Value Design:

selecting design and architecture, based on qualities and costs,
a serious engineering approach

Quarentining
And preparing course
Today at my Summer cabin
On the Oslofjord

(a 3 Hours BCS SPA SG ** Online Course)

For initial presentation
Wednesday 29nd April 2020, 19:00 to 21:00 +

Video URL= https://www.youtube.com/playlist?list=PLKBhokJ0qd3_wlvr0j85YhmNfNj8ZJ8M-

Kolbotn Lake, 5 minutes walk
from my flat
(near Oslo)

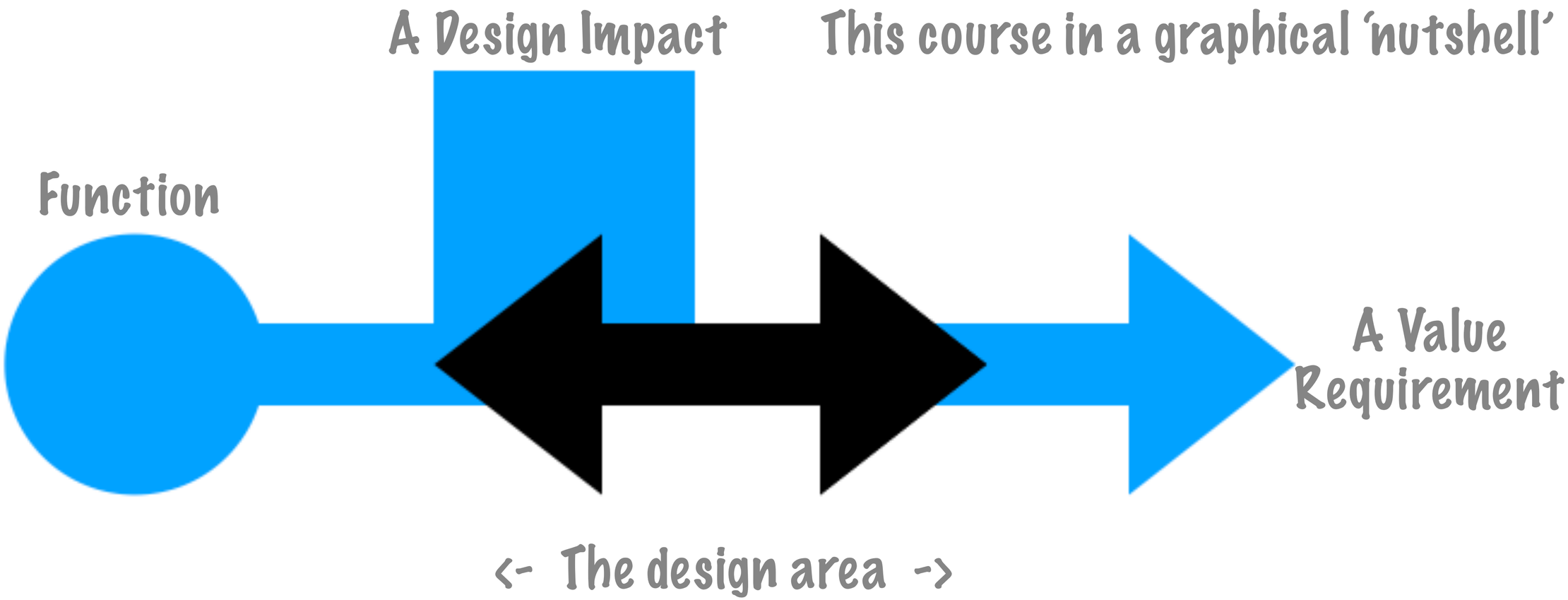
Slide Location: = http://concepts.gilb.com/dl972

By Tom Gilb, in Norway (Kolbotn, near Oslo)
tom@Gilb.com
www.Gilb.com
@ImTomGilb (Twitter)
www.linkedin.com/in/tomgilb

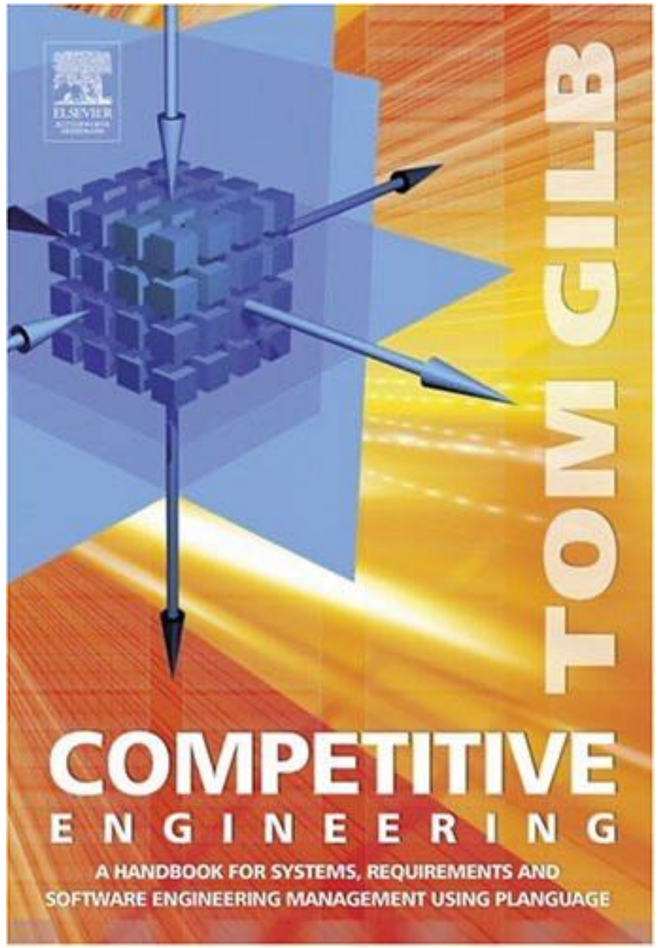
Co-sponsored by BCS Specialist Group on Quality

** British Computer Society, Specialist Group SPA, Software Practice Advancement, http://www.bcs-spa.org/index.php

The ‘Design Theory’ Icon

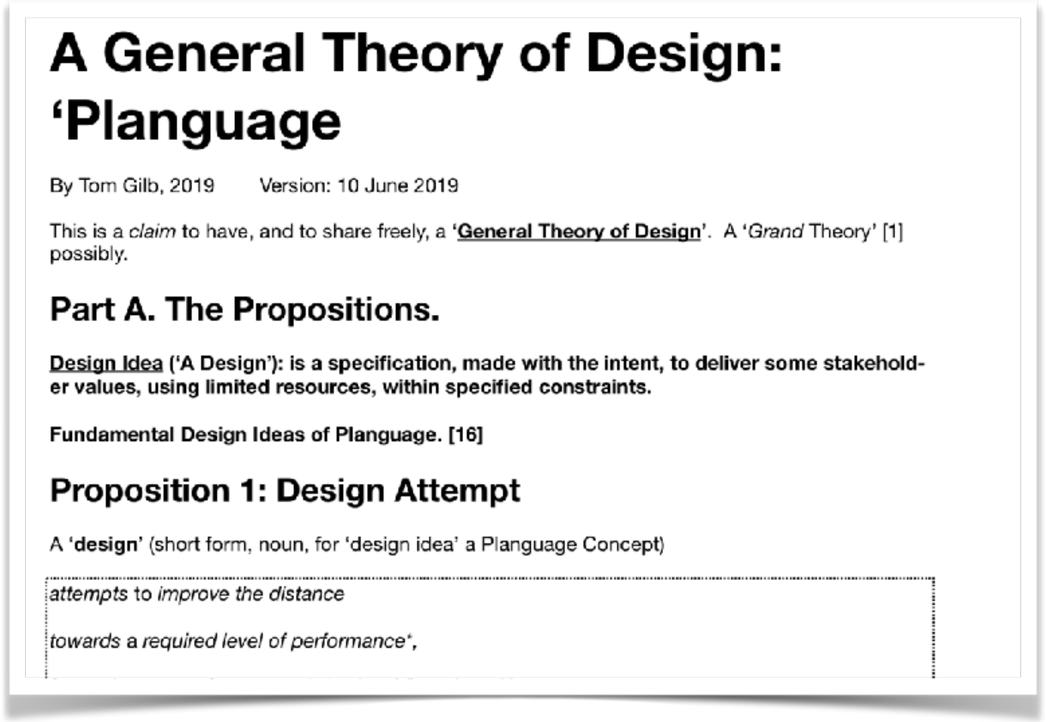


"A General Theory of Design: 'Planguage' "



Get a free e-copy of 'Competitive Engineering' book.
https://www.gilb.com/p/competitive-engineering

(The 'Planguage' design language standard



A General Theory of Design: 'Planguage

By Tom Gilb, 2019 Version: 10 June 2019

This is a *claim* to have, and to share freely, a '**General Theory of Design**'. A '*Grand* Theory' [1] possibly.

Part A. The Propositions.

Design Idea ('A Design'): is a specification, made with the intent, to deliver some stakeholder values, using limited resources, within specified constraints.

Fundamental Design Ideas of Planguage. [16]

Proposition 1: Design Attempt

A '**design**' (short form, noun, for 'design idea' a Planguage Concept)

attempts to *improve the distance*

towards a *required level of performance*',

http://concepts.gilb.com/dl956

51 Page paper

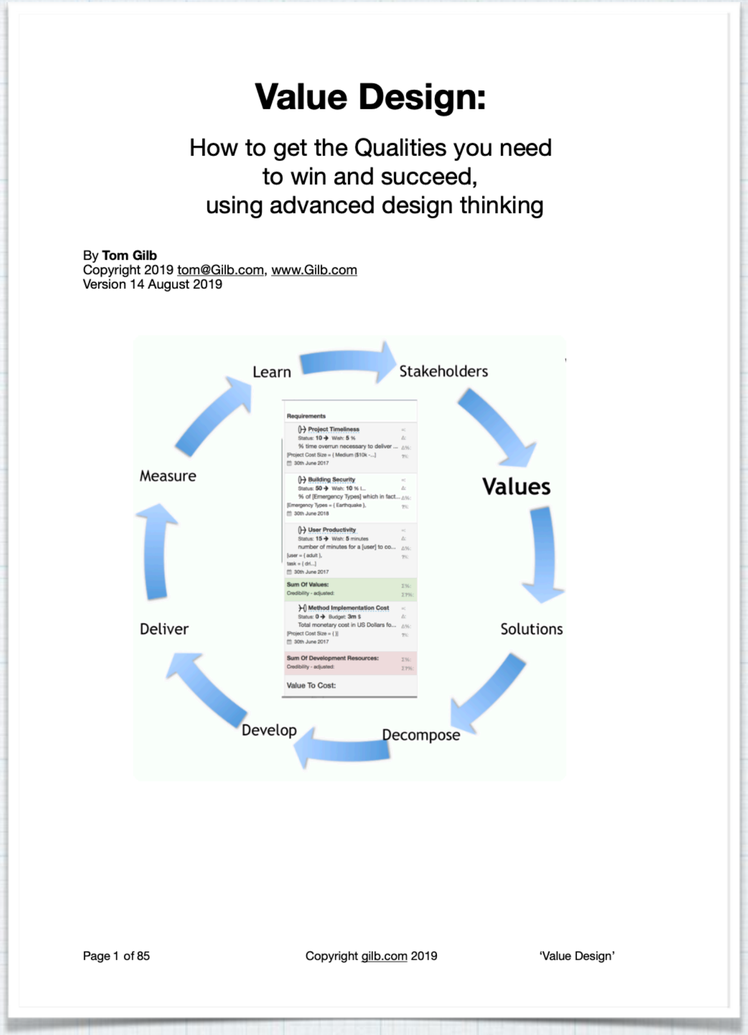
3 Hour Course Content

Main subjects:
Value Requirements: the foundation (previous value objectives course, not a prerequisite, but quite useful) of the design problem articulation.

