The Forgotten “-ilities”
Topics

• What is an “-Ility”?
• How might we organize “-Ilities”?
• How Should Systems Engineers View “-Ilities”?
• Summary
# Most Common Lists of -ilities

<table>
<thead>
<tr>
<th>RAM-T (Eng)</th>
<th>RASR (DBs)</th>
<th>RAMS (Safety)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>Reliability</td>
<td>Reliability</td>
</tr>
<tr>
<td>Availability</td>
<td>Availability</td>
<td>Availability</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Scalability</td>
<td>Maintainability</td>
</tr>
<tr>
<td>Testability</td>
<td>Recoverability</td>
<td>Safety</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RASUI (SW)</th>
<th>FURPS (SW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>Functionality</td>
</tr>
<tr>
<td>Availability</td>
<td>Usability</td>
</tr>
<tr>
<td>Serviceability</td>
<td>Reliability</td>
</tr>
<tr>
<td>Usability</td>
<td>Performance</td>
</tr>
<tr>
<td>Instability</td>
<td>Supportability</td>
</tr>
</tbody>
</table>
Are there more –ilities?

<table>
<thead>
<tr>
<th>Accessibility</th>
<th>Executability</th>
<th>Performability</th>
<th>Supportability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accountability</td>
<td>Extensibility</td>
<td>Portability</td>
<td>Suitability</td>
</tr>
<tr>
<td>Adaptability</td>
<td>Evolvability</td>
<td>Practibility</td>
<td>Survivability</td>
</tr>
<tr>
<td>Administrability</td>
<td>Fidelity</td>
<td>Practicality</td>
<td>Tailorability</td>
</tr>
<tr>
<td>Affordability</td>
<td>Flexibility</td>
<td>Predictability</td>
<td>Testability</td>
</tr>
<tr>
<td>Agility</td>
<td>Functionality</td>
<td>Producibility</td>
<td>Traceability</td>
</tr>
<tr>
<td>Availability</td>
<td>Integratibility</td>
<td>Recoverability</td>
<td>Trainability</td>
</tr>
<tr>
<td>Capability</td>
<td>Interoperability</td>
<td>Reliability</td>
<td>Transportability</td>
</tr>
<tr>
<td>Composability</td>
<td>Interpretability</td>
<td>Repeatability</td>
<td>Trustability</td>
</tr>
<tr>
<td>Configurability</td>
<td>Maintainability</td>
<td>Responsibility</td>
<td>Understandability</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Manageability</td>
<td>Reusability</td>
<td>Upgradability</td>
</tr>
<tr>
<td>Demonstrability</td>
<td>Mobility</td>
<td>Scalability</td>
<td>Usability</td>
</tr>
<tr>
<td>Deployability</td>
<td>Modifiability</td>
<td>Serviceability</td>
<td>Verifiability</td>
</tr>
<tr>
<td>Durability</td>
<td>Operability</td>
<td>Stability</td>
<td>Vulnerability</td>
</tr>
</tbody>
</table>
What is the Definition of “-ility”

The developmental, operational, and support requirements a program must address (e.g., availability, maintainability, vulnerability, reliability, supportability, etc.).
What is an “-ility”: Other Terms

“Feature”

“Characteristic”

“Attribute”

“Constraints”

“Quality Goals”

“Other properties”

Most Common: Non-functional requirement
## Functional vs Nonfunctional Requirements (SW)

<table>
<thead>
<tr>
<th>Functional</th>
<th>Nonfunctional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product features</td>
<td>Product properties</td>
</tr>
<tr>
<td>Describe the work that is done</td>
<td>Describe the character of the work</td>
</tr>
<tr>
<td>Describe the actions with which the work is concerned</td>
<td>Describe the experience of the user while doing the work</td>
</tr>
<tr>
<td>Characterized by verbs</td>
<td>Characterized by adjectives</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Functional</th>
<th>Nonfunctional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific Functions and behaviors</td>
<td>Criteria that can be used to judge the operation of a system</td>
</tr>
<tr>
<td>System Design</td>
<td>System Architecture</td>
</tr>
<tr>
<td>What a system is supposed to DO</td>
<td>What a system is supposed to BE</td>
</tr>
<tr>
<td></td>
<td>Characteristic of a system that applies across a set of functional or system requirements.</td>
</tr>
</tbody>
</table>

