33rd Annual INCOSE international symposium
hybrid event
Honolulu HI USA

# Evaluating 50,000 UAV Concepts against Volatile Requirements

# Purpose of design effort

- To identify a design solution which is consistent as possible with all design requirements
- How do we do this without knowing all the requirements?

# Coarse to Fine Design

- Set initial requirements, then make decision decisions
  - Gather information, set further requirements, then make design decisions
    - Gather information, set further requirements, then make decision decisions

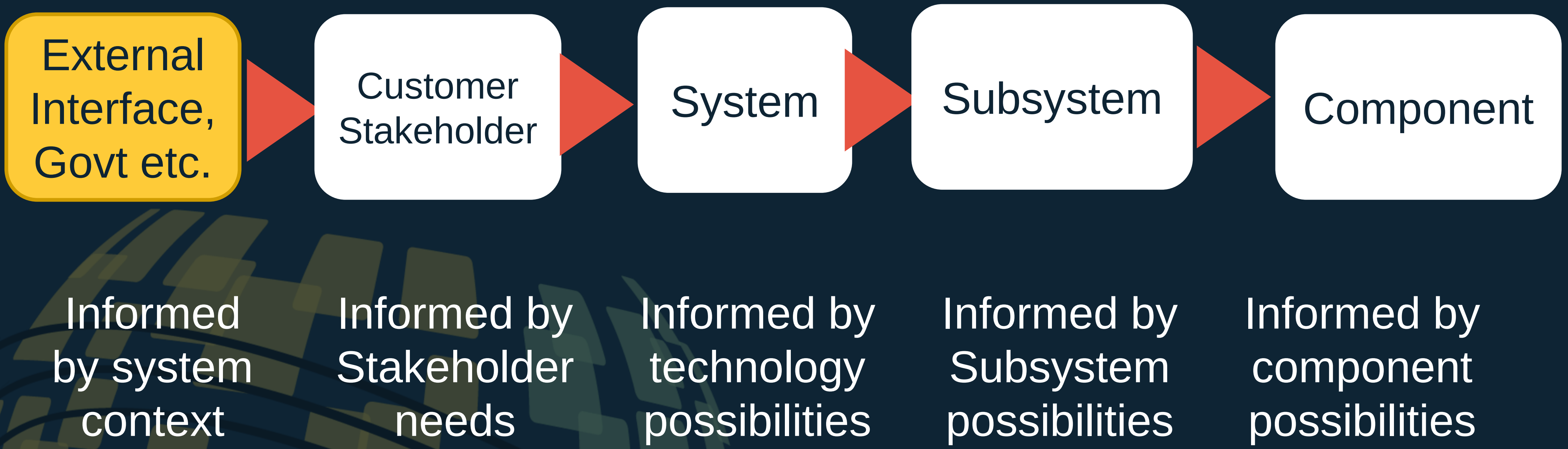
  Repeat until targets and architecture is detailed enough to design component level solutions

- Then verify component level solutions against their requirements
    - Integrate components and verify integration against lowest level architecture requirements
      - Integrate assemblies and verify integration against next lowest level architecture requirements

      Repeat until complete system is developed

- Validate requirements in intended environment

# Progressive Setting of Hard Requirements



# Standard Approach

- Process is valid if
  - Requirements, once set, never change:
    - But 6% chance of core software requirements changing after they have been set
  - Process of adjusting requirements is easy
    - Requirement change is typically disruptive
    - Agile approaches are sometimes (but not always) applicable to reduce chance of design change

# Admit Volatility is Inevitable

Recognize current targets may change to `eventual' targets

- Current requirements only an estimate of noisy requirements

No design solution `satisfies' requirements

- Always some risk of solutions not satisfying requirements

Instead of designing to maximize certain criteria subject to physical and target constraints