What is Design?

1. **Basics for Valid Design**
2. **How not to evaluate a design**
3. **Simple ways to evaluate and compare designs**
4. **More powerful ways to evaluate design options**
5. **Starting a larger project, first design week.**
6. **Decomposing designs, some more ideas**
7. **Risk management for value design**
8. **Design Prioritisation.**
9. **Dynamic Design to Requirements**
10. **Organising the value design process**

The star of this show is the Value Decision table aka Impact Estimation Tables) which allows us to see any set or level designs, with their estimated or real effects on our critical objectives and costs. A unique general tool for all designers. Complex systems at a glance, on a page or screen.



Value Design:

How to get the Qualities you need
to win and succeed,
using advanced design thinking

By **Tom Gilb**
Copyright 2019 tom@Gilb.com, www.Gilb.com
Version 14 August 2019

Page 1 of 85 Copyright gilb.com 2019 'Value Design'

Documentation: the course is based on our recent digital textbook

"Value Design:
How to get the Qualities you need to win and succeed,
using advanced design thinking."

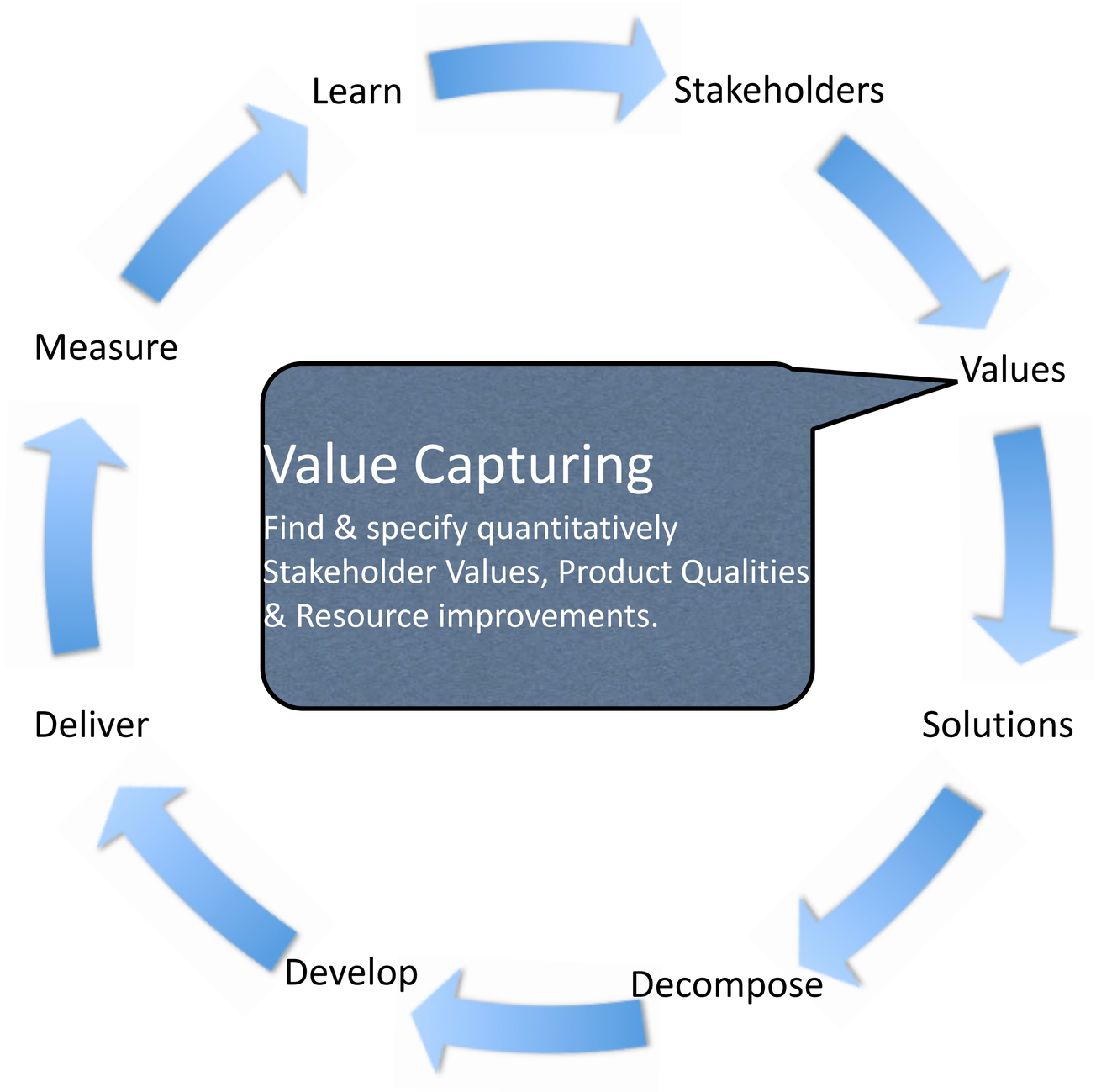
Book Price normally = £1,000 (as of April 1 2020)
Digital copy free to course participants.
Book Knowledge Value > £1,000,000,000

