugging requirements to turn this channel on/off. Pass.
- All domains should be allowed to be enabled/disabled at global level

  Pass
- What does n=10 represent as applies to the length of the moving average?
- What is meant by "snapshots" in the time resolution?
- Must be able to reset to original defaults
  - This is satisfied by the "Factory Defaults" button.
- Should be able to save a new set of defaults
  - Satisfied by "Save as Default" button.

Pass/Fail

Fail

Related PR #s

PR 1550

Reports

REPORT ON DATA ENTRY (pg 80)

- Report should be generated listing all cases where user has not committed to the entered (or default) values.
- Once agreed to, those cases will be removed from successive report generations
  - Satisfied by the "Unconfirmed data Report"

REPORTING REQUIREMENTS (pg 81)

- All text reports should be importable into spread sheets

  Pass. Reports can be viewed in Microsoft Excel or Open Office spreadsheet application.
- All graphical reports should be importable into word processing packages

Pass. Reports can be viewed in Microsoft Word or Open Office word processing application.

- The Short-term Memory, Mean Time Pressure, Tasks Shed, Tasks Delayed, Tasks Interrupted, and Summary Performance Reports are now located in HAWK.

Graphical 1 – meantime pressure (+/- 1SD) vs. normalized time

- Currently a text report

Pass/Fail

Pass

Related PR #s

PR 1445

Graphical 2 – STM size vs. normalized time as a %

- Needs to handle STM>3
- Reasons for shedding
- Criticality
- Currently a text report

Pass/Fail

Pass

Related PR #s

PR 1801, 1445, 1950

Text 3 – Tasks shed vs. normalized time as a %

- Reasons
- Criticality

Pass/Fail

Pass

Related PR #s

PR 1445, 1588, 1843, 1950

Text 4 – Tasks Interrupted vs. normalized time as a %

- Reasons
- Criticality

Pass/Fail

Pass

Related PR #s

PR 1445, 1588, 1843, 1950

Text 5 – Tasks Delayed vs. normalized time as a %

- Reasons
- Criticality

Pass/Fail

Pass

Related PR #s

PR 1445, 1588, 1843, 1950, 1828

Text 6 – Modality Interference

- Pair of tasks
- Interference for each domain
- Does this include compatible task pairs?
- If task pair exists in Compatible Task Pairs, which interference coefficient is shown?

Pass/Fail

Fail

Related PR #s

PR 1871

Text 7 – Summary Performance

- Normalized time
- Shed %
- Delayed %
- Interrupted %
- Successful tasks

Pass/Fail

Pass

Related PR #s

PR 1828, 1843, 1588

(pg 82) Text 8 – Display surface/psychomotor channel vs. normalized time

- Visual display surface attended to
- Psychomotor channel (list each & primary/alternate)?
- Does not specify as a %

This report has not been implemented. This information is currently available in the ipr file.

Text 9 – Unconfirmed Data Report

- List of tasks

Pass/Fail

Pass

Related PR #s

N/A

Multiple Cognitive Categories

This IP/PCT scheduler update should allow the user to select more than one cognitive category for each task in an IPME model.

Verify that the c_{ij} cognitive value is the maximum of all cognitive categories for two tasks and that all cognitive categories are listed in the ipr file

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.

3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.
   a. Task 1->Task 2
   b. Task 1->Task3
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. For Task 3, select Cognitive Categories 1 and 3.
11. Execute the model, collecting the *.ipr file.
12. View the *.ipr file.
13. Notice that the interference value for these two tasks is 1 and that all categories are listed in the *.ipr file.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Verify that the changes to the default task IP components settings are updated

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Change the default settings on Define | Global IP Parameters | Task components to Cognitive Categories 1 and 4.
6. Click OK.
7. Draw a task.
8. Verify that Categories 1 and 4 are selected for the task.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR 1968

Verify that the c_{ij} cognitive value is superceded by selections in the compatibility tab

Testing Procedure

1. Create a new system.

2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.
    a. Task 1 -> Task 2.
    b. Task 1 ->Task 3.
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. For Task 3, select Cognitive Categories 1 and 3.
11. In the compatibility tab, select Task 2 and Task 3.
12. Set the interference value to 0.4.
13. Execute the model, collecting the *.ipr file.
14. View the *.ipr file.
15. Notice that the interference value for these two tasks is 0.4.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Multiple cognitive categories can be saved as the default settings

Testing Procedure

1. Repeat the steps in tests 2.2.23.2 and 2.2.23.3.
2. Click Save As Defaults in the Define | Global IP Parameters window.
3. Click Factory Defaults.
4. Verify that the original settings are restored.
5. Click Reset to Defaults.
6. Verify that all of your changes from tests 2.2.23.2 and 2.2.23.3 are restored.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Multiple cognitive categories can be saved to the Master Database

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.

3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw a network.
6. In the network, add a task.
7. Select Categories 1 and 4 for the Cognitive categories for the task.
8. Save the network to the Master Database.
9. Load the network from the Master Database.
10. Verify that Categories 1 and 4 are selected in the task in the loaded network.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR 1969

The IPcategory function will return “true” for all categories selected.

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw a network.
6. In the network, add a task.
7. Select Categories 1 and 4 for the Cognitive categories for the task.
8. Enable the Speed/Accuracy TPM.
9. In the Speed/Accuracy TPM, enter “debug(1, IPcateory($, cognitive, automatized), 0, 0);”
10. Click OK.
11. Execute the model.
12. View the debug statement.