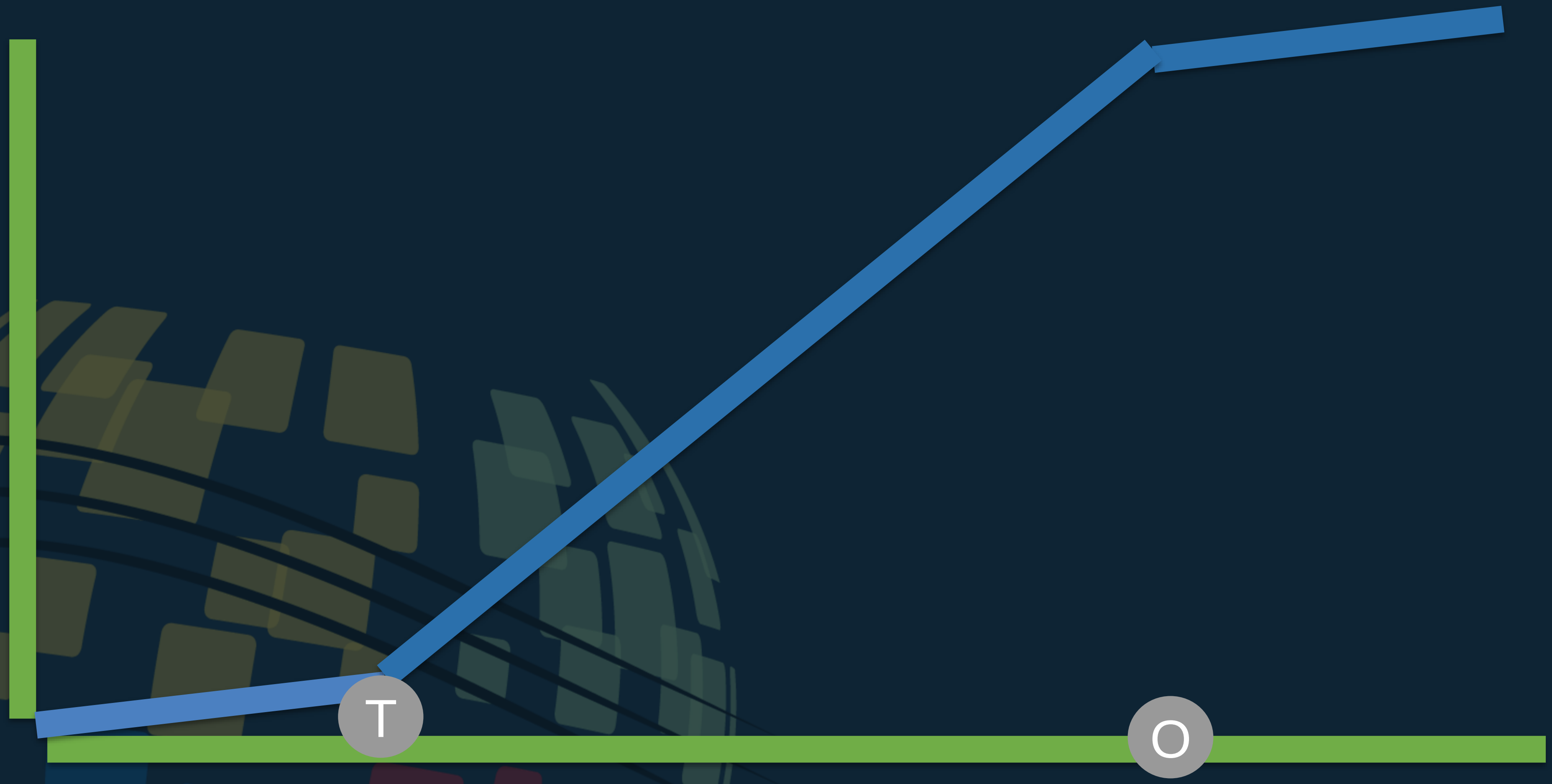
- Design to minimize risk of not meeting requirements subject to physical constraints

