If you would like your presentation included in the 75th MORSS Final Report CD it must:
1. Be unclassified, approved for public release, distribution unlimited, and is exempt from U.S. export licensing and other export approvals including the International Traffic in Arms Regulations (22CFR120 et seq.);
2. Include MORSS Form 712CD as the first page of the presentation;
3. Have an approved MORSS form 712 A/B and
4. Be turned into the MORSS office no later than: DEADLINE: 14 June 2007 (Late submissions will not be included.)

Name of Principal Author and all other author(s): Todd Lindquist, Chad Quill

Phone: 410-564-3206
Fax: 410-712-4954
Email: Todd.R.Lindquist@saic.com

Please use the same title listed on the 75th MORSS Disclosure Form 712 A/B. If the title of the presentation has changed please list both.


If the title was revised please list the original title above and the revised title here:

PRESENTED IN:

| WORKING GROUP: | 8, 26 |
| COMPOSITE GROUP: | |
| SPECIAL SESSION 1: | |
| SPECIAL SESSION 2: | |
| SPECIAL SESSION 3: | |

This presentation is believed to be: Unclassified, approved for public release, distribution unlimited, and is exempt from U.S. export licensing and other export approvals including the International Traffic in Arms Regulations (22CFR120 et seq.)
There is no content in the image.
The Resource Allocation Strategy Evaluator (RASE)

**Mr. Todd Lindquist**
Team Lead/OR Analyst, SAIC
Operations Research and Systems Analysis Division

**Mr. Chad Quill**
Team Lead/OR Analyst, SAIC
Operations Research and Systems Analysis Division
RASE – What is it?

• A decision support tool developed in 2006 to:
  – Highlight the implications of different strategies available for resourcing requirements
  – Automate resource allocation
RASE – Methodology

Requirements
- Resource Needs
- Utility
- Requesting Org.

Resource Constraints
- Funding (O&M, Acq., RDT&E)
- Personnel (PM, CS, EE)
- Other

RASE
- Selection of Algorithm
- Execution of Algorithm

Evaluate Results and Perform Sensitivity Analysis

Select Best Fitting Resource Allocation
RASE – Algorithms

• Heuristics
  – Minimum Loading
    • Binding Resource
    • Proportional Impact
  – Maximum Loading
    • Binding Resource
    • Proportional Impact
  – Balanced Loading
RASE – Algorithms (cont.)

• Heuristics (cont.)
  – Utility Loading
  – Equity Based Loading

• Optimal Search
  – Maximize Number Resourced
  – Maximize Aggregate Utility
RASE – Features

- Manual override allocation
- Run and compare performance of all algorithms
- Select the number of constraints to consider
- Choice of optimization engine (LINGO, Excel Solver)
- Basic sorting
RASE – Future Additions

• Optimal Equity Algorithm (Nonlinear)
• Consider Ancillary Constraints
  – “If requirement X is resourced, requirement Y must be resourced”
• Formal sensitivity analysis capability
• Display multiple optimal solutions when present
• Allow user to stop optimal search at “close enough” solution
• Highlight binding constraint
• Advanced Sorting
RASE – Screenshots
## RASE Screenshots