Test Platform

RedHat 7.2

Pass/Fail

Fail: PR 1971 (IPcategory function error)

Related PR #s

PR 1971

Model Used

development/IPmodels/IPcategory_error_sys.dat

The IPategory function will return “true” for all categories selected using the IPsetCategory function.

Testing Procedure

1. Create a new system.

2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw a network.
6. In the network, add a task.
7. Select Categories 1 and 4 for the Cognitive categories for the task.
8. Enable the Speed/Accuracy TPM.
9. In the Speed/Accuracy TPM, enter "IPsetCategory($, cognitive, verbal); debug(1, IPcateory($, cognitive, verbal), 0, 0);"
10. Click OK.
11. Execute the model.
12. View the debug statement.

Test Platform

RedHat 7.2

Pass/Fail

Fail: PR 1972 (IPsetCategory error)

Related PR #s

PR 1972

Model Used

development/IPmodels/IPsetCategory_error_sys.dat

The ip_cij_cognitive variable returns the correct value.

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.
   a. Task 1 -> Task 2.
   b. Task 1 ->Task 3.
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. For Task 3, select Cognitive Categories 1 and 3.
11. In a new conflict transform, enter "debug(1, ip_cij_cognitive, 0, 0);"
12. Execute the model.
13. View the debug statement.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Model Used

development/IPmodels/ConflictTrans_ipcijcognitive_sys.dat

The Modality Interference report returns the maximum of all task cij values.

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.
   a. Task 1 -> Task 2.
   b. Task 1 ->Task 3.
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. For Task 3, select Cognitive Categories 1 and 3.
11. In the Modality Interference report, select Tasks 2 and 3.
12. View the interference value.
13. View the results.
14. Change Task 3 to Cognitive Category 1 only.
15. Repeat steps 11-12.
16. View the results.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Multiple cognitive categories are imported/exported using the .dat file.

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.

   a. Task 1 -> Task 2.
   b. Task 1 ->Task 3.
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. Export the System using the .dat file method.
11. Import the system, renaming as necessary.
12. View the Task IP settings for Task 2.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Multiple cognitive categories are imported/exported using the XML files.

Testing Procedure

1. Create a new system.
2. Change mode to IP/PCT.
3. Add Operator1 to the crew model.
4. Open the task network model.
5. Draw three tasks.
6. Connect them as follows.
   a. Task 1 -> Task 2.
   b. Task 1 ->Task 3.
7. Change the decision to Multiple.
8. Assign Tasks 2 and 3 to Operator1.
9. For Task 2, select Cognitive Categories 1 and 3.
10. Export the System using the XML file method.
11. Import the system, renaming as necessary.
12. View the Task IP settings for Task 2.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

HAWK

Execution Viewer (EV)

Ignore Branch Logic

The user should be allowed to select whether an IPME network will ignore branching decision during the creation of an OSD file.

Testing Procedure

1. Select Execute | Execution Settings from the menu.
2. On the Main tab, select "Output OSD (*.osd) File".
3. Make sure "Ignore Branch Logic" is turned off.
4. Click OK.
5. Select Execute | Execute Simulation from the menu.
6. Click "Pause/Step" on the Execution Dialogue.
7. Notice that after a decision, only one task is executed.
8. Launch HAWK.
9. Select Tools | Execution Viewer.
10. In the Execution Viewer, select File | Open.
11. Open "Branch_Logic_net.osd".
12. Select Mode | OSD.
13. Notice that only one task out of each decision pair is listed.
14. In IPME, select Execute | Execution Settings from the menu.
15. On the Main tab, make sure "Output OSD (*.osd) File" is still selected.
16. Select "Ignore Branch Logic".
17. Click OK.
18. Select Execute | Execute Simulation from the menu.
19. Click "Pause/Step" on the Execution Dialogue.
20. Notice that every task is executed.
21. When the model finished executing, click "Dismiss".
22. Open the .osd file in the Execution Viewer.
23. Select Mode | OSD.
24. Notice that all tasks are listed.

Model Used

testmodels2/development/DRDC_Test/Branch_Logic.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Graphical Debugging Tool

Start Tasks

The Graphical Debugging Tool diagram should include tasks triggered via the start ( ) function.

Testing Procedure

1. Select Execute | Execution Settings from the menu.
2. On the Main tab, select "Output OSD (*.osd) File".
3. Make sure "Ignore Branch Logic" is turned off.
4. Click OK.
5. Select Execute | Execute Simulation from the menu.
6. Click "Start/Resume".
7. When the model finished executing, click "Dismiss".
8. Launch HAWK.
9. Select Tools | Execution Viewer.
10. In the Execution Viewer, select File | Open.
11. Open "Start_Stop_net.osd".
12. Select Mode | GDT.
13. Verify that the task 3 started at time 5.0.
14. Verify that task 3 is not connected to a task started before it.
15. Verify that text on task 3 says that it was started by task 2.
16. Verify that task 4 was executed after task 3.

Model Used

testmodels2/development/DRDC_Test/Start_Stop_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR404

Stopped Tasks

The OSD diagram should display a task that was started, but was subsequently stopped.

Testing Procedure

1. Select Execute | Execution Settings from the menu.
2. On the Main tab, select "Output OSD (*.osd) File".
3. Make sure "Ignore Branch Logic" is turned off.
4. Click OK.
5. Select Execute | Execute Simulation from the menu.
6. Click "Start/Resume".
7. When the model finished executing, click "Dismiss".
8. Launch HAWK.
9. Select Tools | Execution Viewer.