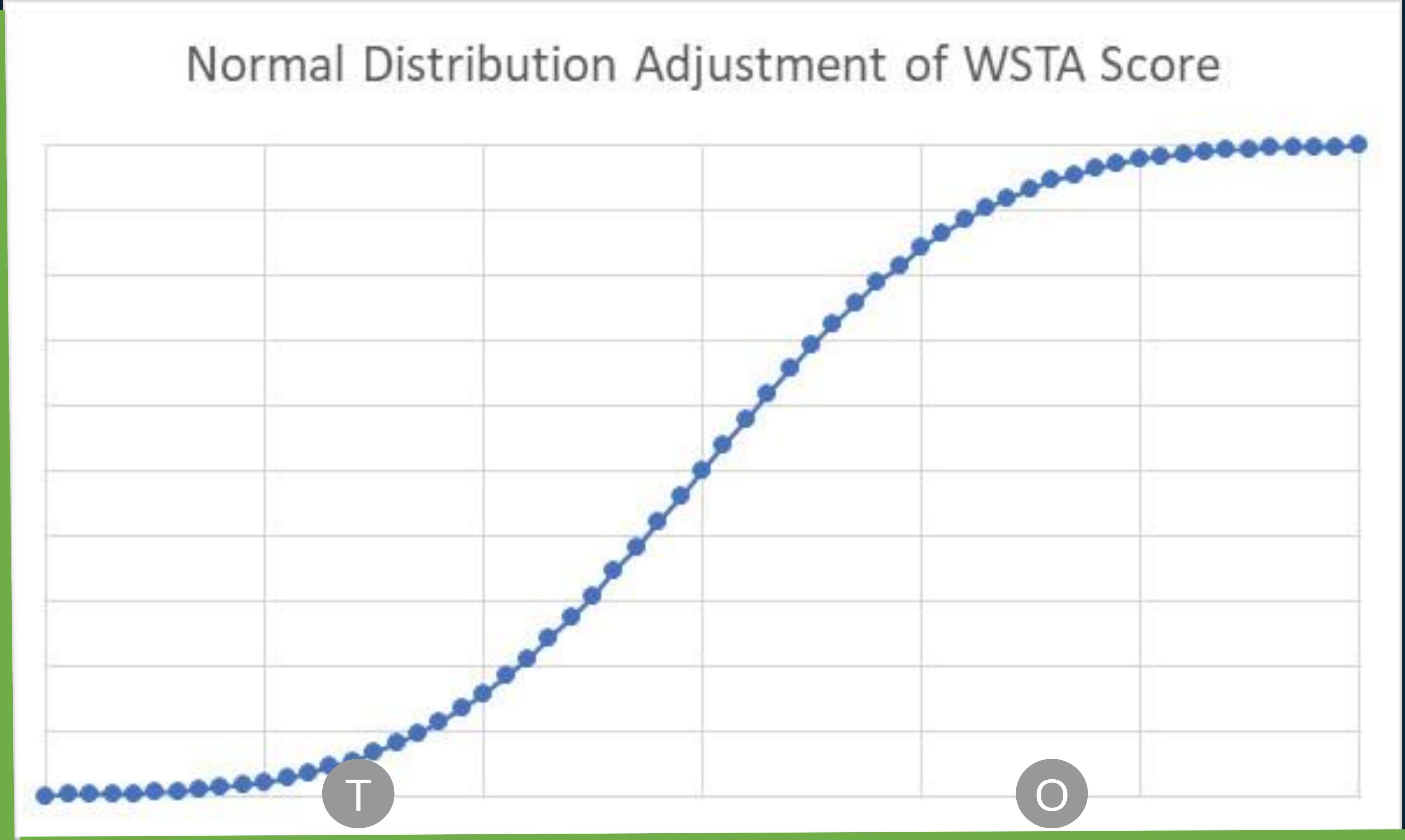
# Parnell,Cilli's Whole Systems Trade Analysis

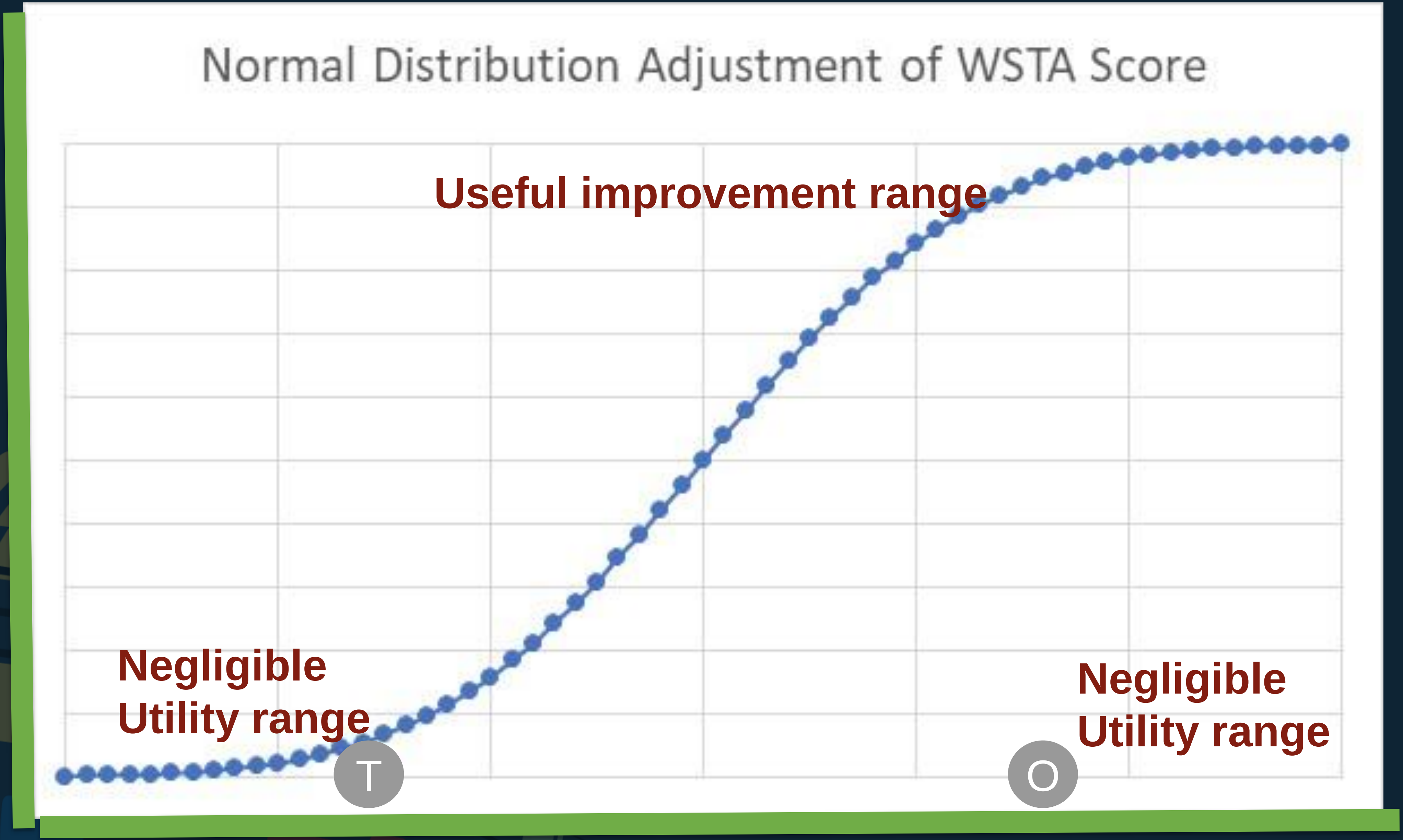
Two targets specified for each requirement