Software Architecture Notes: Making theilities come true [http://www.softwarearchitecturenotes.com/architectureRequirements.html](http://www.softwarearchitecturenotes.com/architectureRequirements.html)
Are there more –ilities?

<table>
<thead>
<tr>
<th>Accessibility</th>
<th>Executability</th>
<th>Performability</th>
<th>Supportability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accountability</td>
<td>Extensibility</td>
<td>Portability</td>
<td>Suitability</td>
</tr>
<tr>
<td>Adaptability</td>
<td>Evolvability</td>
<td>Practibility</td>
<td>Survivability</td>
</tr>
<tr>
<td>Administrability</td>
<td>Fidelity</td>
<td>Practicality</td>
<td>Tailorability</td>
</tr>
<tr>
<td>Affordability</td>
<td>Flexibility</td>
<td>Predictability</td>
<td>Testability</td>
</tr>
<tr>
<td>Agility</td>
<td>Functionality</td>
<td>Producibility</td>
<td>Traceability</td>
</tr>
<tr>
<td>Availability</td>
<td>Integratibility</td>
<td>Recoverability</td>
<td>Trainability</td>
</tr>
<tr>
<td>Capability</td>
<td>Interoperability</td>
<td>Reliability</td>
<td>Transportability</td>
</tr>
<tr>
<td>Composability</td>
<td>Interpretability</td>
<td>Repeatability</td>
<td>Trustability</td>
</tr>
<tr>
<td>Configurability</td>
<td>Maintainability</td>
<td>Responsibility</td>
<td>Understandability</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Manageability</td>
<td>Reusability</td>
<td>Upgradability</td>
</tr>
<tr>
<td>Demonstrability</td>
<td>Mobility</td>
<td>Scalability</td>
<td>Usability</td>
</tr>
<tr>
<td>Deployability</td>
<td>Modiﬁability</td>
<td>Serviceability</td>
<td>Veriﬁability</td>
</tr>
<tr>
<td>Durability</td>
<td>Operability</td>
<td>Stability</td>
<td>Vulnerability</td>
</tr>
</tbody>
</table>
How Can We Organize “-ilities”?
How can we organize this disparate List?

- Lifecycle phase
- Dependency and Priority
- Cost and value
- Criticality

Questions:
- Do –ilities describe the product
- Are they more associated with SE functions leading to design?
- Do they drive product design
- Are they key to ensuring the product parts can be integrated?
- How do they relate to SE processes?

Group these by
- Relationship
- Timeline on Lifecycle
- Dependencies
- Aggregation
- Priority
- Value
Similar Pairs

Interoperability - Compatibility

Flexibility - Adaptability

Availability - Reliability

© 2011 Systems and Proposal Engineering Company. All rights reserved.
Dynamic Relationship

Seeking to establish and maintain balance between two attributes in a dynamic environment
Hierarchical Relationships: Example 1

Agility

- Debggability
- Extensibility
- Scalability
- Securability
- Testability

Understandability

Hierarchical Relationships: Example 2

- General Utility
  - Portability
  - Usability
    - Reliability
      - Human Engineering
    - Efficiency
    - Testability
      - Understandability
        - Modifiability
        - Device-Independence
          - Self-Containedness
          - Accuracy
          - Completeness
          - Robustness
          - Consistency
          - Accountability
          - Device Efficiency
          - Accessibility
          - Communications
          - Self-descriptiveness
          - Structuredness
          - Conciseness
          - Legibility
          - Augmentability

this looks like fresh thinking…but ….
... it was initially put forward 35 years ago
How Should Systems Engineers View “-ilities”? 
What is a System?