10. In the Execution Viewer, select File | Open.
11. Open “Start_Stop_net.osd”.
12. Select Mode | GDT.
13. Verify that the task 5 started at time 5.0.
14. Verify that text on task 5 says that it was started by task 3.
15. Verify that task 5 is not connected to a task started before it.
16. Verify that text on task 5 says that it was stopped by task 3.
17. Verify that there is no task connected to task 5.

Model Used

testmodels2/development/DRDC_Test/Start_Stop_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Debugger must be mode independent

The user should be allowed to use the Graphical Debugging Tool in any simulation mode.

Testing Procedure

1. Select Execute | Execution Settings from the menu.
2. On the Main tab, select “Output OSD (*.osd) File”.
3. Make sure “Ignore Branch Logic” is turned off.
4. Next to Simulation Mode, select “IPME” from the list.
5. Click OK.
6. Select Execute | Execute Simulation from the menu.
7. Click “Start/Resume” on the Execution Dialogue.
8. Launch HAWK.
9. Select Tools | Execution Viewer.
10. In the Execution Viewer, select File | Open.
11. Open “Test_Modes_net.osd”.
12. Select Mode | GDT.
13. Verify that all tasks are listed under the correct operator.
14. Repeat steps 1- 13 for POP and IP/PCT modes.

Model Used

testmodels2/development/DRDC_Test/Test_Modes_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Scheduled Effects Settings

The user should be allowed to select options for viewing shed, interrupted, and delayed tasks. The user should also be able to view the reason for the scheduled effect.

Testing Procedure

1. Select Execute | Execution Settings from the menu.
2. On the Main tab, select "Output OSD (*.osd) File".
3. Make sure "Ignore Branch Logic" is turned off.
4. Next to Simulation Mode, select "IP/PCT" from the list.
5. Click OK.
6. Select Execute | Execute Simulation from the menu.
7. Click "Start/Resume" on the Execution Dialogue.
8. Click OK on the debug statements that appear explaining the scheduled effects.
9. Launch HAWK.
10. Select Tools | Execution Viewer.
11. In the Execution Viewer, select File | Open.
12. Open "Test_OSD_net2.osd".
13. Select Mode | GDT.
14. Verify that all tasks are listed under the correct operator.
15. Select Options | Scheduler Effects.
16. On the Scheduler Effects dialogue, verify that "Shed" is selected under both "Display Tasks:" and "Display Cause:".
17. Select "Interrupted" and "Delayed" for both "Display Tasks:" and "Display Cause:".
18. Click OK.
19. Verify that task 10 says "delayed: scheduler".
20. Verify that tasks 3 and 6 say "interrupted: scheduler".

Model Used

testmodels2/development/DRDC_Test/Test_Sched_Effects_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass PR388, PR405
Fail PR419

Related PR #s

PR388, PR405, PR419

Network Drawing Tool

HAWK should provide the user access to task network drawing capabilities similar to that found in IPME. The following tests are all performed on the task network drawing tool. To access the task network drawing:

1. In the project view window, expand the project.
2. Right click on the system.
3. Select "Add Model → Task Network Model" from the menu.

4. Expand the task network model.
5. Right click on Networks.
6. Select "Edit Layout" from the menu.
7. The Tool Palette, Overview Window, and Drawing Window will open.

Draw Network Objects

Palette Tool

Place each network object to verify that drawing objects on the palette tool work properly. Verify that objects are drawn correctly and appear in the appropriate area.

37. Networks
38. Tasks
39. Paths

Testing Procedure

1. Select a task from the Task Network Palette.
2. Drag the task to an area next to Start in the drawing window.
3. Verify that a task is placed in the drawing window.
4. Verify that the task is listed in the project view window.
5. Click on the task and drag it around the window.
6. Verify that the task moves freely.
7. Click on Start and drag a line to the task.
8. Verify that a path was created.
9. Drag a network from the palette tool to a spot next to the first task.
10. Verify that a network is placed in the drawing window.
11. Verify that the network is listed in the project view window.
12. Click on the network and drag it around the window.
13. Verify that the network moves freely.
14. Click on the diamond part of the task and drag a line to the network.
15. Double-click on the network.
16. Drag a task inside the network.
17. Verify that a task is placed in the drawing window.
18. Verify that the task is listed in the project view window.
19. Connect it to the external in.
20. Double click on the external in.
21. Verify that the parent network is still correct.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR350, PR353, PR395, PR358, PR349

Right Click

Place each network object to verify that drawing objects work properly. Verify that objects are drawn correctly and appear in the appropriate area.

1. Networks
2. Tasks
3. Paths

Testing Procedure

1. In the drawing window, right click.
2. Select Add Task from the menu options.
3. Verify that a task is placed in the drawing window.
4. Verify that the task is listed in the project view window.
5. Click on the task and drag it around the window.
6. Verify that the task moves freely.
7. Click on Start and drag a line to the task.
8. Verify that a path was created.
9. In the drawing window, right click.
10. Select Add Network from the menu options.
11. Verify that a network is placed in the drawing window.
12. Verify that the network is listed in the project view window.
13. Click on the network and drag it around the window.
14. Verify that the network moves freely.
15. Click on the diamond part of the task and drag a line to the network.
16. Double-click on the network.
17. In the drawing window, right click.
18. Select Add Task from the menu options.
19. Verify that a task is placed in the drawing window.
20. Verify that the task is listed in the project view window.
21. Connect it to the external in.
22. Double click on the external in.
23. Verify that the parent network is still correct.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR350, PR353, PR395, PR358

Queues

Verify that queues appear on the right tasks and that they move with the tasks. Verify also that queues are listed in the project window.

