Using Dynamic Multi-Attribute Tradespace Exploration to Develop Value Robust Systems

INCOSE North Texas Chapter meeting

Dr. Adam M. Ross, MIT
adamross@mit.edu
October 18, 2007
Meeting Customer Needs

• Goal of design is to create value (profits, usefulness, voice of the customer, etc…)
• Requirements capture a mapping of needs to specifications to guide design
Deploying a “Valuable” System...
Deploying a “Valuable” System…
Deploying a “Valuable” System…
Deploying a “Valuable” System...
Deploying a “Valuable” System…

Contexts change…
Meeting Customer Needs (cont.)

• Goal of design is to create value (profits, usefulness, voice of the customer, etc…)
• Requirements capture a mapping of needs to specifications to guide design
Meeting Customer Needs (cont.)

- Goal of design is to create value (profits, usefulness, voice of the customer, etc…)
- Requirements capture a mapping of needs to specifications to guide design
Meeting Customer Needs (cont.)

- Goal of design is to create value (profits, usefulness, voice of the customer, etc…)

- People change their minds…
- To continue to deliver value, systems may need to change as well…
According to Dr. Marvin Sambur, “Systems Engineering for Robustness” means developing systems that are…

- Capable of *adapting* to changes in mission and requirements
- *Expandable/scalable*, and designed to accommodate growth in capability
- Able to *reliably* function given changes in threats and environment
- Effectively/affordably *sustainable* over their lifecycle
- Developed using products designed for use in various *platforms* and systems
- Easily *modified* to leverage new technologies

- “Robustness” scope expanded beyond classical robustness …

- Experts questioned…
  - What does it mean?
  - How can it be measured/analyzed?
  - Who is going to pay for it?

What is System Success?

Success is defined across **multiple perspectives** and **multiple time periods**

**System success, \( \Psi \), across \( N \) decision makers at time \( t \)**

\[
\Psi(t) = \sum_{i=1}^{N} \left[ X_{DMI}(t) + \varepsilon_C X_{DMI}(t) \right] \geq Y_{DMI}(t) + \varepsilon_C Y_{DMI}(t)
\]

\[0 \leq \Psi(t) \leq N\]

**\( X_{DMI}(t) \)**: Decision maker i unaffected system “experience” at time t

**\( Y_{DMI}(t) \)**: Decision maker i unaffected system “expectation” at time t

**\( \varepsilon_C X_{DMI}(t) \)**: Context effect on decision maker i “experience” at time t

**\( \varepsilon_C Y_{DMI}(t) \)**: Context effect on decision maker i “expectation” at time t

System Success: Net “experience” must meet or exceed net “expectations”
Characterizing the System Design Opportunity

Exogenous Influences

Decision Maker

Resources

Expectations

Needs

Constraints

Success?

Designer

Exogenous Influences

Expectations

Value

Value

Designs

$
Characterizing the System Design Opportunity

Exogenous Influences

Decision Maker

Resources

Expectations

Needs

Constraints

Exogenous Influences

Designs

Success?

Value

Experience

Value

Experience

Value

Experience

Value

Experience

Value

Expectations

Success?
Characterizing the System Design Opportunity

- **Expectations**
- **Success?**
- **Exogenous Influences**
- **Resources**
- **Needs**
- **Designs**
- **Constraints**
- **System**
- **Context**
- **Experience**

- **Decision Maker**
  - Value
- **Designer**
  - Value

- **Success?**
  - $☺☺ ☺☺ ☺☺ ☺☺$
Characterizing the System Design Opportunity

Exogenous Influences

Decision Maker

Value

Resources

Needs

Expectations

Constraints

Designs

System

Context

Experience

Exogenous Influences

Designer

Value

Expectations

Experience

Success?
Characterizing the System Design Opportunity

Exogenous Influences

Decision Maker

Resources

Expectations

Needs

Constraints

System

Context

Designer

Value

Experience

Exogenous Influences

Time

Expectations

Experience

Value

…
Types of Changes

- Δ Designs (including technology)
- Δ Context (including operating environment, competition)
- Δ Constraints (including “laws”)
- Δ Needs (including attributes)
- Δ DMs (including individuals and groups)
- Δ Resources (including dollars and time)

How can System Designers cope with these types of changes during design?
Aspects of Dynamic MATE*

How can System Designers cope with these types of changes during design?

• System Success criteria
  – Expanding scope of system “value”

• Tradespace exploration
  – Understanding success possibilities across a large number of designs

• Change taxonomy
  – Specifying and identifying change types

• Tradespace networks
  – Analyzing changeability of designs

• System Epoch/Era analysis
  – Quantifying effects of changing contexts on system success

*MATE = Multi-Attribute Tradespace Exploration
Tradespace Exploration

Value-driven design...

- Firm
- Designer
- Customer
- User

- Value
- Attributes
- Utility

- Analysis

Concept

Design Variables

"Cost"

Tradespace: \{Design, Attributes\} $\leftrightarrow$ \{Cost, Utility\}

Each point is a specific design

Total Lifecycle Cost ($M2002$)
Tradespace Exploration

Value-driven design...