- Threshold level: below which there is negligible value
- Objective level: above which further improvement has negligible value
- Incremental value to designs improving beyond threshold until they reach objective

# Whole Systems Trade Analysis Scoring Rule




Normal Distribution Adjustment of WSTA Score
T
O


Normal Distribution Adjustment of WSTA Score
Useful improvement range
Negligible
Utility range
T
O
Negligible
Utility range

# Normal Utility as Scoring Function

Normal Utility

- Scores degree of compliance with requirements
- Not a probability measure

Mean:

Not the mean of an uncertain quantity

Standard deviation:

Not the standard deviation of an uncertain quantity

# Normal Utility as Scoring Function

Mean: $\frac{threshold\ plus\ objective}{2}$

Suppose maximum negligible utility is 2.5%

- Then useful improvement range includes 95% of the utility
- Analogous to `confidence/credibility interval' in statistics
- Standard deviation, s, is approximately $\left|\frac{objective\ mnus\ threshold}{4}\right|$

# Measuring Volatility

## Treat mean value as normally distributed

- With mean m and variance v
- Variance describes volatility

## Then standard deviation changes from s to

$$\sqrt{s^2 + v}$$

# Modeling Multiple Requirements

- Utility for a single requirement:
  - Normal distribution
- Utility for multiple requirements
  - Multivariate normal distribution
  - Requires specification of correlation between requirements
    - Positive correlation: satisfying one requirement is more useful when another requirement is also satisfied
    - Negative correlation: satisfying one requirement is more useful when another requirement is not satisfied

# Advantages of Utility

- Handles Volatility
- Addresses interaction between requirements
- Easily computed using spreadsheets
- Easy to Explain: Uses the Normal distribution
- Theoretically defensible
  - Extends standard utilities to requirement-based design
- Quality Interpretation:
  - Degree of conformance to requirements

# Handling Emergent Requirements


User has initial requirements

But seeing solutions based on those requirements leads User to adjust those Requirements

I will know it when I see it

# How to handle emergent requirements

Problem:

- User adjusts requirements after seeing initial solutions to those requirements
- Iteration exhausters human designers

Solution

- Have generative algorithms create feasible designs
- Have users adjust requirements based on artificial feasible design
- Once requirements stabilize, issue request for proposals to have human design teams begin  work on more stable* requirements.

*changes in external environment can still necessitate requirement changes

# Application: Set-Based Design

Tentative requirements developed with three levels

- Marginally acceptable
- Fully acceptable
- Stretch target

Key physical characteristics of different drone designs specified

Based on these key characteristics of UAV designs, models predicted the uncertain score of different designs on eleven fundamental objectives, e.g.,

- Easy of carrying drone
- Dwell time
- Ability to detect human movement at night

# Application: Use of Normal Utility

50,000 designs created by set-based design

Computationally demanding to evaluate designs with piecewise linear scoring function

Normal Utility used instead

For each requirement,

- Mean define by fully acceptable level
- Standard deviation defined by quarter of absolute difference between stretch target and marginally acceptable
- Anticipated volatility in requirement used to increase the standard deviation

Interactions between every pair of  requirements modeled with correlation coefficients

# Application: Use of Normal Utility

Pareto chart plotted of overall utility

- versus cost, schedule, risk

Small set of dominant solutions developed

User team

- discussed strengths and limitations of each solution
- Developed changes to tentative requirements
- Example: reduced the proposed budget for design effort