### Hueristics

<table>
<thead>
<tr>
<th>Measures of Performance</th>
<th>Min Load</th>
<th>Min Load - Binding Constraint</th>
<th>Max Load</th>
<th>Max Load - Binding Constraint</th>
<th>Max Load - Proportional Impact</th>
<th>Balanced Loading</th>
<th>Utility Based Loading</th>
<th>Equitable Hueristics</th>
<th>Equitable - Balanced Loading</th>
<th>Equitable - Loading</th>
<th>Equitable - Utility Loading</th>
<th>Optimal Searches</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leftover D&amp;M Funds</td>
<td>43.7</td>
<td>60.6</td>
<td>75.2</td>
<td>(3.2)</td>
<td>76.7</td>
<td>62.3</td>
<td>14.0</td>
<td>14.0</td>
<td>33.1</td>
<td>41.7</td>
<td>61.0</td>
<td></td>
</tr>
<tr>
<td>Leftover Acq. Funds</td>
<td>3.7</td>
<td>22.0</td>
<td>13.2</td>
<td>7.7</td>
<td>0.4</td>
<td>26.0</td>
<td>15.9</td>
<td>18.9</td>
<td>2.4</td>
<td>22.2</td>
<td>5.6</td>
<td></td>
</tr>
<tr>
<td>Leftover RD&amp;E Funds</td>
<td>6.8</td>
<td>17.7</td>
<td>69.2</td>
<td>27.2</td>
<td>14.6</td>
<td>4.7</td>
<td>13.2</td>
<td>15.2</td>
<td>21.9</td>
<td>30.9</td>
<td>9.1</td>
<td></td>
</tr>
<tr>
<td>Leftover Comp. Sci</td>
<td>28.0</td>
<td>46.0</td>
<td>47.2</td>
<td>26.0</td>
<td>36.0</td>
<td>32.0</td>
<td>35.0</td>
<td>35.0</td>
<td>19.0</td>
<td>46.0</td>
<td>46.0</td>
<td></td>
</tr>
<tr>
<td>Leftover Elec. Eng</td>
<td>20.0</td>
<td>26.0</td>
<td>52.0</td>
<td>17.0</td>
<td>32.0</td>
<td>48.0</td>
<td>37.0</td>
<td>37.0</td>
<td>46.0</td>
<td>41.0</td>
<td>26.0</td>
<td></td>
</tr>
<tr>
<td>Leftover Comm. Eng</td>
<td>54.0</td>
<td>26.0</td>
<td>30.0</td>
<td>40.0</td>
<td>24.0</td>
<td>21.0</td>
<td>32.0</td>
<td>32.0</td>
<td>51.0</td>
<td>19.0</td>
<td>23.0</td>
<td></td>
</tr>
</tbody>
</table>

### Number of Req. Recessed

<table>
<thead>
<tr>
<th></th>
<th>11</th>
<th>13</th>
<th>6</th>
<th>6</th>
<th>9</th>
<th>7</th>
<th>11</th>
<th>11</th>
<th>7</th>
<th>13</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Aggregate Utility Score</td>
<td>389.6</td>
<td>385.0</td>
<td>341.0</td>
<td>337.0</td>
<td>393.0</td>
<td>421.0</td>
<td>414.0</td>
<td>414.0</td>
<td>394.0</td>
<td>362.0</td>
<td>501.0</td>
</tr>
<tr>
<td>Std. Deviation of Org. Equity</td>
<td>140.0</td>
<td>93.6</td>
<td>167.0</td>
<td>168.6</td>
<td>168.6</td>
<td>163.3</td>
<td>155.3</td>
<td>155.3</td>
<td>178.9</td>
<td>110.6</td>
<td>89.0</td>
</tr>
</tbody>
</table>

### Manually Assigned Requirements

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R11</td>
<td>R20</td>
<td>R30</td>
<td>R33</td>
<td>R11</td>
<td>R11</td>
<td>R11</td>
<td>R11</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
</tr>
<tr>
<td>R15</td>
<td>R23</td>
<td>R26</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
</tr>
<tr>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
</tr>
<tr>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
</tr>
</tbody>
</table>

### RASE Assigned Requirements

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R11</td>
<td>R20</td>
<td>R30</td>
<td>R33</td>
<td>R11</td>
<td>R11</td>
<td>R11</td>
<td>R11</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
</tr>
<tr>
<td>R15</td>
<td>R23</td>
<td>R26</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
<td>R23</td>
</tr>
<tr>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
<td>R20</td>
</tr>
<tr>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
<td>R36</td>
</tr>
</tbody>
</table>
1. MINIMUM LOADING STRATEGY