- Firm
- Designer
- Customer
- User

- Value
- Attributes
- Utility

- Concept
- Design
- Variables
- "Cost"

Tradespace: \{Design,Attributes\} \leftrightarrow \{Cost,Utility\}

X-TOS
Small low-altitude science mission

Example

DESIGN VARIABLES:
Design trade parameters
- Orbital Parameters
  - Apogee Altitude (km)
  - Perigee Altitude (km)
  - Orbit Inclination (deg)
- Spacecraft Parameters
  - Antenna Gain
  - Communication Architecture
  - Propulsion Type
  - Power Type
  - Total Delta V

ATTRIBUTES:
Design decision metrics
- Data Lifespan (yrs)
- Equatorial Time (hrs/day)
- Latency (hrs)
- Latitude Diversity (deg)
- Sample Altitude (km)

Assessment of cost and utility of large space of possible system designs
Differing types of trades

1. Local point solution trades

Design$_i$ = {DV$_1$, DV$_2$, DV$_3$, …, DV$_j$}
Differing types of trades

1. Local point solution trades
2. Frontier subset solutions

Design$_i$ = {DV$_1$, DV$_2$, DV$_3$, ..., DV$_j$}
Tradespace Exploration: Avoiding Point Designs

Differing types of trades

1. Local point solution trades
2. Frontier subset solutions
3. Frontier solution set

Design_i = \{DV_1, DV_2, DV_3, \ldots, DV_j\}
Differing types of trades

1. Local point solution trades
2. Frontier subset solutions
3. Frontier solution set
4. Full tradespace exploration

Design$_i$ = \{DV$_1$, DV$_2$, DV$_3$, ..., DV$_j$\}
Tradespace Exploration: Avoiding Point Designs

Differing types of trades

1. Local point solution trades
2. Frontier subset solutions
3. Frontier solution set
4. Full tradespace exploration
5. Dynamic tradespace relations

Design$_i$ = \{DV$_1$, DV$_2$, DV$_3$, \ldots, DV$_j$\}

Tradespace exploration enables big picture understanding
**Example “Real Systems”**

### Spacetug vs CX-OLEV

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Wet Mass kg</td>
<td>1405</td>
<td>1400</td>
</tr>
<tr>
<td>Dry Mass kg</td>
<td>805</td>
<td>670*</td>
</tr>
<tr>
<td>Propellant kg</td>
<td>600</td>
<td>730*</td>
</tr>
<tr>
<td>Equipment kg</td>
<td>300</td>
<td>213*</td>
</tr>
<tr>
<td>DV m/s</td>
<td>12000 – 16500***</td>
<td>15900**</td>
</tr>
<tr>
<td>Utility</td>
<td>0.69</td>
<td>0.69</td>
</tr>
<tr>
<td>Cost</td>
<td>148</td>
<td>130*</td>
</tr>
</tbody>
</table>

### XTOS vs Streak

<table>
<thead>
<tr>
<th></th>
<th>XTOS (2002 study)</th>
<th>Streak (Oct 2005 launch)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wet Mass kg</td>
<td>325 - 450</td>
<td>420</td>
</tr>
<tr>
<td>Lifetime (yrs)</td>
<td>2.3 - 0.5</td>
<td>1</td>
</tr>
<tr>
<td>Orbit</td>
<td>300 - 185 km @ 20°</td>
<td>321a-296p -&gt; 200 @ 96°</td>
</tr>
<tr>
<td>LV</td>
<td>Minotaur</td>
<td>Minotaur</td>
</tr>
<tr>
<td>Utility</td>
<td>0.61 - 0.55</td>
<td>0.57 - 0.54*</td>
</tr>
<tr>
<td>Modified Utility**</td>
<td>0.56 - 0.50</td>
<td>0.59</td>
</tr>
<tr>
<td>Cost SM</td>
<td>75 - 72</td>
<td>75***</td>
</tr>
<tr>
<td>Instruments</td>
<td>Three (?)</td>
<td>Ion gauge and atomic oxygen sensor</td>
</tr>
</tbody>
</table>
If the “best” design changes over time, how does one select the “best” design?
Tradespace Networks

Transition rules are mechanisms to change one design into another.
The more outgoing arcs, the more potential change mechanisms.

Tradespace designs = nodes
Applied transition rules = arcs
Tradespace Networks

Example: X-TOS Transition Rules

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
<th>Change agent origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1: Plane Change</td>
<td>Increase/decrease inclination, decrease ΔV</td>
<td>Internal (Adaptable)</td>
</tr>
<tr>
<td>R2: Apogee Burn</td>
<td>Increase/decrease apogee, decrease ΔV</td>
<td>Internal (Adaptable)</td>
</tr>
<tr>
<td>R3: Perigee Burn</td>
<td>Increase/decrease perigee, decrease ΔV</td>
<td>Internal (Adaptable)</td>
</tr>
<tr>
<td>R4: Plane Tug</td>
<td>Increase/decrease inclination, requires “tugable”</td>
<td>External (Flexible)</td>
</tr>
<tr>
<td>R5: Apogee Tug</td>
<td>Increase/decrease apogee, requires “tugable”</td>
<td>External (Flexible)</td>
</tr>
<tr>
<td>R6: Perigee Tug</td>
<td>Increase/decrease perigee, requires “tugable”</td>
<td>External (Flexible)</td>
</tr>
<tr>
<td>R7: Space Refuel</td>
<td>Increase ΔV, requires “refuelable”</td>
<td>External (Flexible)</td>
</tr>
<tr>
<td>R8: Add Sat</td>
<td>Change all orbit, ΔV</td>
<td>External (Flexible)</td>
</tr>
</tbody>
</table>