...combination of interacting elements organized to achieve one or more stated purposes.

...an integrated set of elements, subsystems, or assemblies that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements.
What is a System?

People

Things

Processes
LML Taxonomy Provides System Descriptions

- Technical
  - *Action (Processes)*
  - Artifact
  - *Asset (People & Things)*
  - *Characteristic (“ilities”)*
  - Input/Output
  - Link
  - Statement

- Programmatic/Technical
  - Cost
  - Issue
  - Location
    - Physical, Orbital, Virtual
  - Risk
  - Time
    - Duration, Timeframe, Point-in-Time
What is a System?

People (Asset)  Things (Asset)  Processes (Actions)

Characteristics
- Operability
- Suitability
- Survivability
- Trainability
- Understandability

Affordability
Adaptability
Agility

Usability
Verifiability
Vulnerability

Integratability
Performability
Repeatability
Systems Engineering Lifecycle

Current Operations and Support

Future Operations and Support

Program Management

Affordability
Deployability
Flexibility
Interoperability
Maintainability
Operability
Reliability
Serviceability
Supportability
Upgradability
Usability

INCOSE Systems Engineering Handbook v. 3.2.1
INCOSE-TP-2003-002-03.2.1 January 2011

© 2011 Systems and Proposal Engineering Company. All rights reserved.
Systems Engineering Lifecycle: Traceability

Current Operations and Support
- Architecture Development
- System Design
- Hardware/Software Acquisition

Future Operations and Support
- Operational T&E and Transition
- Integration and Test

Demolition and Disposal

Traceability

Decomposition

Integration

Program Management

INCOSE Systems Engineering Handbook v. 3.2.1
INCOSE-TP-2003-002-03.2.1 January 2011
Systems Engineering Lifecycle: Integratability
Systems Engineering Lifecycle: Verifiability

Current Operations and Support

Future Operations and Support

Demolition and Disposal

Architecture Development

Operational T&E and Transition

System Design

Verifiability

Hardware/Software Acquisition

Verifiability

Decomposition

Integration

Program Management

© 2011 Systems and Proposal Engineering Company. All rights reserved.
Measurement of –ilities

• Standard measurements – not always agreed to
• Some accepted measurements
  – Availability -  \( P_A = 1 - \frac{MTTR}{MTBF} \)
  – Maintainability - MTTR  *mean to repair (or restore)*
  – Reliability – MTBF  *mean time between failure*
  – SW Maintainability - Lines-of-code measures, McCabe Measures, Halstead Complexity Measures
  – Security – Malware statistics, Firewall statistics, Vulnerability
-ility Related Research

• 2006-2007  John W. Dahlgren MITRE
  – “System Complexity, the “ilities” and Robustness” Project

• Current  - SEAri Systems Engineering Advancement Research Initiative - MIT
  – “Ingenuity, Innovation, and the ilities: Creating Capabilities for the Long Run“
Increasing Emphasis and Demand

In DoD

• Interoperability
  – CJCSI 6212.01E  Interoperability And Supportability Of Information Technology And National Security Systems

• Producibility
  – DoDI 5000.02  Operation of the Defense Acquisition System

• DOTMLPF (Doctrine, Organization, Training, Materiel, Leadership and education, Personnel, and Facilities)
  – Embedded throughout Joint and Service Standards

Commercial World…
Even Commercial Interest is Increasing
Summary

• Little recent SE discussion and writing on ‘-ilities’
• ‘-ilities’ are key system attributes
• Many useful and/or necessary ‘-ilities’ are
  – Not understood well
  – Often forgotten…or ignored
• Systems Engineers should work to integrate more –ilities into systems development

• Recommendation: Increase discussion and interchange among SEs on the topic of ‘-ilities’ and how to best incorporate them into SE