Testing Procedure

1. Add a task to the network.

2. Right click on the task.
3. Select Add Queue from the menu options.
4. Verify that a queue is added to the task.
5. Verify that a queue appears in the project window underneath the correct task.
6. Click on the task and drag the task around the window.
7. Verify that the queue moves with the task.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Decisions

Verify that decisions appear when a task is connected to multiple objects.

Testing Procedure

1. Add three tasks to the network.
2. Connect task1 to tasks 2 and 3.
3. Verify that the decision shows an “M” for multiple (the default decision type).
4. Verify that the decision appears beneath the correct task in the project window.
5. Add two tasks and a network to the drawing window.
6. Connect the first task to the network and the other tasks.
7. Verify that the decision shows an “M” for multiple.
8. Verify that the decision appears beneath the correct task in the project window.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR356, PR401, PR373, PR372, PR370, PR355, PR394

Resizable Task and Network Nodes

Verify that task and networks are resizable.

Testing Procedure

1. Repeat steps 1-8 of 2.3.2.1.2.
2. Click on the task.
3. Verify that blue squares appear around the task.
4. Move the mouse over a blue square.
5. Verify that an arrow appears.
6. Click and drag the square to get the desired size or shape.
7. Verify that the name of the task appears as the size grows larger.
8. Add a network.

9. Click on the network.
10. Verify that blue squares appear around the network.
11. Move the mouse over a blue square.
12. Verify that an arrow appears.
13. Click and drag the square to get the desired size or shape.
14. Verify that the name of the network appears as the size grows larger.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR371

Edit Network Objects

Verify that data can be entered from the network drawing tool. Verify that the data is visible in the tree view window. Verify also that sub-networks can be edited in a separate window while still viewing the parent network.

Double Click

Verify that double click performs the required action for that object.

1. Networks
2. Tasks
3. External Ins/Outs
4. Decisions
5. Queues

Testing Procedure

1. Add three tasks.
2. On task 1, attach a queue.
3. Draw paths from task 1 to tasks 2 and 3.
4. Add a network.
5. Connect task 3 to the network.
6. Add a task.
7. Connect the network to the new task.
8. Double click on the network.
9. Verify that the sub-network is displayed.
10. Add a task.
11. Connect it to the external in and the external out.
12. Double click the external in.
13. Verify that the parent network is displayed.
14. Double click on the network.
15. Verify that the sub-network is displayed.
16. Double click on the external out.

17. Verify that the parent network is displayed.
18. Double click on a task.
19. Verify that the edit dialogue is displayed.
20. Click “Cancel” to close dialogue.
21. Repeat steps 19, 20, and 21 for queue and decision.
22. Verify that the edit dialogues are displayed.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR353, PR360, PR361, PR403

Right Click

Verify that right click displays an edit menu through which edit dialogues can be accessed.

1. Networks
2. Tasks
3. External Ins/Outs
4. Decisions
5. Queues
6. Paths

Testing Procedure

1. Add a task and a network.
2. Right click on the network.
3. Select Edit Network.
4. Verify that the edit dialogue appears.
5. Right click on a path.
6. Select “Delete Path”.
7. Verify that the path is deleted.
8. Right click on a task.
9. Select Edit Task.
10. Verify that the edit dialogue displays.
11. Click “Cancel” to close dialogue.
12. Repeat steps 8 – 11 on queues and decisions.
13. Verify that the edit dialogues are displayed.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR362

Tool Tips

Verify that tools tips are available to display full text of the task or network.

1. Networks
2. Tasks
3. External In/External Out

Testing Procedure

1. Add a task.
2. Add a network.
3. Right click on the task.
4. Select Edit Task.
5. Rename the task "TestToolTip1".
6. Click OK.
7. Right click on the network.
8. Select Edit Network.
9. Rename the network "TestToolTip2".
10. Click OK.
11. Move the pointer over the task.
12. Verify that a tool tip displays showing the name and ID of the task.
13. Move the pointer over the network.
14. Verify that a tool tip displays showing the name and ID of the network.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR364

Cut/Paste Network Objects

Cut/paste will be performed on each object and in groups of different objects.

Verify that objects are pasted with the correct data. Verify also that objects have the correct numbering.

1. Networks
2. Tasks
3. Queues
4. Decisions
5. External In/External Out

Testing Procedure

1. Add a task.
2. Right click on the task.
3. Select "Cut Task" from the list of options.
4. Verify that the task has been removed from the drawing window and from the project window.

5. Right click on the drawing window.
6. Select “Paste Task” from the list of options.
7. Verify that a task appears in the drawing window and in the project window.
8. Verify that the task is designated with the next available task ID.
9. Add a network.
10. Double click on the network.
11. Add a task inside the network.
12. At the parent level, right click on the network.
13. Select “Cut Network” from the list of options.
14. Verify that the network and the task contained in the network have been removed from the drawing window and from the project window.
15. Right click on the drawing window.
16. Select “Paste Network” from the list of options.
17. Verify that the network and the task contained in the network appear in the drawing window and in the project window.
18. Verify that the network and task are designated with the next available ID numbers.
19. Repeat tests with groups of tasks and networks including queues and decisions.
20. Verify that connections, external ins, and external outs are correct.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR370, PR372, PR395

Copy/Paste Network Objects

Copy/paste will be performed on each object and in groups of different objects.

Verify that objects are pasted with the correct data. Verify also that objects have the correct numbering.