https://tinyurl.com/FreeValueDesign

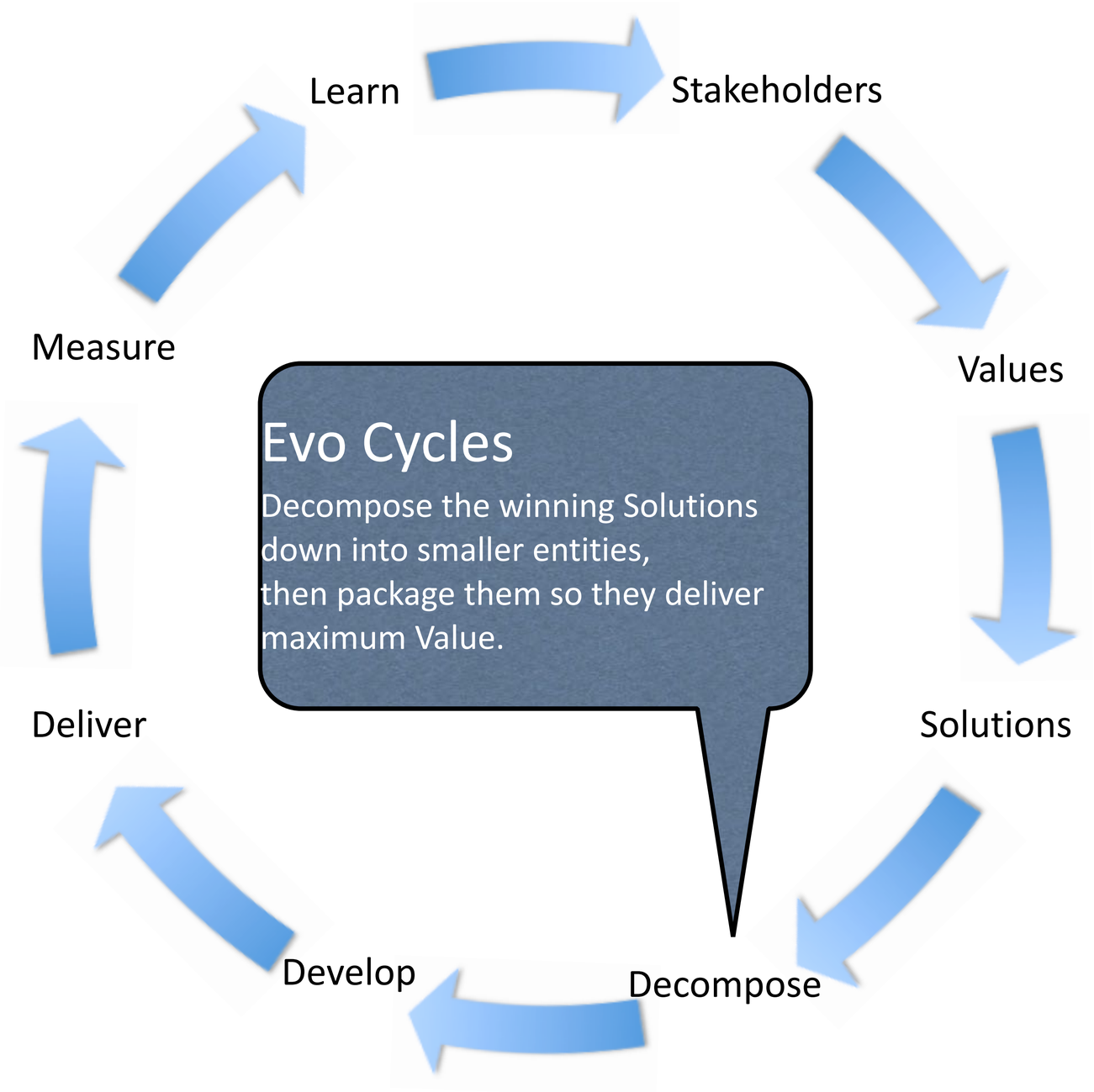
Gilb Value Cycle

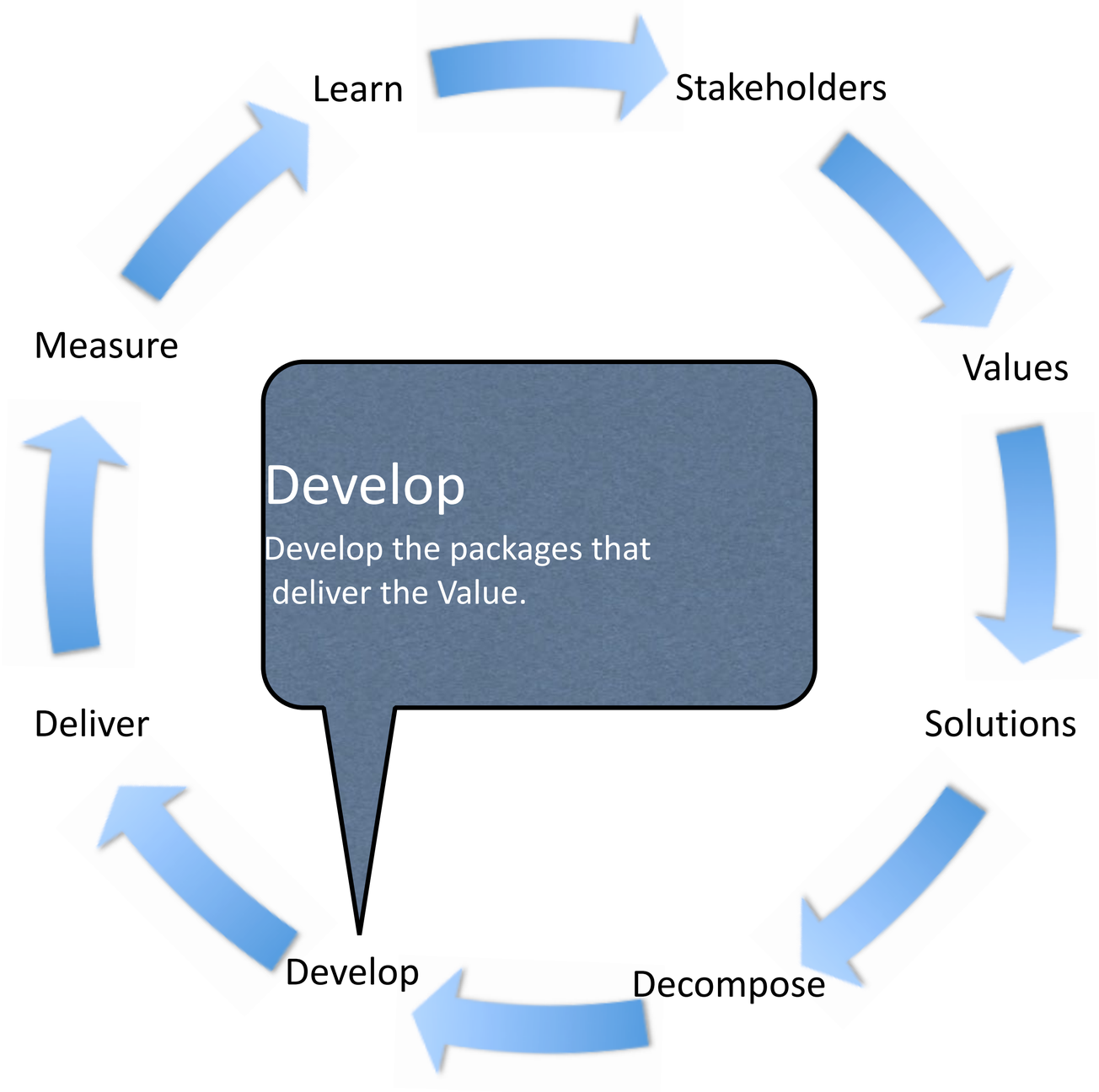
Copyright 2020 kai@Gilb.com

Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Identify Stakeholders
Who and what cares about the outcome of our project?

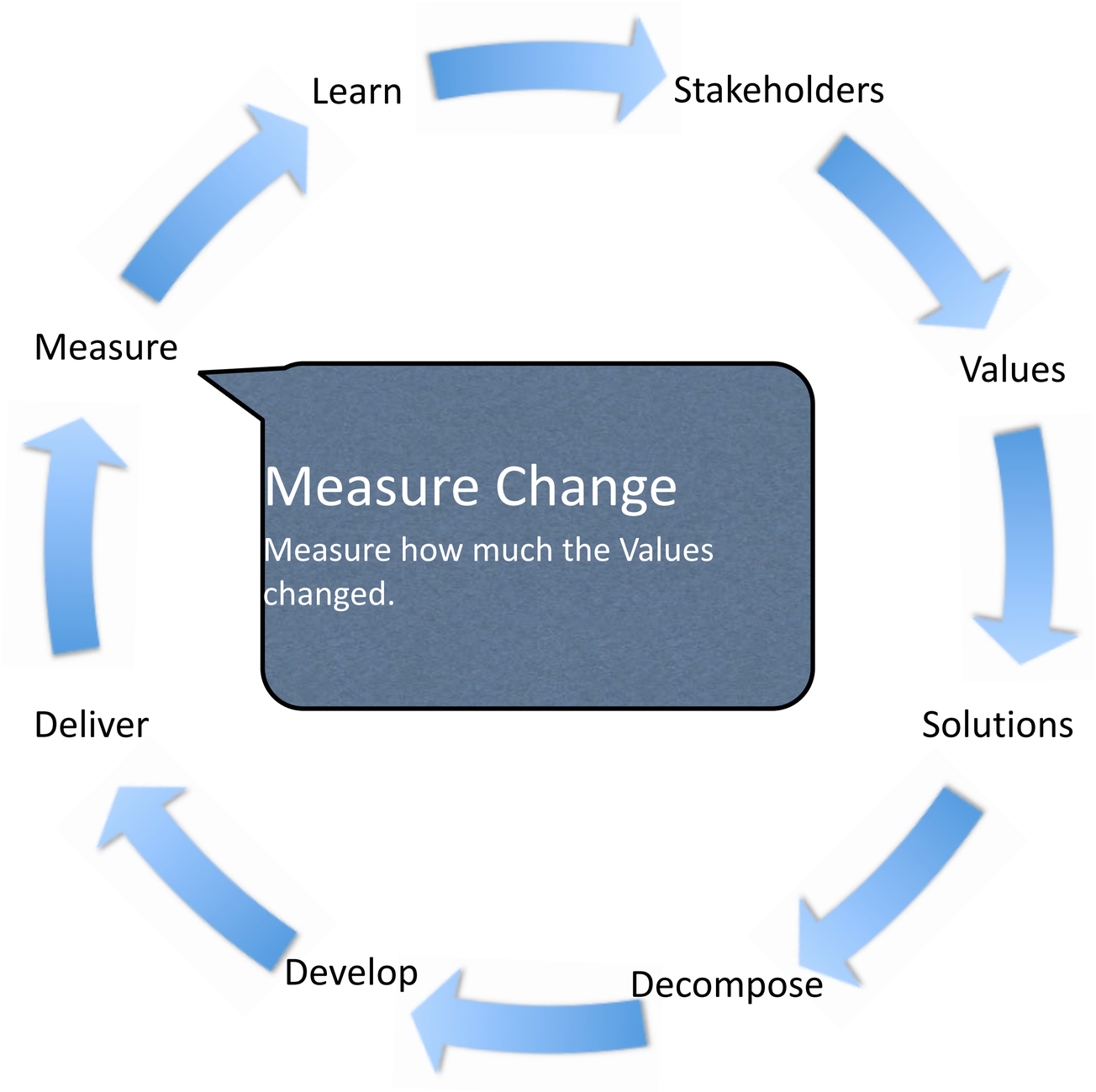

Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Value Capturing
Find & specify quantitatively
Stakeholder Values, Product Qualities
& Resource improvements.

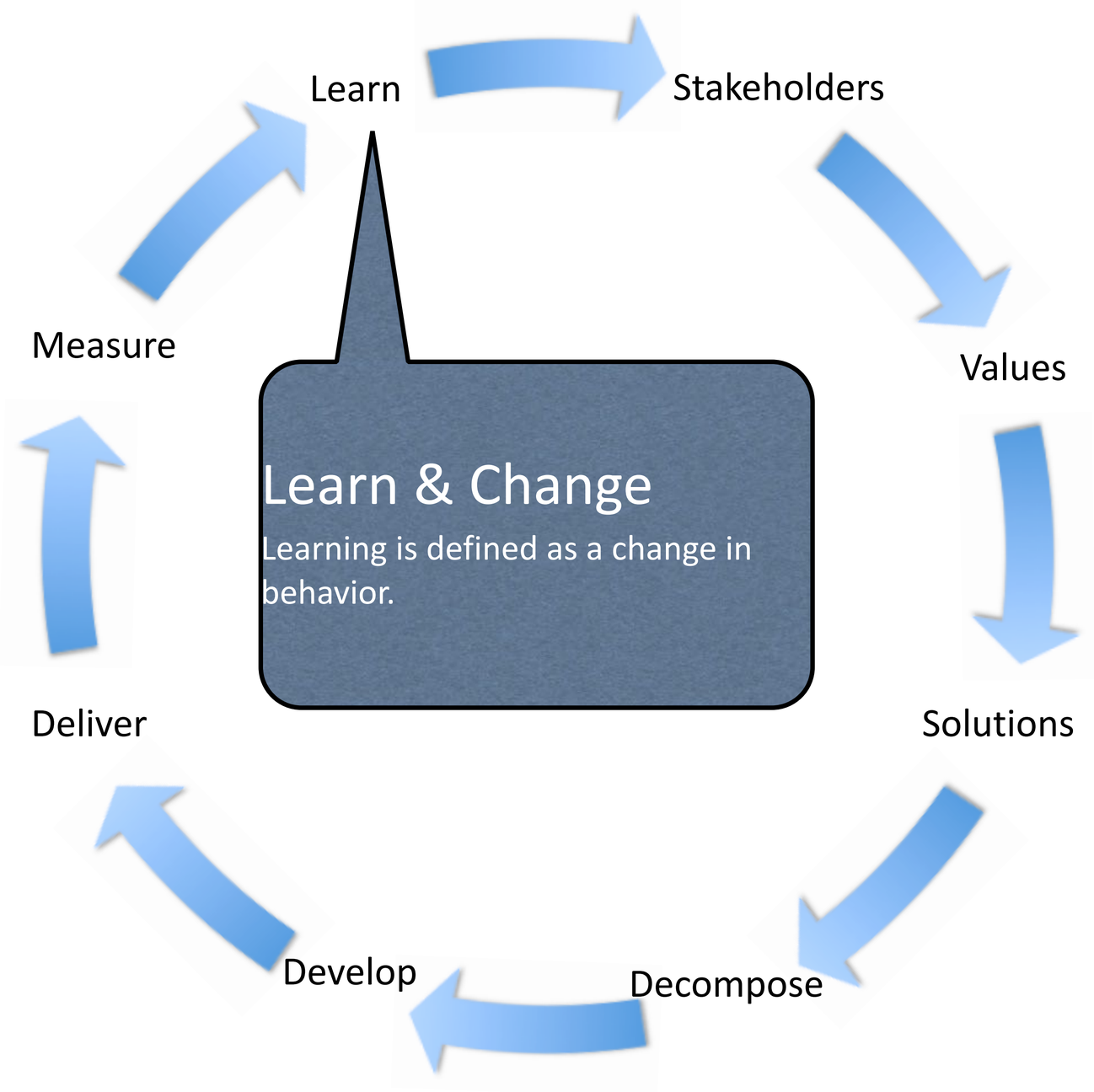
Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Solution Prioritization
Find, Evaluate & Prioritize Solutions
to satisfy Requirements.


Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Evo Cycles
Decompose the winning Solutions down into smaller entities, then package them so they deliver maximum Value.