Tradespace designs  = nodes
Applied transition rules  = arcs

Transition rules are mechanisms to change one design into another. The more outgoing arcs, the more potential change mechanisms.
Select changeable designs that can approximate "best" designs in new contexts
Determining Changeability

The Question: Is the system _____________?
(Flexible, Adaptable, Robust, Scalable, Modifiable, Changeable, Rigid, etc…)

The Answer: It depends!

The question of changeability is partly subjective: Is the “cost” for change acceptable?
Changeability Metric: Filtered Outdegree

**Objective**

Outdegree

# outgoing arcs from a given node

Subjective Filtered Outdegree

# outgoing arcs from design at acceptable “cost”

(measure of changeability)

Filtered outdegree is a measure of the apparent changeability of a design
Pareto Set designs (903, 1687, 2471) are not the most changeable
Design 7156 becomes relatively more changeable as cost threshold increases

Outdegree functions reveal differential nature of apparent changeability
Tradespace Networks in the System Era

Pareto Tracing across Epochs

Changeability Quantified as Filtered Outdegree

Temporal strategy can be developed across networked tradespace
Example System Timeline

Example system: Serviceable satellite

System timeline with “serviceability”-enabled paths allow value delivery
Ex: Desiring “No” Change:
Value Robustness

Preferences t=1

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Choose 3

$U(3) > U(2) > U(1) > U(4)$

Sizes: big > small
Loudness: loud > quiet

Preferences t=2

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Attribute “priority”
Ex: Desiring “No” Change: Value Robustness

Preferences t=1

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
</tbody>
</table>

Choose 3

Preferences t=2

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

Choose 4

New

Attribute “priority”

U(3) > U(2) > U(1) > U(4)

U(4) > U(2) > U(3) > U(1)

big > small
loud > quiet
red > gray > black
Ex: Desiring “No” Change: Value Robustness

Preferences t=1
Attribute $k_i$
Size 0.5
Loudness 0.2
Choose 3

Preferences t=2
Attribute $k_i$
Size 0.5
Loudness 0.2
Color 0.6
Choose 4

Attribute
big>small
loud>quiet
red>gray>black

Choose 3
Change?

Attribute “priority”
New
Ex: Desiring “No” Change: Value Robustness

Preferences t=1

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
</tbody>
</table>

Choose 3

Preferences t=2

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$k_i$</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

Change?

If switching costs are high, option 2 may be better choice (i.e. robust in value)
Ex: Desiring “No” Change: Value Robustness

Preferences t=1

<table>
<thead>
<tr>
<th>Attribute</th>
<th>k_i</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
</tbody>
</table>

Choose 3

Preferences t=2

<table>
<thead>
<tr>
<th>Attribute</th>
<th>k_i</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Loudness</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

New

Choose 4

If switching costs are high, option 2 may be better choice (i.e. robust in value)

Clever designs or changeable designs can achieve value robustness
Achieving Value Robustness

Research suggests two strategies for “Value Robustness”

New Context Drivers
- External Constraints
- Design Technologies
- Value Expectations

1. Passive
   - Choose “clever” designs that remain high value
   - Quantifiable: Pareto Trace number

2. Active
   - Choose changeable designs that can deliver high value when needed
   - Quantifiable: Filtered Outdegree

Value robust designs can deliver value in spite of inevitable context change
Dynamic MATE Summary

A Layered Approach

Perform Static MATE
- Attributes
- Designs
- Proposed Rules

Define Epochs
- Potential Contexts
- $\Delta DV, X, R$

Construct Eras
- Epoch Series
- Dynamic Strategies

Changeability provides insight for achieving dynamic \textit{value robustness}
Designing for Value Robustness

**Mindshift:** recognize dynamic contexts and fallacy of static preferences; the inevitability of “change”

- Matching changing systems to changing needs leads to sustained system success
- Methods for increasing Changeability
  - Increase number of paths (change mechanisms)
  - Lower “cost” or increase acceptability threshold (apparent changeability)
- Concept-independent measure of changeability: filtered outdegree
  - Change mechanism drives filtered outdegree, as does consideration of alternative systems (future states)
  - Quantifiable filtered outdegree couples both objective and subjective measures (irreducible subjectivity → acceptability threshold)
- Changeability can be used as an explicit and consistent metric for designing systems

**Designed for changeability, systems will be empowered to become value robust, delivering value in spite of context and preference changes**
Thank you for your attention!
Any questions?
adamross@mit.edu

For further details on topic please visit: 
http://seari.mit.edu