Testing Procedure

1. Add a task.
2. Right click on the task.
3. Select “Copy Task” from the list of options.
4. Verify that the task remains in the drawing window and in the project window.
5. Right click on the drawing window.
6. Select “Paste Task” from the list of options.
7. Verify that a copy of the task appears in the drawing window and in the project window.
8. Verify that the new task is designated with the next available task ID.
9. Add a network.
10. Double click on the network.
11. Add a task inside the network.
12. At the parent level, right click on the network.

13. Select “Copy Network” from the list of options.
14. Verify that the network and the task contained in the network remain in the drawing window and in the project window.
15. Right click on the drawing window.
16. Select “Paste Network” from the list of options.
17. Verify that a copy of the network and the task contained in the network appear in the drawing window and in the project window.
18. Verify that the new network and task are designated with the next available ID numbers.
19. Repeat tests with groups of tasks and networks including queues and decisions.
20. Verify that connections, external ins, and external outs are correct.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR370, PR372, PR395

Delete Network Objects

Delete will be performed on each object and in groups of different objects. Verify that delete works on all network objects. Verify that objects are removed from both the drawing window and the tree view.

Testing Procedure

1. Add a task.
2. Right click on the task.
3. Select “Delete Task” from the list of options.
4. Verify that the task has been removed from the drawing window and from the project window.
5. Add a network.
6. Double click on the network.
7. Add a task inside the network.
8. At the parent level, right click on the network.
9. Select “Delete Network” from the list of options.
10. Verify that the network and the task contained in the network have been removed from the drawing window and from the project window.
11. Verify that the new network and task are designated with the next available ID numbers.
12. Repeat tests with groups of tasks and networks including queues and decisions.
13. Verify that objects are removed from both the drawing window and from the project window.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR354

External Ins/External Outs

Connections to a Root Task

Test the creation of decisions from a task connected to a network. Also verify that external ins are drawn properly.

Testing Procedure

1. Add a task and a network.
2. Connect the task to the network.
3. Double click the network.
4. Verify the existence of the external in with the connected task’s ID.
5. Add a task.
6. Connect the task to the external in.
7. Move back to the parent level.
8. Verify that no decision was created.
9. Double click on the network.
10. Add a task.
11. Connect the task to the external in.
12. Move back to the parent level.
13. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR375

Connections to an Internal Task

Test the creation of decisions from a network connected to multiple tasks. Also verify that external outs are drawn properly.

Testing Procedure

1. Add a network and a task.
2. Connect the network to the task.
3. Double click the network.
4. Verify the existence of the external out with the connected task’s ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Add a task.
9. Connect the network to the newly created task.
10. Double click on the network.
11. Verify that a second external out was created.
12. Connect the task to the second external out.

13. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR359

Internal to Internal Connections

Test the creation of decisions from a network connected to a network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected network's ID.
10. Add a task.
11. Connect it to the external in.
12. Move back to the parent level.
13. Double click on the first network.
14. Verify that the external out now contains the new task ID.
15. Move back to the parent level.
16. Double click the second network.
17. Add a task.
18. Connect it to the external in.
19. Move back to the parent level.
20. Double click the first network.
21. Verify that there are two external outs with the correct task IDs.
22. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Two Level Internal Connections

Variant 1

Test the creation of decisions from a network with a sub-network to another network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click on the sub-network.
8. Verify the existence of the external out with the connected network's ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task's ID.
14. Add a task.
15. Connect it to the external in.
16. Move back to the parent level.
17. Double click on the first network.
18. Verify that the external out now contains the new task ID.
19. Double click the sub-network.
20. Verify that there is no decision on the task.
21. Move up 2 levels to the parent level.
22. Double click the second network.
23. Add a task.
24. Connect it to the external in.
25. Move back to the parent level.
26. Double click the first network.
27. Verify that there are two external outs with the correct task IDs.
28. Double click the sub-network.
29. Verify that the task now has a decision.
30. Verify that there are two external outs with the correct task IDs.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR376

Variant 2

Test the creation of decisions from a network to a network with a sub-network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network’s ID.
5. Add a task.
6. Connect the task to the external out.
7. Move back to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected task’s ID.
10. Add a network.
11. Connect the network to the external in.
12. Double click on the sub-network.
13. Add a task.
14. Connect the task to the external in.
15. Move up 2 levels to the parent level.
16. Double click on the first network.
17. Verify that the external out now contains the new task ID.
18. Verify that there is no decision on the task.
19. Move back to the parent level.
20. Double click on the second network.
21. Double click the sub-network.
22. Add a task.
23. Connect the new task to the external in.
24. Move up 2 levels to the parent level.
25. Double click the first network.
26. Verify that there are two external outs with the correct task IDs.
27. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR376

Variant 3

Test the creation of decisions from a network with a subnetwork to a network with a sub-network. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click on the sub-network.
8. Verify the existence of the external out with the connected network's ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task's ID.
14. Add a network.
15. Connect the network to the external in.
16. Double click on the sub-network.
17. Add a task.
18. Connect the task to the external in.
19. Move up 2 levels to the parent level.
20. Double click on the first network.
21. Verify that the external out now contains the new task ID.
22. Double click the sub-network.
23. Verify that there is no decision on the task.
24. Move up 2 levels to the parent level.
25. Double click on the second network.
26. Double click the sub-network.
27. Add a task.
28. Connect the new task to the external in.
29. Move up 2 levels to the parent level.
30. Double click the first network.
31. Verify that there are two external outs with the correct task IDs.
32. Double click the sub-network.
33. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR376

Variant 4

Test the creation of decisions from a network to a network with a sub-network and a task. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a task.
6. Connect the task to the external out.
7. Move up to the parent level.
8. Double click the second network.
9. Verify the existence of the external in with the connected task's ID.
10. Add a network.
11. Connect the network to the external in.
12. Double click on the sub-network.
13. Add a task.
14. Connect the task to the external in.
15. Move up 2 levels to the parent level.
16. Double click on the first network.
17. Verify that the external out now contains the new task ID.
18. Verify that there is no decision on the task.
19. Move up to the parent level.
20. Double click on the second network.
21. Add a task.
22. Connect the new task to the external in.
23. Move up to the parent level.
24. Double click the first network.
25. Verify that there are two external outs with the correct ID numbers.
26. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR376

Variant 5

Test the creation of decisions from a network with a sub-network to a network with a sub-network and a task. Also verify that external ins and outs are drawn properly.