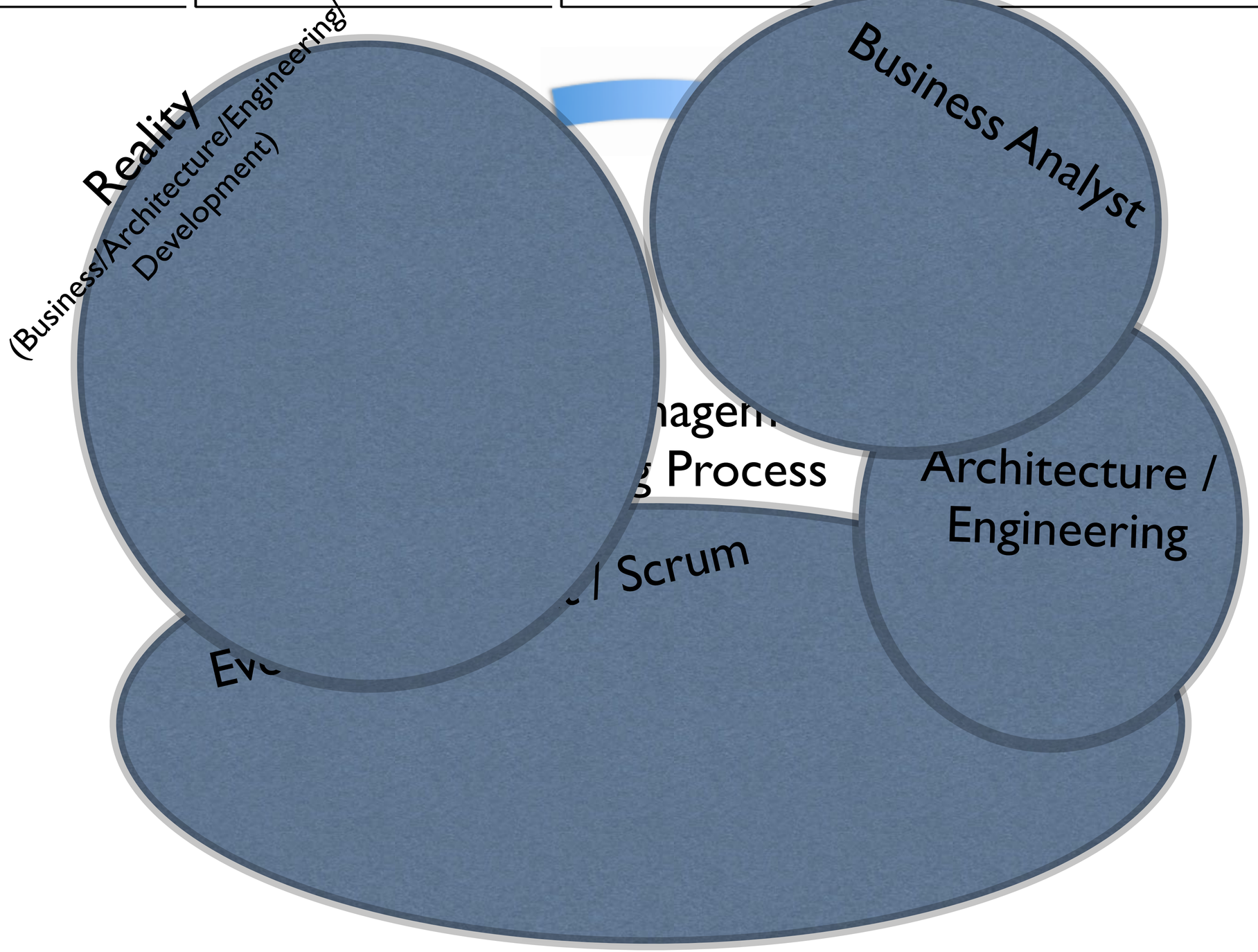

Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Develop
Develop the packages that deliver the Value.

Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Deliver
Deliver to Stakeholders
improved Value.
(not always a thing or code)


Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Measure Change
Measure how much the Values changed.


Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Learn & Change
Learning is defined as a change in behavior.

Learn
Stakeholders
Values
Solutions
Decompose
Develop
Deliver
Measure
Value Management
Learning Process


Reality
(Business/Architecture/Engineering/
Development)
Business Analyst
Process
Architecture /
Engineering
/ Scrum

Learn

Stakeholders

Measure

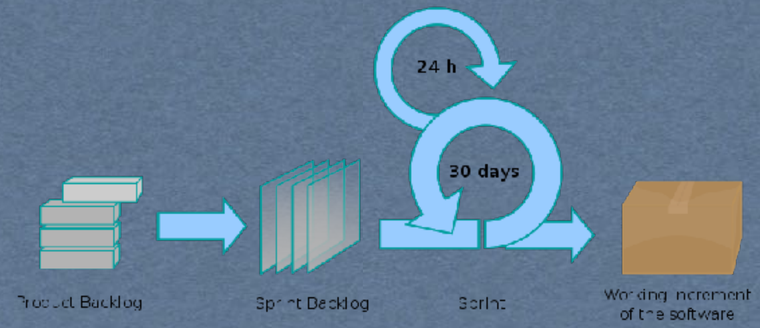
Values

Value Management
Learning Process

Scrum

Deliver

Solutions



Develop

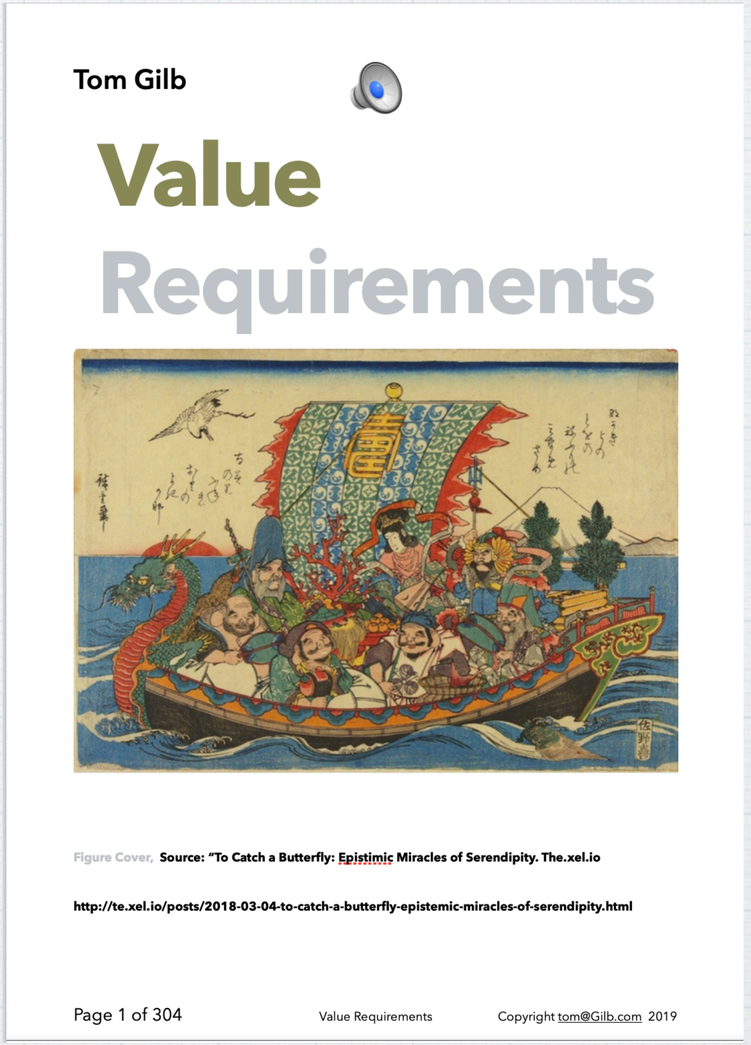
Decompose

Value Requirements: the foundation (previous value Requirements course, not a prerequisite, but quite useful) of the design problem articulation.

Wednesday 22nd April 2020

Video URL= https://lnkd.in/dNEDuc6

Slide Location: = http://concepts.gilb.com/dl970



https://www.dropbox.com/s/hxg1rx9rzesw2id/Value%20RequirementsPDF%20BEST%20%2070MBQ%20011019%202245%202.pdf?dl=0

Free download During Corona

Normal Price £ 1,000 from 1st April 2020

Value: > £1 million/project. :) avg.

Value Requirements and Resource Constraints

Requirements

Project Timeliness
Status: **10** → Wish: **5** %
% time overrun necessary to deliver ...
[Project Cost Size = { Medium ($10k -...]
30th June 2017

Building Security
Status: **50** → Wish: **10** % l...
% of [Emergency Types] which in fact...
[Emergency Types = { Earthquake }.
30th June 2018

User Productivity
Status: **15** → Wish: **5** minutes
number of minutes for a [user] to co...
[user = { adult },
task = { dri...]
30th June 2017

Sum Of Values:
Credibility - adjusted:

Method Implementation Cost
Status: **0** → Budget: **3m** $
Total monetary cost in US Dollars fo...
[Project Cost Size = { }]
30th June 2017

Sum Of Development Resources:
Credibility - adjusted:

Value To Cost:

Which numeric improvements do stakeholders need, *critically*?

We can, and must always, express *their* values with *well-defined* numbers

Define both failure and success numerically

and

keep learning what those critical numbers are continuously

In our planning language, Planguage, these 3 levels might be expressed like this.

Security is the reference tag for the entire specification.

Scale is a parameter in Planguage for defining a value variable, such as Security, so that the various levels of Security can be expressed numerically.

Status gives us the moving current change of status in the level. Step by step feedback.

Tolerable gives us the bare minimum level which is acceptable. Worst acceptable case.

Wish is the stakeholder-desired, or stakeholder -needed, level of Security, on that Scale.

Wish = The 'Success level.

Security:

Scale: % probability of detecting a hacker within 5 seconds.

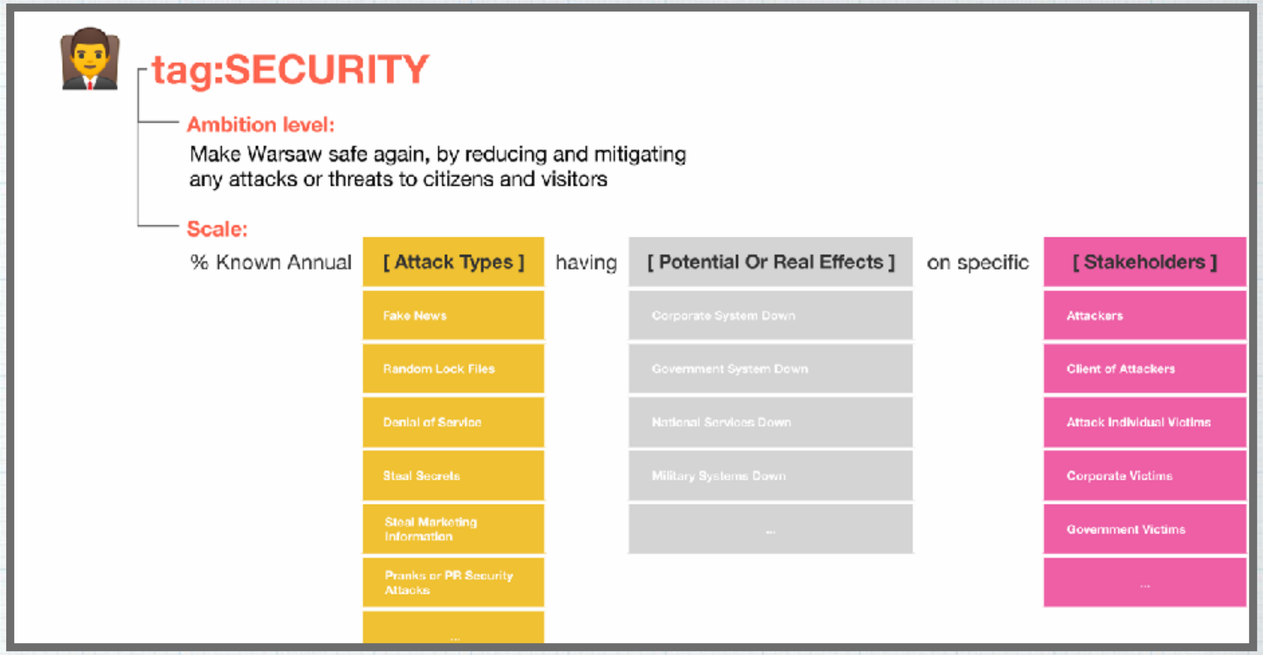
Status: 10% last year.
(Benchmark level)

Tolerable: 80% by End this year.
(Constraint Level)

Wish: 98% by End Next Year.
(Target Level)

Scale [Parameters]

- **[Scale Parameters] might seem 'complicated' at first sight.**
- **But it is in fact a way of *simplifying* very complex problems,**
  - **by allowing us to carefully extract, something simple**
  - **that we can work on, and**
  - **deliver some value improvements <u>early</u>, for critical subsets. (Hint: 'Agile as it should be')**
- **Early partial value delivery is *also* about**
  - **'learning about complex realities',**
  - **…. before we commit to 'scaling up'.**



Scale: Star Rating number for [**Person Type**] and [**Car Specs**] for [**Safety Equipment**] with [**Alternative Model Validity**] for a [**Publication Date**] by a [**Rating Agency**].