A. Binding Resource heuristic - With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced” by setting $X_i = 0$

b) The resources required for each requirement “i” for each resource type “k” is summed across all requirements to estimate $S_k = \sum r_{ik}$

c) The degree of loading (e.g. binding) for each resource type will be the fraction of available resources that is needed (normalized by available resources), computed as: $\Delta k = S_k / R_k$ for each $k$

d) The probable (but not assured) binding resource will be the index defined by $RB = \text{MAX}[\Delta k]$

e) Sort requirements from smallest to largest resources needed for resource type $RB$.

f) Set $RL_k = R_k$

g) Go to the next CNO requirement on the sorted list (if this is the first time, go to the first requirement on the sorted list).

h) For this requirement, compute $RL_k = RL_k - r_{ik}$ for all “k”

i) If $RL_k >= 0$ for all “k”, then mark this requirement as resourced by setting $X_i = 1$, then go to step 1.g, otherwise continue to step 1.j

j) For the resource type “k” that went below zero in step 1.i, all available resources have been allocated and the resource allocation process is complete.
1. **MINIMUM LOADING STRATEGY**

**B. Proportional Impact heuristic** - With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced” by setting $X_i = 0$

b) Set $RL_k = R_k$ for all “k”

c) The fraction of the totally available resources that are needed by each requirement is computed as: $P_i = \sum r_{ik} / RL_k$ (summed across index “k”)

d) Sort requirements from smallest to largest values of $P_i$

e) Select the top CNO requirement on the sorted list that was not previously selected.

f) For this requirement, compute $RL_k = RL_k - r_{ik}$ for all “k”

g) If $RL_k \geq 0$ for all “k”, then mark this requirement as resourced by setting $X_i = 1$, then go to step c, otherwise continue to step h

h) For the resource type “k” that went below zero in step 1.g, all available resources have been allocated and the resource allocation process is complete.
2. MAXIMUM LOADING STRATEGY

A. Binding Resource heuristic - With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced” by setting $X_i = 0$

b) The resources required for each requirement “i” for each resource type “k” is summed across all requirements to estimate $S_k = \sum r_{ik}$

c) The degree of loading (e.g. binding) for each resource type will be the fraction of available resources that is needed (normalized by available resources), computed as: $D_k = \frac{S_k}{R_k}$ for each k

d) The probable (but not assured) binding resource will be the index defined by $RB=\text{MAX}[\Delta k]$

e) Sort requirements from largest to smallest resources needed for resource type RB.

f) Set $RL_k = R_k$

g) Go to the next CNO requirement on the sorted list (if this is the first time, go to the first requirement on the sorted list).

h) For this requirement, compute $RL_k = RL_k - r_{ik}$ for all “k”

i) If $RL_k \geq 0$ for all “k”, then mark this requirement as resourced by setting $X_i = 1$, then go to step 2.g, otherwise continue to step 2.j

j) For the resource type “k” that went below zero in step 2.i, all available resources have been allocated to large requirements. If there are still requirements on the list, it is possible that one may require resources less then what is currently left over (since we sorted from highest to lowest). For this reason, we allow the algorithm to continue checking the sorted list all the way to the end. If this is the end of the list, then stop. If not, then for all “k” set $RL_k = RL_k + r_{ik}$ (e.g return the resources that we removed from the last requirement that would not fit) and go to step g.
2. **MAXIMUM LOADING STRATEGY**

**B. Proportional Impact heuristic** - With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced” by setting $X_i = 0$

b) Set $R_{Lk} = R_k$ for all “k”

c) The fraction of the totally available resources that are needed by each requirement is computed as: $P_i = \sum r_{ik} / R_{Lk}$ (summed across index “k”)

d) Sort requirements from largest to smallest values of $P_i$

e) Select the top CNO requirement on the sorted list that was not previously selected.