Testing Procedure

1. Add two networks.
2. Connect the networks.

3. Double click the first network.
4. Verify the existence of the external out with the connected network's ID.
5. Add a network.
6. Connect the network to the external out.
7. Double click the sub-network.
8. Verify the existence of the external out with the connected network's ID.
9. Add a task.
10. Connect it to the external out.
11. Move up 2 levels to the parent level.
12. Double click the second network.
13. Verify the existence of the external in with the connected task's ID.
14. Add a network.
15. Connect the network to the external in.
16. Double click on the sub-network.
17. Add a task.
18. Connect the task to the external in.
19. Move up 2 levels to the parent level.
20. Double click on the first network.
21. Verify that the external out now contains the new task ID.
22. Double click on the sub-network.
23. Verify that there is no decision on the task.
24. Move up to the parent level.
25. Double click on the second network.
26. Add a task.
27. Connect the new task to the external in.
28. Move up to the parent level.
29. Double click the first network.
30. Verify that there are two external outs with the correct ID numbers.
31. Double click on the sub-network.
32. Verify that the task now has a decision.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR376

Basic Destination Split and Merge

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.

2. Connect the networks.
3. Double click the second network.
4. Add two tasks.
5. Connect both tasks to the external in.
6. Move up to the parent level.
7. Double click the first task.
8. Add a task.
9. Connect the new task to one external out.
10. Move up to the parent level.
11. Double click the second task.
12. Verify that there are now two external ins.
13. Verify that each task is connected to an external in.
14. Move up to the parent level.
15. Double click on the first task.
16. Connect the task to the other external out.
17. Move up to the parent level.
18. Double click on the second network.
19. Verify that there is only one external in.
20. Verify that both tasks are connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Multi-level Split and Merge

Variant 1

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the second network.
4. Add a network.
5. Connect the sub-network to the external in.
6. Double click the sub-network.
7. Add two tasks.
8. Connect both tasks to the external in.
9. Move up to the parent level.
10. Double click the first network.

11. Add a task.
12. Connect the new task to one external out.
13. Move up to the parent level.
14. Double click the second task.
15. Verify that there are external ins.
16. Double click the sub-network.
17. Verify that each task is connected to an external in.
18. Move up to the parent level.
19. Double click on the first task.
20. Connect the task to the other external out.
21. Move up to the parent level.
22. Double click on the second network.
23. Verify that there is only one external in.
24. Double click the sub-network.
25. Verify that there is only one external in.
26. Verify that both tasks are connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Variant 2

Verify network-to-network connections with multiple external ins. Verify the splitting and merging of external ins.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the second network.
4. Add a network.
5. Connect the sub-network to the external in.
6. Add a task.
7. Connect the task to the external in.
8. Double click the sub-network
9. Add a task.
10. Connect the task to the external in.
11. Move up to the parent level.
12. Double click the first network.
13. Add a task.
14. Connect the new task to one external out.
15. Move up to the parent level.

16. Double click the second task.
17. Verify that there are two external ins.
18. Verify that the task is connected to one external in and the network is connected to the other.
19. Double click the sub-network.
20. Verify that there is one external in.
21. Move up to the parent level.
22. Double click on the first task.
23. Connect the task to the other external out.
24. Move up to the parent level.
25. Double click on the second network.
26. Verify that there is only one external in.
27. Verify that the network and the task are both connected to it.
28. Double click the sub-network.
29. Verify that there is only one external in.
30. Verify that the task is connected to the external in.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Basic Source Split and Merge

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add two tasks.
5. Connect both tasks to the external out.
6. Move up to the parent level.
7. Double click on the second network.
8. Add a task.
9. Connect the task to one of the external ins.
10. Move up to the parent level.
11. Double click on the first network.
12. Verify that there are two external outs.
13. Verify that each task is connected to one of the external outs.
14. Move up to the parent level.
15. Double click on the second network.
16. Connect the task to the other external in.

17. Move up to the parent level.
18. Double click on the first network.
19. Verify that only one external out is present and that both tasks are connected to it.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Multi-level Source Split and Merge

Variant 1

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add a network.
5. Connect the network to the external out.
6. Double click on the sub-network.
7. Add two tasks.
8. Connect both tasks to the external out.
9. Move up to the parent level.
10. Double click on the second network.
11. Verify that there are two external ins.
12. Add a task.
13. Connect the task to one of the external ins.
14. Move up to the parent level.
15. Double click on the first network.
16. Verify that only one external out is present.
17. Double click on the sub-network
18. Verify that there are two external outs.
19. Verify that each task is connected to one external out.
20. Move up to the parent level.
21. Double click on the second network.
22. Connect the task to the other external in.
23. Move up to the parent level.
24. Double click on the first network.
25. Verify that there is only one external out.
26. Double click on the sub-network.