Wish: 5 Stars, by Next Year,

Person Type = All,

Car Specs= {Tesla 3, RWD, 4 Door, 2019}, **Safety Equipment**= {Front Airbag, Belt Pretensioner, Belt Load Limiter, Knee Airbag, Side Head Airbag, …},

Alternative Model Validity=Dual Motor AWD Model 3,

Publication Date =2019,

Rating Agency= All.

Air Quality

Level: Stakeholder, **Type:** Value, **Labels:** critical Edit

Is Part Of: Top Values

Status	Status	Tolerable	Tolerable	Goal	Goal	Stretch
9.5k	**1.5k**	**200**	**100**	**950**	**150**	**0**

Goal [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] @ 2028 : **950** People <- *Sadiq Khan, Mayor of London Website, Press Release Toxic Air*

AQ.Ambition Level: Drastically improve air quality in London to acceptable legal levels as stated in the Paris Accord (Paris Agreement)

Scale: Number of [Persons] who reside in London Boroughs dying from exposure over [Time] to [Pollution] per year in [Area]

Meter: Recent Hospital records from London hospitals for deaths by Pollution Exposure related illness

Status: 9.5k People [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] When 2019

Status: 1.5k People [Persons = **Senior**, Pollution = **NO2**, Area = **<All>**] When 2019

Tolerable: 200 People [Persons = **Senior**, Time = **5 years**, Pollution = **NO2**, Area = **Greater London**] When 2022

Tolerable: 100 People [Persons = **Child**, Time = **1 Year**, Pollution = **{NO2, Carcinogens}**, Area = **Greater London**] When 2020

Goal: 950 People [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] When 2028

Goal: 150 People [Persons = **Senior**, Pollution = **NO2**, Area = **Greater London**] When 2029

Stretch: 0 People [Persons = **Senior**, Time = **5 years**, Pollution = **NO2**, Area = **Greater London**] When 2030

Stakeholders: Mayor Of London. Mayor of London

Assumption: IssueActionStatistic on number of deaths from pollution per year from Mayor of London press

We're Online!
How may I help you today?

Detailed 'Value Requirement' specification

(1-line/statement view)

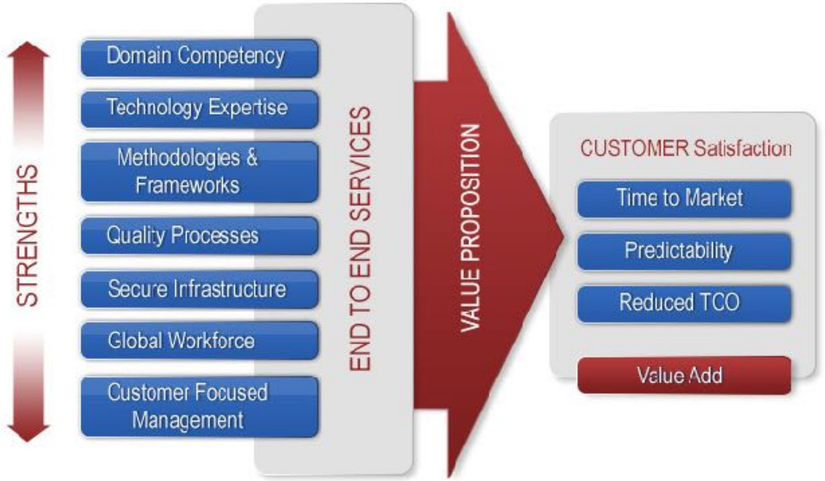
One of the critical value requirements in previous slide

"Air Quality'

Source BCS Exercise Sept 2017, 'London Congestion for Air Quality'. Notice that this is also a definition of 'Project Value' using the 11 decomposed different values, as the definition-by-subset.. Figure 1.19 in Value Requirements book

Predictability of Time To Market:

- **TTMP: Predictability of Time To Market:**
  - *Ambition: From Ideas created to customers can use it. Our ability to meet agreed specified customer and self-determined targets.*
  - **Scale: % overrun of actual Project Time compared to planned Project Time**
    - **Project Time: Defined:** time from the date of Toll-Gate 0 passed, or other Defined Start Event,
      to, the Planned- or Actually- delivered Date of All [Specified Requirements], and any set of agreed requirements.
    - **Specified Requirements: Defined:** written approved Quality requirements for products with respect to Planned levels and qualifiers [when, where, conditions].
      And, other requirements such as function, constraints and costs.
  - **Meter: Productivity Project or Process Owner will collect data from all projects, or make estimates and put them in the Productivity Database for reporting this number.**
  - **Past [1994, A-package] < 50% to 100%> <- Palli K. guess.**
    [1994, B-package] 80% ?? <- Urban Fagerstedt and Palli K. guess
  - **Record [IBM Federal Systems Division, 1976-80] 0%**
    <- RDM 9.0 quoting Harlan Mills in IBM SJ 4-80
  - ***"all projects on time and under budget"***
  - **[Raytheon Defense Electronics, 1992-5] 0% <- RDE SEI Report 1995 Predictability.**
  - **Fail [All future projects, from 1999] 5% or less <- discussion level TG**
  - **Goal [All future projects, from 1999] 0% or less <- discussion level TG**



From Ericsson case study on engineering productivity in the Value Requirement slides

The Evo Process., the One Page Summary

A Systems Engineering Agile Process

Stakeholder Value Requirements

Clarify your environment: critical-stakeholders' territory

1. Identify your critical **stakeholders**: the ones that can make or break your project
2. Identify their critical **values**
3. **Quantify** and **clarify** your critical values: what degree of values do you expect the design to deliver to stakeholders
4. Identify design **constraints**: legality, political, cultural, policy, other plans
5. Identify design **resource-limitations**: time, money, operational costs for example.

Top-Level Design: Architecture Level

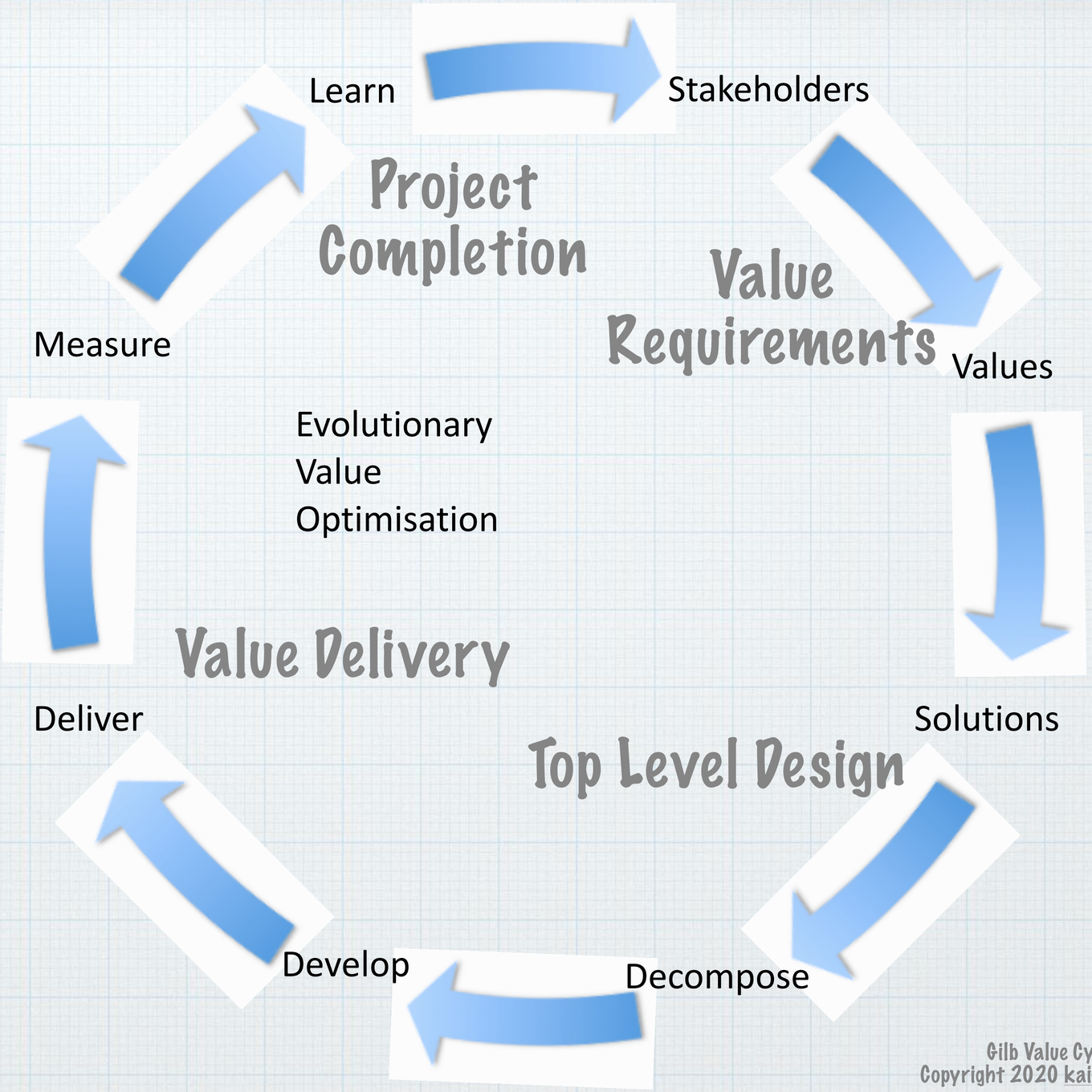
6. The Project Startup Week: an **architecture overview**
7. Identification and **prioritization** of top-level architecture
8. **Decomposition** of top-level architecture into design components.

Value Delivery Cycle (Designs deliver the value !)

9. The **Evo Value delivery steps** (about a week, or 2% of total project budget)
   1. Select your highest **priority value**, and the most-critical scale-parameter attributes.
   2. Find a design component which will deliver the most **value-for-resources** to your priority requirement.
   3. Ready the design component for delivery: **integration to the existing system.**
   4. **Deliver** the design component to the **real system**
   5. **Measure** the results (values and costs) of the design increment
   6. If results are **negative**, attempt design **improvement, and redeliver.**
   7. If results **ok** then **repeat** this value delivery cycle, scale up, until 'done'.