f) For this requirement, compute $R_{Lk} = R_{Lk} - r_{ik}$ for all “k”

g) If $R_{Lk} \geq 0$ for all “k”, then mark this requirement as resourced by setting $X_i = 1$, then go to step c, otherwise continue to step h

h) For the resource type “k” that went below zero in step 2.g, all available resources have been allocated to large requirements. If there are still requirements on the list, it is possible that one may require resources less than what is currently left over (since we sorted from highest to lowest). For this reason, we allow the algorithm to continue checking the sorted list all the way to the end. If this is the end of the list, then stop. If not, then for all “k” set $R_{Lk} = R_{Lk} + r_{ik}$ (e.g return the resources that we removed from the last requirement that would not fit) and go to step e.
3. BALANCED LOADING STRATEGY

With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced” by setting $X_i = 0$

b) The resources required for each requirement “i” for each resource type “k” is summed across all requirements to estimate $S_k = \Sigma r_{ik}$

c) The degree of loading (e.g. binding) for each resource type will be the fraction of available resources that is needed (normalized by available resources), computed as: $\Delta k = \frac{S_k}{R_k}$ for each k

d) The probable (but not assured) binding resource will be the index defined by $RB=\text{MAX}[\Delta k]$

e) Using resource type RB as your criteria, sort the CNO requirements from smallest to largest. Set the small counter $S=1$ and the large counter $N=\text{number of requirements}$.

f) Set $RL_k = R_k$

g) Go to the CNO requirement on the sorted list corresponding to S

h) For this requirement, compute $RL_k = RL_k - r_{Sk}$ for all “k”

i) If $RL_k \geq 0$ for all “k”, then mark this requirement as resourced by setting $XS = 1$, then go to step 2.j, otherwise STOP.

j) If $S\geq N$, STOP (all requirements have been resourced). Otherwise, go to the CNO requirement on the list corresponding to N

k) For this requirement, compute $RL_k = RL_k - r_{Nk}$ for all “k”

l) If $RL_k \geq 0$ for all “k”, then mark this requirement as resourced by setting $XN = 1$, also set $S=S+1$ and $N=N-1$, then go to step 2.g. Otherwise continue

m) If $S > N$, then STOP. Otherwise continue.

n) For the resource type “k” that went below zero in step 2.l, all available resources have been allocated to large requirements. If there are still requirements on the list, it is possible that one may require resources less than what is currently left over (since we sorted from highest to lowest). For this reason, we allow the algorithm to continue checking the sorted list all the way to the end. If this is the end of the list, then stop. If not, then for all “k” set $RL_k = RL_k + r_{ik}$ (e.g return the resources that we removed from the last requirement that would not fit) and go to step g.
4. EQUITY BASED STRATEGY

With this approach, the following steps are used to assign resources to CNO requirements:

a) Separate all requirements into J lists (represented by $\Lambda$) where each list represents requirements associated with each organization.

b) Mark all CNO requirements as "not resourced" by setting $X_{ij} = 0$.

c) The resources required for each requirement "ij" for each resource type "k" is summed across all requirements to estimate $S_k = \sum r_{ik}$.

d) The degree of loading (e.g., binding) for each resource type will be the fraction of available resources needed (normalized by available resources), computed as: $\Delta_k = S_k / R_k$ for each k.

e) The probable binding resource will be the index defined by $RB = \text{MAX}[\Delta_k]$.

f) Set the list counter $j = 1$ and $RL_k = R_k$.

g) If sub objective = "utility based", sort CNO requirements in each $\Lambda_j$ list from highest to lowest utility score.

h) If sub objective is not = "utility based", then sort each $\Lambda_j$ list of CNO requirements from smallest to largest resources needed for resource type RB.

i) If sub objective is not = "balanced", then go to step k.

i. Set low counter $L_j = 1$ and the high counter $H_j = \text{number of requirements for organization j}$ for all J.

ii. Go to the CNO requirement on the sorted list $L_j$ corresponding to $L_j$.

iii. If $XL_{jj} = 1$ then go to step j.5. Otherwise, for this requirement, compute $RL_k = RL_k - r_{Ljk}$ for all "k".

iv. If $RL_k >= 0$ for all "k", then mark this requirement as resourced by setting $XL_{jj} = 1$, and go to step j.5. Otherwise STOP.