27. Verify that there is only one external out.
28. Verify that both tasks are connected to the external out.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Variant 2

Verify network-to-network connections with multiple external outs. Verify the splitting and merging of external outs.

Testing Procedure

1. Add two networks.
2. Connect the networks.
3. Double click the first network.
4. Add a network.
5. Connect the network to the external out.
6. Add a task.
7. Connect the task to the external out.
8. Double click on the sub-network.
9. Add a task.
10. Connect the task to the external out.
11. Move up to the parent level.
12. Double click on the second network.
13. Verify that there are two external ins.
14. Add a task.
15. Connect the task to one of the external ins.
16. Move up to the parent level.
17. Double click on the first network.
18. Verify that two external outs are present.
19. Verify that the task is connected to one external out and the network is connected to the other.
20. Double click on the sub-network
21. Verify that there is one external out.
22. Move up to the parent level.
23. Double click on the second network.
24. Connect the task to the other external in.
25. Move up to the parent level.
26. Double click on the first network.
27. Verify that there is only one external out.
28. Verify that both the network and the task are connected to the external out.
29. Double click on the sub-network.

30. Verify that there is only one external out.
31. Verify that the task is connected to it.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR378

Print

Graphical Printing

Verify that all objects are printed correctly.

Testing Procedure

1. Add a task.
2. Double click on the task.
3. Enter a name for the task.
4. Click OK.
5. Expand the task to fit the name.
6. Add two more tasks.
7. Connect these tasks to the first task.
8. Right click on the first task.
9. Select “Edit Decision” from the list of options.
10. Change the decision type to “Tactical”.
11. Add a network.
12. Connect it to the second task.
13. Right click on the network.
14. Select “Edit Network” from the list of options.
15. Enter a new name for the network.
16. Click OK.
17. Expand the network to fit the name.
18. Double click on the network.
19. Add a task.
20. Connect it to the external in.
21. Close the network window.
22. Right click in the parent level window.
23. Select “Print Network” from the list of options.
24. Verify that the print out matches what is shown on the screen.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Textual Printing

Verify that all data are printed correctly.

Testing Procedure

1. Add a task.
2. Double click on the task.
3. Enter a name for the task.
4. Click OK.
5. Add two more tasks.
6. Connect these tasks to the first task.
7. Right click on the first task.
8. Select “Edit Decision” from the list of options.
9. Change the decision type to “Tactical”.
10. Add a network.
11. Connect it to the second task.
12. Right click on the network.
13. Select “Edit Network” from the list of options.
14. Enter a new name for the network.
15. Click OK.
16. Double click on the network.
17. Add a task.
18. Connect it to the external in.
19. Close the network window.
20. Select File | Print Project to File.
21. Enter a name for the file.
22. Click “Save”.
23. View the file in an HTML viewer.
24. Verify that:
    - The task name is correct
    - A Tactical decision is listed on the first task
    - The network name is correct

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

N/A

Undo/Redo Editing

Verify that Undo/Redo works for all edits to drawing. Verify that changes are also made in the tree view window.

Testing Procedure

1. Add a task.
2. Connect the task to the start.
3. Select Edit | Undo.
4. Verify that the connection is removed.
5. Select Edit | Undo.
6. Verify that the task is removed from the drawing window.
7. Verify that the task is removed from the project window.
8. Select Edit | Redo.
9. Verify that the task is added to the drawing window.
10. Verify that the task is added to the project window.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR411

Task Network Overview

Verify that the entire network can be viewed in the Task Network Overview window. Verify also that the network can be navigated using the Task Network Overview window.

Testing Procedure

1. Add three tasks to the network.
2. Verify that tasks can be seen in the Task Network Overview.
3. Drag a task to the bottom right corner of the window.
4. Verify that the task moved in the Task Network Overview.
5. Drag the task to the bottom right of the window until the other tasks are no longer visible in the drawing window.
6. In the Task Network Overview window, click inside the square and drag the square to the top left portion of the window.
7. Verify that the focus in the drawing has returned to the two tasks remaining near the start.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR351, PR357

Windows Menu

Verify that multiple windows are easy to navigate. Verify also that sub-networks can be edited in a separate window while viewing the parent network.

Testing Procedure

1. Add a network.
2. Right click on the network.
3. Select Edit Network.
4. Enter a name for the network.
5. Double click in the network to open a new window.
6. Verify that the name of the network and the network ID are in the window caption.
7. Move the window on the screen.
8. Verify that the parent network is still visible.
9. Select “Windows” from the file menu.
10. Verify that all open windows are listed in the windows menu.
11. Select “NetModel1” from the list.
12. Verify that the window called “NetModel1” received the focus.
13. Select the window for the network added in step 1.
14. Close the window.
15. Select “Windows” from the file menu.
16. Verify that the name of the closed window has been removed from the list.

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR377, PR396

XML Import/Export in HAWK

XML Import/Export should be tested for each type of file: project, system, crew, and task network.

Testing Procedure

1. Select File | Export → Project from the menu.
2. Enter a name for the file at the prompt.
3. Click “Save”.
4. Right click on the project name in the project window.
5. Select “Clear Project” from the list of options.
6. Click “Yes” when asked to continue.
7. Select File | Import → Project from the menu.
8. Select the name of the file.
9. Click “Open”.
10. Verify that all data was imported correctly.
11. Verify that there is no data loss.

12. Repeat test at the system, crew, and task network levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR1874

XML Export from HAWK Import into IPME

The XML Import file format should be tested for each type of import: project, system, environment, crew, task network, PSF.