Project Completion: All Value is delivered

10. When **all value requirements are reached**, or when **critical resources are used up**. Stop.



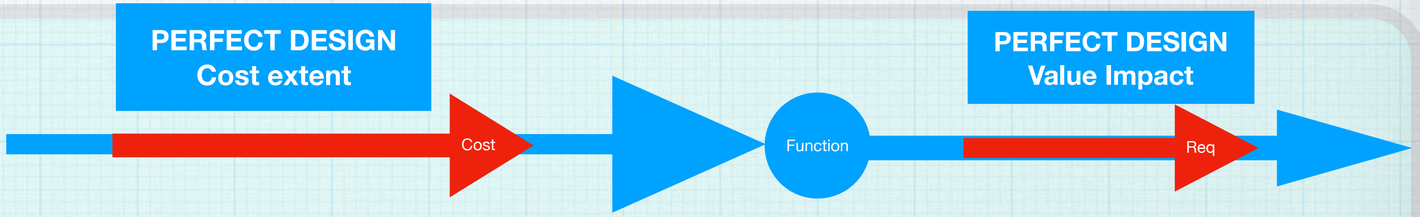
Gilb Value Cycle
Copyright 2020 kai@Gilb.com

Part 0. What is Design ?

A design is

a suggested solution
to the problem specified
by the design requirements.

It might not turn out to be valid, or as efficient as *alternative* solutions.



The Design costs
Do not exceed
Resource constraint,
Budgets

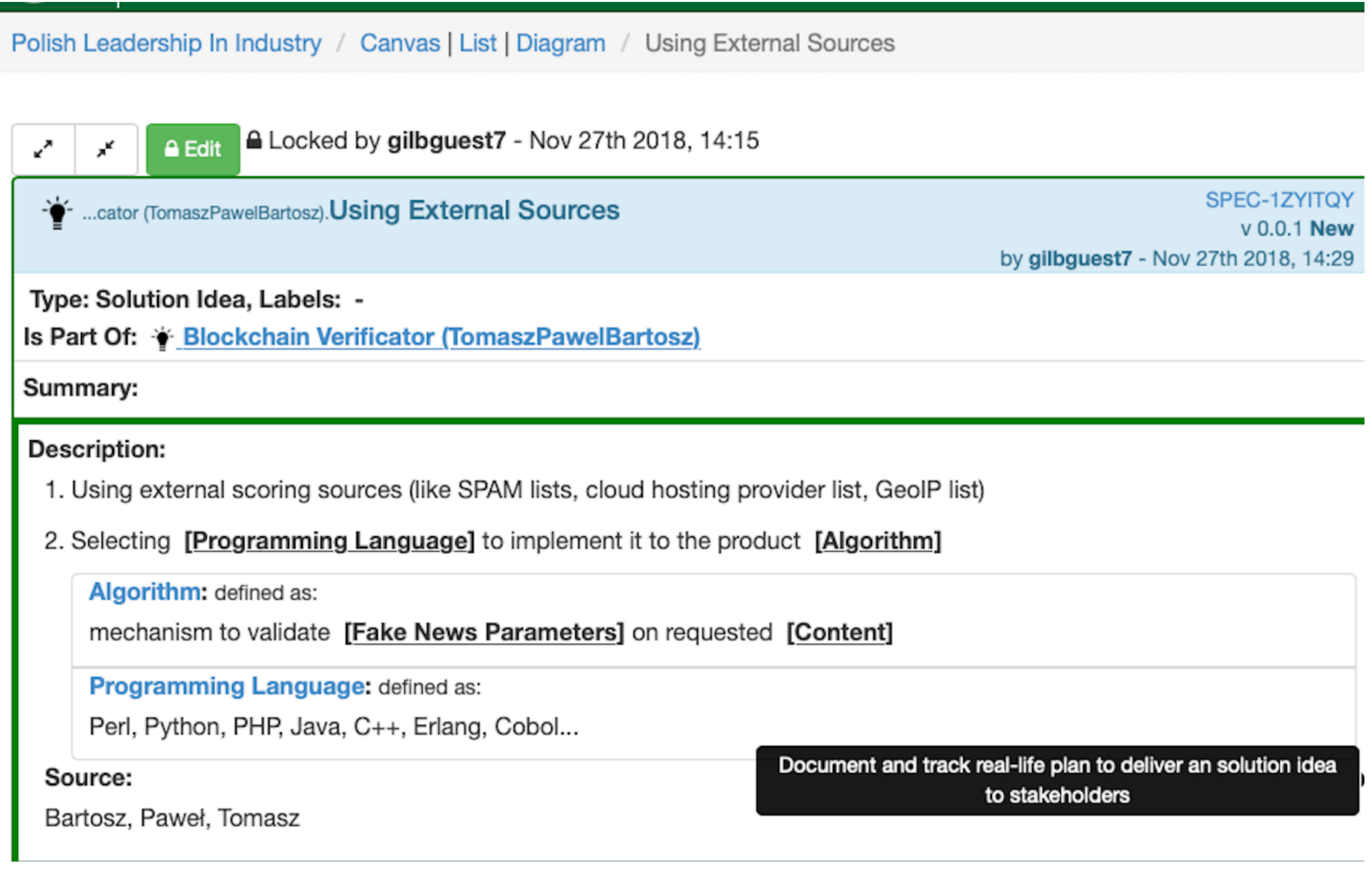
The design impacts
Values
So that all Goal
levels are reached

A perfect valid design specification would

- deliver all value requirement levels within their deadlines
- While not exceeding any resource budgets
- And not exceeding any other specified constraints

'Design' basic concept synonyms:
Architecture, Technology, Means,
Solutions, Strategies

Design Specification Example



Part 1.
Basics for Valid Design

Validating that the design will deliver enough value.

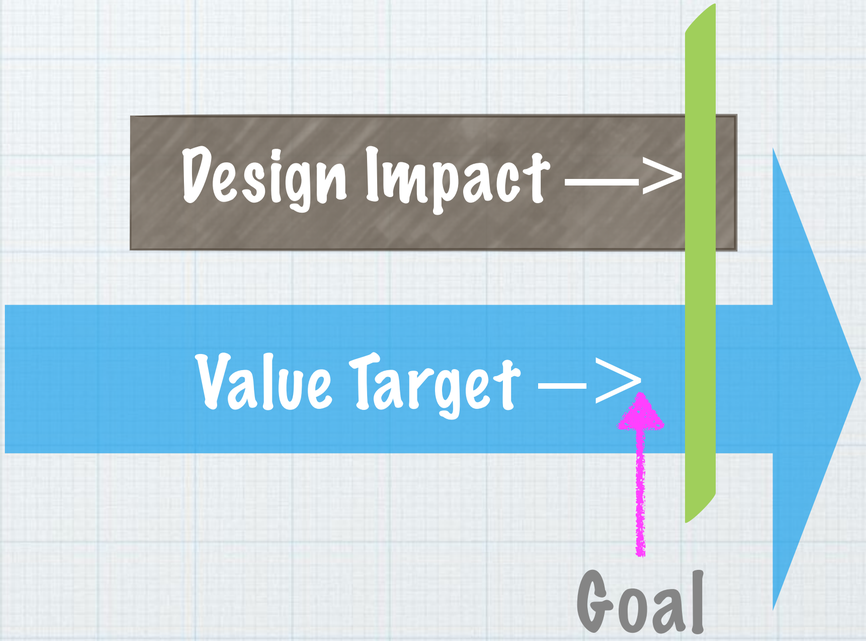
* **ARE WE FOCUSED ON A CRITICAL GOAL AND DEADLINE?** The Value Requirement levels specified (example, Goal 2030, 95%) is the driving force of the design process.

* **ALL CONCURRENT DESIGNS MUST MEET ALL CRITICAL VALUE REQUIREMENTS**: We look at those value requirement specs, and each *one* of our *design problems* is 'defined' by the total set of several value requirement statements.

* **WE NEED TO RESPECT A SET OF OTHER CONSTRAINTS TOO**: We need to ask ourselves, for example
    * **which design will get me to 95% by the year 2030** (including the other [conditions] detail, included in our problem specification)?

* **WHAT IF WE HAVE NO VALID DESIGN IDEAS?** If we have no design ideas at all, and neither does anyone else we communicate with;
    * then we might possibly have a problem spec,
    * for which there is no 'known' design solution.
    * That means we cannot satisfy the requirement level, yet.

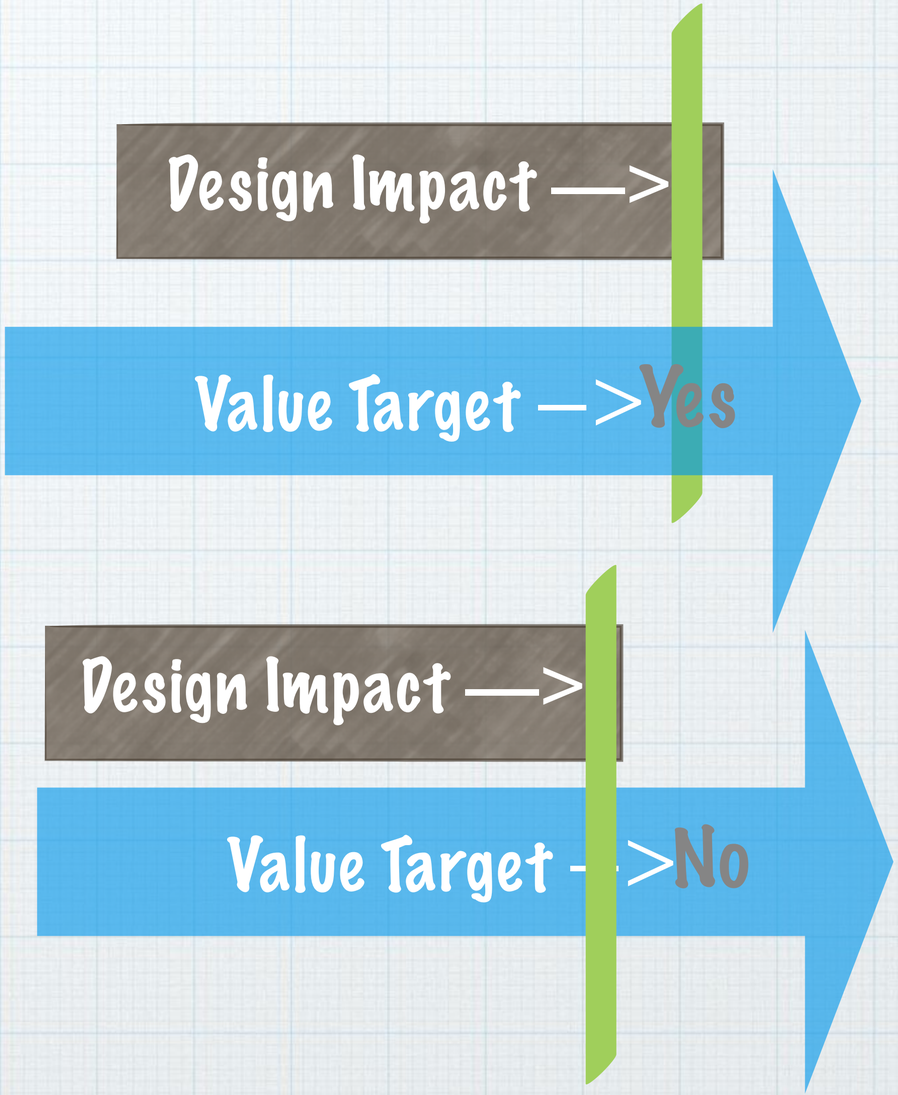
* **WE NEED AN ENGINEERING DISCIPLINE TO VALIDATE MANY DESIGNS AND REQUIREMENTS:** If we have several valid ideas, then the design process needs to get more-disciplined.
    * Looking at several design options
    * Looking for side effects on other value requirements
    * Looking at various resources needed, costs.