v. If $L_j >= H_j$, go to step j.9 (all requirements for this organization are resourced). Otherwise, go to next requirement on the list corresponding to $H_j$.

vi. If $XL_{jj} = 1$ then go to step j.8. Otherwise, for this requirement, compute $RL_k = RL_k - r_{Hjk}$ for all "k".

vii. If $RL_k >= 0$ for all "k", then mark this requirement as resourced by setting $XL_{jj} = 1$, then go to step j.8. Otherwise: set $RL_k = RL_k + r_{Hjk}$ for all k, set $L_j = L_j + 1$, $H_j = H_j - 1$. If $L_j > H_j$, then go to step j.9. Otherwise go to step j.2.

viii. Set $L_j = L_j + 1$, and $H_j = H_j - 1$. If $L_j <= H_j$, Go to step j.2. Otherwise continue.

ix. Set $j = j + 1$. If $j > J$ set $j = 1$. Go to step j.2.

j) The sub objective is = "utility based", so perform the following steps:

i. Set counter $L_j = 1$ for organization j for all J.

ii. Go to the CNO requirement on the sorted list $L_j$ corresponding to $L_j$.

iii. For this requirement, compute $RL_k = RL_k - r_{Ljk}$ for all "k".

iv. If $RL_k >= 0$ for all "k", then mark this requirement as resourced by setting $XL_{jj} = 1$, and go to step k.9. Otherwise continue.

v. Set $RL_k = RL_k + r_{Hjk}$ for all k.

vi. Set $L_j = L_j + 1$. If $L_j < \text{Number of requirements for this organization}$, then go to step k.2. Otherwise continue.

vii. Set $j = j + 1$. If $j > J$ set $j = 1$. If all $X_{ij} = 1$ then STOP, otherwise go to step k.2.
5. **UTILITY BASED STRATEGY**

With this approach, the following steps are used to assign resources to CNO requirements:

a) Mark all CNO requirements as “not resourced by setting $X_i = 0$

b) Sort CNO requirements from highest utility score to lowest score, and set $S=1$.

c) Set $RL_k = R_k$

d) Go to the CNO requirement on the sorted list corresponding to $S$

e) For this requirement, compute $RL_k = RL_k - r_{Sk}$ for all “$k$”

f) If $RL_k \geq 0$ for all “$k$”, then mark this requirement as resourced by setting $X_S = 1$, then go to step 2.g, otherwise continue to step 2.h

g) Set $S=S+1$, and go to step d

h) For the resource type “$k$” that went below zero in step 2.f, all available resources have been allocated and the resource allocation process is complete.
6. **OPTIMAL SEARCH STRATEGY**

With this approach, the following steps are used to assign resources to CNO requirements:

a) An objective function is created to guide the search process. The objective function takes the following form based on which sub objective was selected:

1. If sub objective = “Maximum requirements”, then the Objective function is defined as:  \[ \text{Max } \sum X_i \]
2. If sub objective = “Utility based”, then the Objective function is defined as:  \[ \text{Max } \sum u_i X_i \]

b) A constraint is formulated for each resource type, using the general form:  \[ \sum \sum r_{ijk} X_{ij} < R_k \] for each k (summation is across i and j)
RASE – Algorithm Terminology

i = An index used to designate a specific CNO requirement

j = An index used to designate a specific organization

k = An index used to indicate a type of resource (funds, people, etc) needed to support requirement i

k = The index of the resource selected by the user to achieve an equitable distribution of resources across organizations (e.g. “funds”)

Pi = The proportion of the available resources (across all “k” types of resources) that is needed by CNO requirement “i” for organization “j”

rijk = The resources of type k needed to execute requirement “i” for organization “j”

Rk = The amount of resource type k that is available and can be allocated across the CNO requirements.

RLk = The amount of available resources left over as each requirement is allocated its necessary resources

Xij = A decision variable indicating whether requirement ij has been chosen to be resourced

uij = The benefit (utility) of assigning resources to requirement “ij”

Lj = List of requirements associated with organization j