Basic Export/Import

Testing Procedure

1. Select File | Export → Project from the menu.
2. Enter a name for the file at the prompt.
3. Click “Save”.
4. Open IPME.
5. Select File | Import → Import XML on the main IPME window.
6. Select “Project” from the list of options.
7. Enter the name of the file (or use the “Browse” button to find the file).
8. Click “OK”.
9. Verify that all data was imported correctly.
10. Verify that there is no data loss.
11. Repeat test at the system, environment, crew, task network, and PSF levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

Re-Export from IPME into HAWK

Testing Procedure

Continue from the previous test:

1. Select File | Export → Export XML from the Main IPME window.
2. In the dialogue, select the project name.
3. Enter a name for the file at the prompt.
4. Click “OK”.

5. Open HAWK.
6. Select File | Import → Project from the menu.
7. Select the file name.
8. Click “Open”.
14. Verify that all data was imported correctly.
15. Verify that there is no data loss.
16. Repeat test at the system, crew, and task network levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

Load from Remote Database

Testing Procedure

1. Select Database | Remote → Connect to Database.
2. Enter username and password to connect to the IPME database.
3. Select Database | Remote → Load from Remote Database.
4. Locate the project name and select it.
5. Click “Load”.
6. Verify that there is no data loss in the project.
7. Repeat test at the system, crew, and task network levels.

Model Used

DRDC_Test/xml_import_export_sys.dat

Test Platform

RedHat 7.2

Pass/Fail

Pass

Related PR #s

PR414

6 Features Not To Be Tested

IPME

Environment Model

Performance Shaping Model

Workspace Diagram

POP Scheduler

POPIP Scheduler

7 Approach

Test Objectives

The objectives of the testing process were to identify and report problems and resolve them appropriately. A problem will be considered resolved when it has been fixed and retested and the problem no longer exists or the project manager deems it inappropriate to fix the problem. In the event that an identified problem was the result of known problems existing in third-party applications, the problem is considered resolved.

Types of Testing

The results of the initial development testing performed by the development team are not detailed here. The testing results provided in this document are the results of the system testing phase performed by the Quality Assurance project staff.

8 Artifacts

Test Reports

Issue Summary Report

Total number of defects found during testing:	158
Number of remaining open defects:	21
Percentage of issues resolved to date:	*87%*

Test Summary Report

Number of tests performed:	1055
Number of failed tests:	51
Percentage of tests passed:	*95%*

9 Elements Of Test

Staffing

The following people on the team supported testing this project:

Project Manager	Anna Fowles-Winkler
Quality Assurance Manager	Christy Lorenzen
Quality Assurance Staff	Shannon LaBay

Roles & Responsibilities

Quality Assurance tests the software and reviews documentation to ensure compliance with the contract.

Schedule

Testing was performed when the development team reported completion of a contract item. This occurred periodically throughout the contract Period of Performance.

Resources

MA&D computer hardware resources were used for testing purposes. Microsoft Word was utilized to compose this report.

DOCUMENT CONTROL DATA

(Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document, Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Publishing:	DRDC Toronto
Performing:	MicroAnalysis and Design, Inc., 4949 Pearl East Circle, Suite 300, Boulder, CO 80301
Monitoring:	
Contracting:	DRDC Toronto

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification is indicated by the appropriate abbreviation (S, C, R, or U) in parenthesis at the end of the title)

IPME Stabilization and HAWK Development (U)
Stabilisation du IPME et développement de HAWK : rapport final

4. AUTHORS (First name, middle initial and last name. If military, show rank, e.g. Maj. John E. Doe.)

Anna Fowles−Winkler

5. DATE OF PUBLICATION (Month and year of publication of document.)

February 2005

6a NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)

125

6b. NO. OF REFS (Total cited in document.)

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Contract Report

8. SPONSORING ACTIVITY (The names of the department project office or laboratory sponsoring the research and development − include address.)

Sponsoring:

Tasking:

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant under which the document was written. Please specify whether project or grant.)

13ia11

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document)

DRDC Toronto CR 2004−126

10b. OTHER DOCUMENT NO(s). (Any other numbers under which may be assigned this document either by the originator or by the sponsor.)

W7711−027844

11. DOCUMENT AVAILABILIY (Any limitations on the dissemination of the document, other than those imposed by security classification.)

Unlimited distribution

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11), However, when further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited announcement

DOCUMENT CONTROL DATA
(Security classification of the title, body of abstract and indexing annotation must be entered when the overall document is classified)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

(U) This report describes the work completed by Micro Analysis and Design for DRDC−Toronto under contract W7711−027844/A, “IPME Stabilization and HAWK Development.” The contract task items were performed during the period April 2003 – April 2004. Modifications were made to the Integrated Performance Modelling Environment (IPME) and the Human Analysis Work Kit (HAWK) per customer specification. The purpose of the work was two fold: to improve IPME stability by correcting software bugs and re−engineering problem areas of the code to conform to modern software design standards; to add capability to HAWK providing the user with tools to aid data collection, management, and model development.

(U) On décrit dans ce rapport le travail effectué par Micro Analysis and Design pour le RDDC − Toronto au titre du contrat W7711−027844/A intitulé « Stabilisation du IPME et développement de HAWK ». Les modalités de ce contrat ont été réalisées entre avril 2003 et avril 2004. Les modifications apportées au IPME (environnement intégré de la modélisation de la performance) et à HAWK (trousse d'analyse de la performance humaine) sont celles exprimées par le client.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

(U) Integrated Performance Modelleing Environment; human modelling; performance modelling