More Design Discipline: some questions to ask about your designs.

1. REALISTIC GOAL?: Has anyone ever actually reached such a level (95%) with ***any design*** at all? Even with designs we cannot realistically consider.
2. TIMELY?: Has anyone ever reached such a level within the deadline (2030, X years from when we start)?
3. REALISTIC SUGGESTED DESIGN?: Has anyone ever reached such a level using the design 'hypothesis' (= we *think* it might work) which we have suggested?
4. IS THE DESIGN CONSISTENTLY SUCCESSFUL?: Have people *regularly* reached such a level, using our design hypothesis?
5. IS PILOT EXPERIMENT FEASIBLE?: How costly would it be to try out (pilot, prototypes, experiment) with the design hypothesis, and see how it works for us, before committing to it, on a longer-term, larger-scale basis?
6. IS THE DESIGN SPEC UNAMBIGUOUS?: Is the design idea so *vaguely specified*, that interpretations, in fact, *could* include both winning designs, and *failures*? (and we *might* implement a failure).
    * Do we need to clarify design specs, so the successful interpretations are the only possible interpretations .
7. NO CURE NO PAY GUARANTEE? Would a sub-supplier agree to a no-cure no-pay contract, based on actual value delivered, with '95%' (the Goal) or better, giving a them double bonus?

The answers to these questions will guide you into pursuing or abandoning the design idea, and if necessary looking for other better ideas.



Understanding that we probably have the required resources to use the design. Can we afford it? Are there cheaper ideas?

* Let's assume you have one or more design candidates that are acceptable, in terms of the questions above.

* And let us assume all candidates look roughly as good as any other.

* So they *might* deliver the value levels you require.
  * But can you afford them?
  * And is any option much cheaper or faster than the others?

* We can ask the following questions about the design options, in order to pick a 'resource winner':

1. is the design specified <u>in enough detail</u>, that we can hope to estimate costs roughly ?

   (Order of magnitude, or maximum).

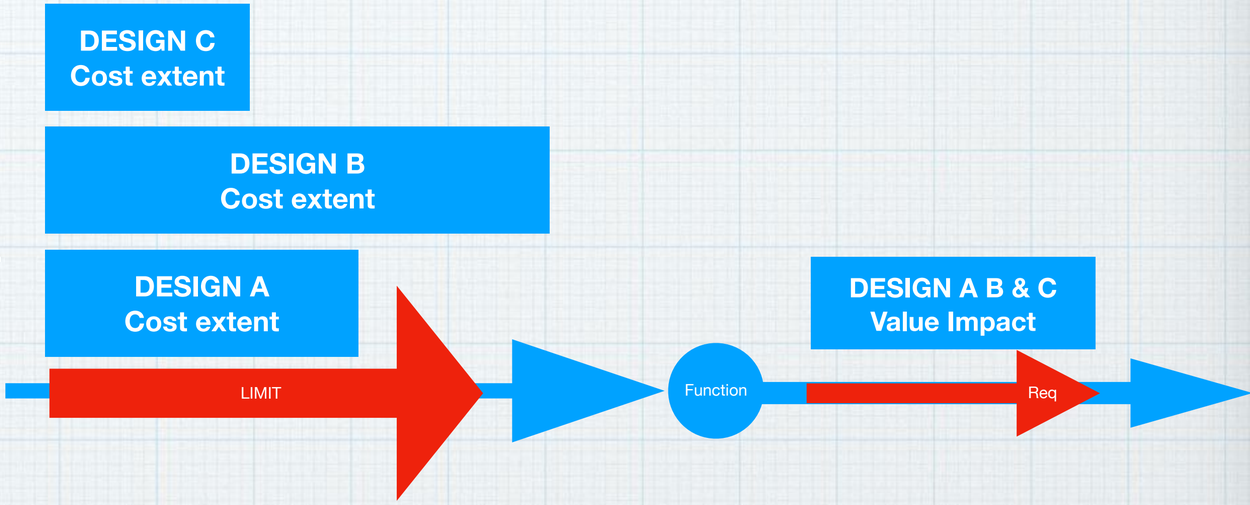
2. Vague ideas have a very broad 'cost range'.

3. Do we know any resource information (time, people, money) at all about any previous

   uses of our design options, by anyone, anywhere?

4. Can we get a sub-supplier to give us a fixed price, fixed-delivery-time contract, for

   the design options?



Cost-effective design
Efficient Design

These questions will help you point to a likely, cost-effective ('efficient design') candidate.

In some situations, that might be enough to go ahead and try the promising-designs out.

In other situations you would be gambling, too much of someone else's money and lives; so you might like some even-more-advanced design-methods for those cases.

Making sure that the design does not violate any other specified constraints.

* It is not enough to have a cost-effective design.

* It would be nice if it were 'legal' too!

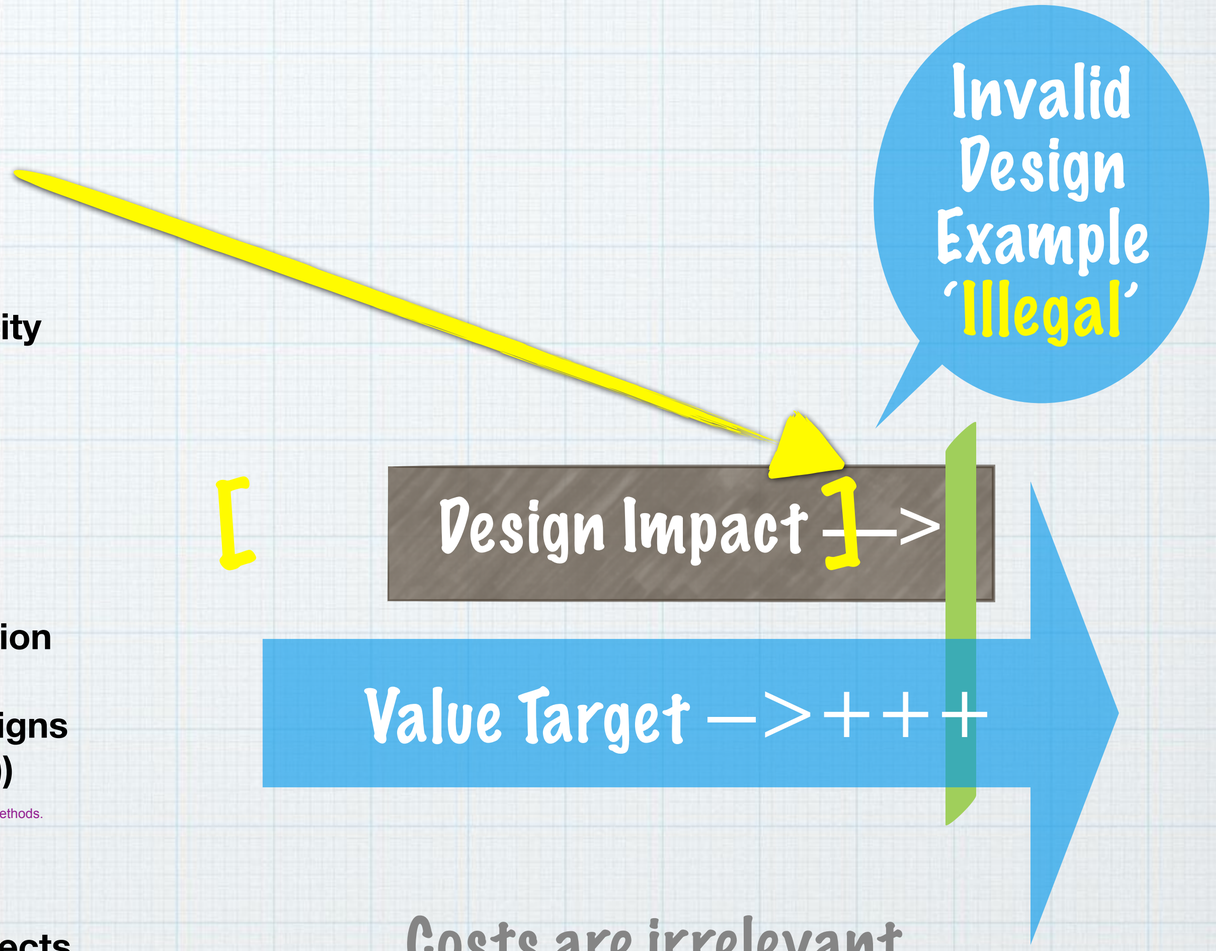
* There are a number of other considerations, we call them '[design constraints]', that your design needs to satisfy.

* You can ignore them, and no one will bother you for a while.
  * But recent Facebook and Google negative (EU, Congress) publicity
  * is a reminder of the price of arrogance.

* Design Constraints need to be formally specified, integrated into the overall rquirements, and acknowledged, as valid stakeholder needs,
  * Have we identified and documented all critical constraints?
  * Are the designers/Architects aware that we need to respect constraints 100%, or if not, then we are in legal/financial/reputation trouble?
  * Is there any rigorous review, or quality control, showing that designs actually respect the list of all constrants? (like Spec QC process))

AGILE INSPECTIONS: Reviews by sampling and measuring defects.
Extreme inspection and reviews based on objective and quantitative review methods.
http://www.gilb.com/dl239

* *Stakeholder analysis* will give you access to many design constraints.
  * Hard experience reminds you of others.
  * Did your organization learn anything specific, from previous projects that had problems?
  * Is this experience embedded in reusable checklists of design constraints that apply in most of your projects?
  * Try to lay your hands on that checklist!



Costs are irrelevant,
when a design exceeds other constraints,
such as legal, national, policy.

DESIGN CONSTRAINT CHECKLIST

* Take another look at your 50+ stakeholder list.
  * Do any stakeholders have any potentially critical design constraints?
    * Asking stakeholders what they want, is not the same as asking what they will not tolerate (constraints)
  * Would all stakeholders accept <u>any</u> design
    * (even if it killed Whales, with plastic bags?)
* Do not limit yourself to what the stakeholders would demand up front.
  * What would they say if later confronted with that fact that 'you chose plastic over wood' for xample?
  * Would you prefer to 'surprise them' later?
* CONSTRAINT EXAMPLES
  * Contract conditions
  * Laws: council, national, international
  * Regulations: health, finance, transport safety, building, employment
  * Organizational Policy, Partner Policy
  * Local culture, environmentalist, equality, consensus, anti-corruption
  * Public Opinion, media, respect

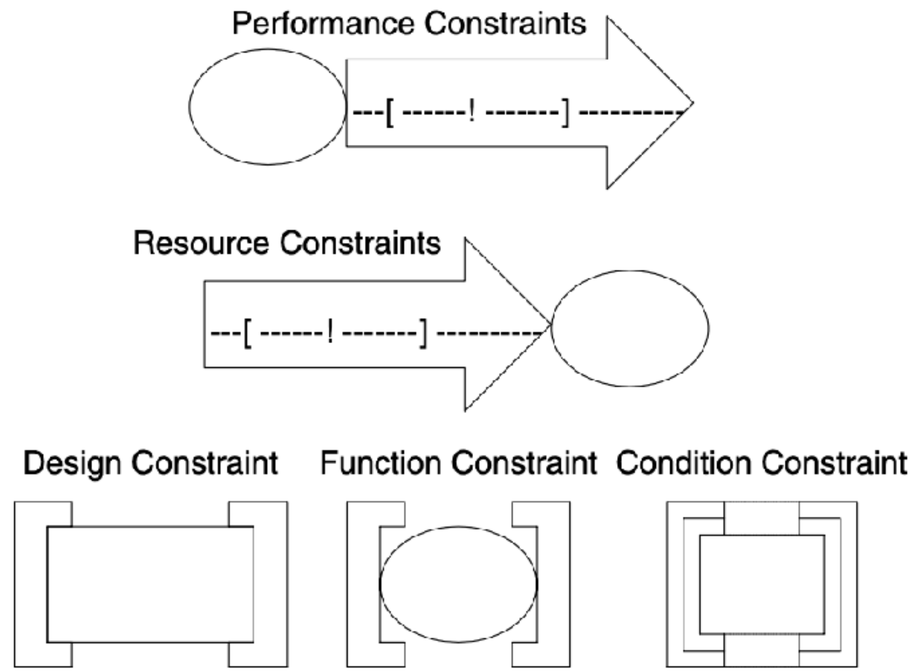


Figure G4
Drawn Icons for Constraints.

Competitive Engineering (book), glossary page 343

DESIGN CONSTRAINT Examples

* Performance constraint:
  * Tolerable 50%
* Resource constraints:
  * Budget £30M
  * Tolerable £40M
* Design Constraint:
  * Constraint [Data Storage]: National cloud only
* Function Constraint:
  * Function [Access]: Password and Secret Daily-Code.
* Condition Constraint:
  * Goal 50% For People [Over 80 Years], Living [At Own Home], Pre-existing Conditions [None Known].

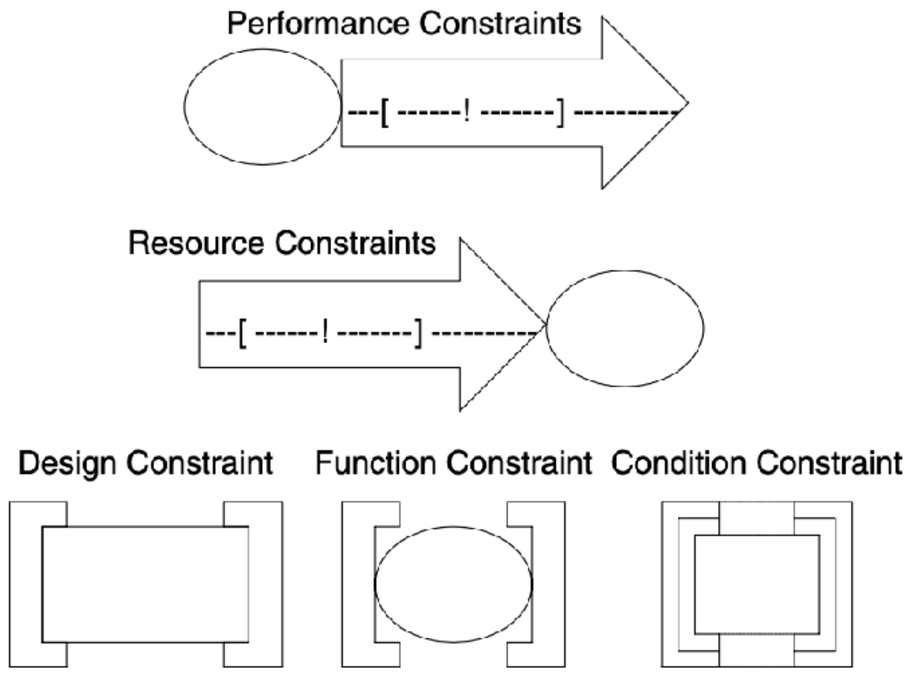


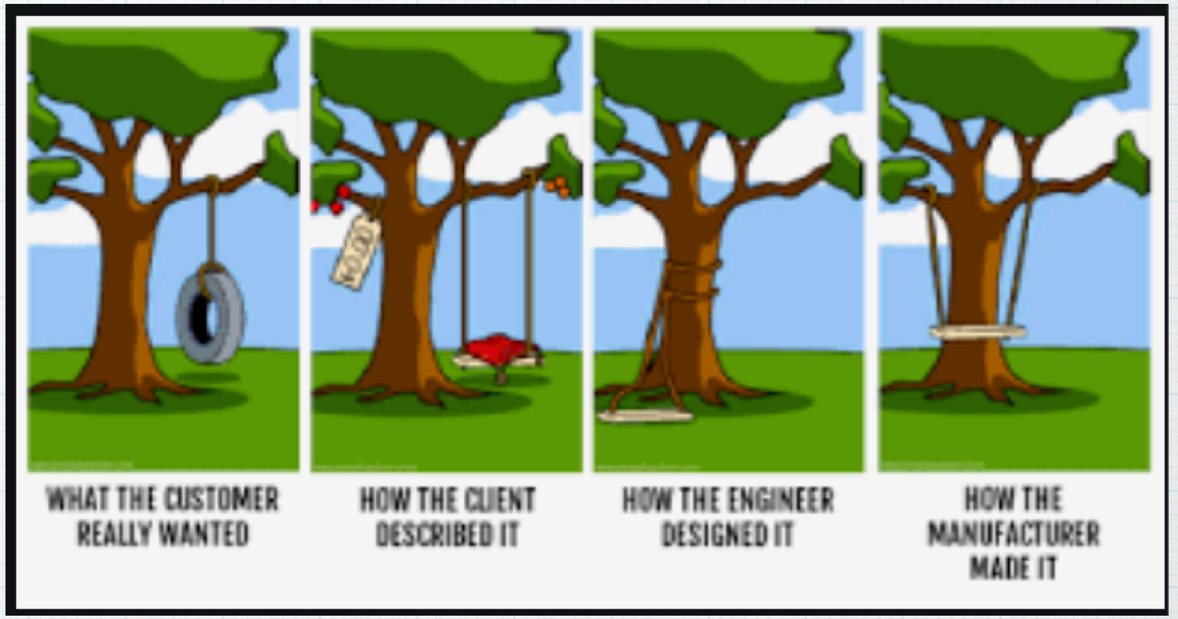
Figure G4
Drawn Icons for Constraints.

Competitive Engineering (book), glossary page 343

Part 2.

How not to evaluate a design

- Everybody is using that design
- The design is effective for X-ility
- The Powers That Be have asked for a particular design.
- We always use that design
- We do not know of any other design
- The supplier recommends this design
- Our architect has specified this design
- It is the cheapest option
- Other designs will take too long to implement
- This design is necessary because of another design



‘Everybody’ is using that design

* Popularity is not a good reason for you to use a design.
* Your only valid reason is that the design will credibly deliver the value required.
* The popular design might be good enough for other people’s value levels.

* But maybe *your* value requirements are different, and *your* design needs to be different.

* Maybe they have all chosen a bad design,
  * because it was well marketed,
  * and they are getting bad results;
  * or getting results that they do no realize are so bad,
    * since they have no clear numeric idea of the level they need, or the level they got.

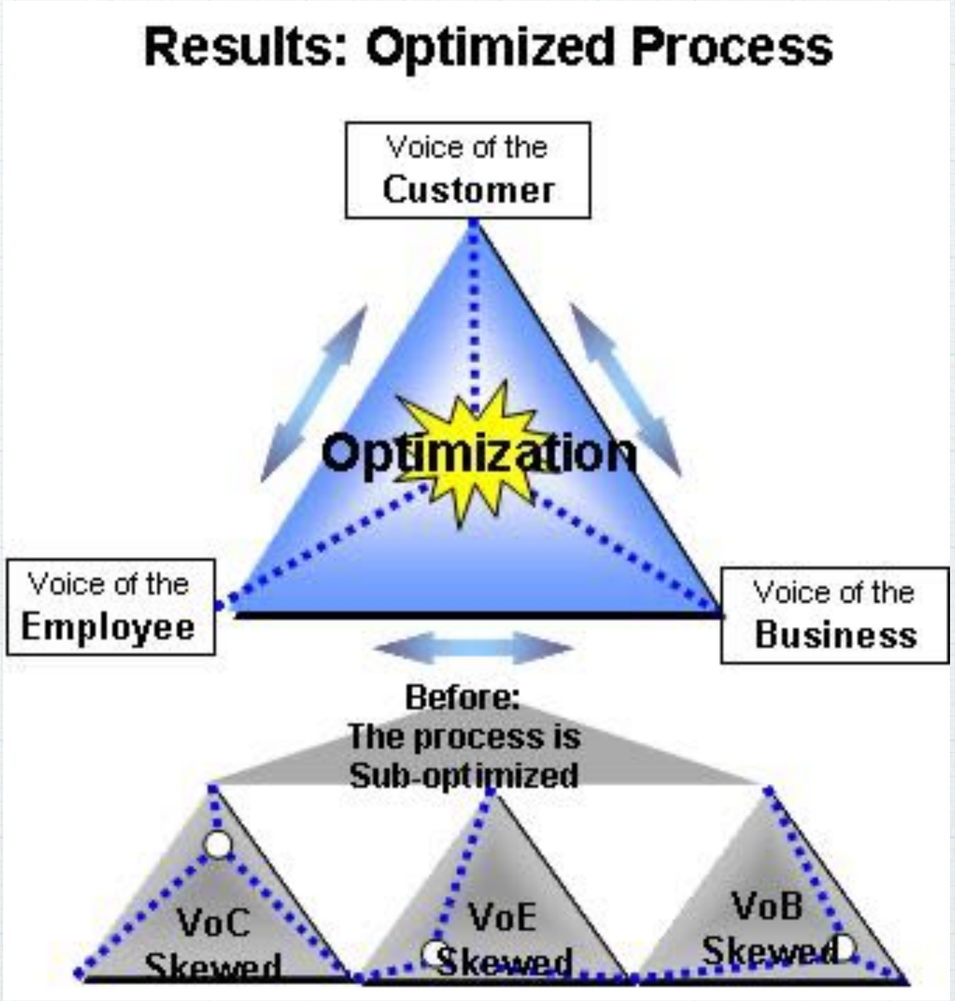
'The design is effective for X-ility' (a quality)

* The design is the best for 'security', or some value of interest.

* This might, or might not be true.
  * Is there serious evidence to prove that?
  * Or is it just a subjective opinion,
    * by someone who does not even know what 'serious evidence' means?

* But even if it is, the best, there are other considerations before we make a design commitment.
* What negative side-effects does it have on other critical values, like Usability?
* What does it cost, now and later, in money and time?
  * Does it steal all the budget?
* Does it violate any of our constraints?
  * Is it legal in all our markets?

* In other words, such an argument ('best') is insufficient to lead to adoption of the design.



http://lsspmp.com/sixsigma.html

‘The Powers That Be have asked for a particular design.’

* The ‘Power’ might be a big potential customer, your boss, your biggest client, the contract, even the ‘law’.

* However, that does not automatically mean you blindly and unquestioningly use that design.

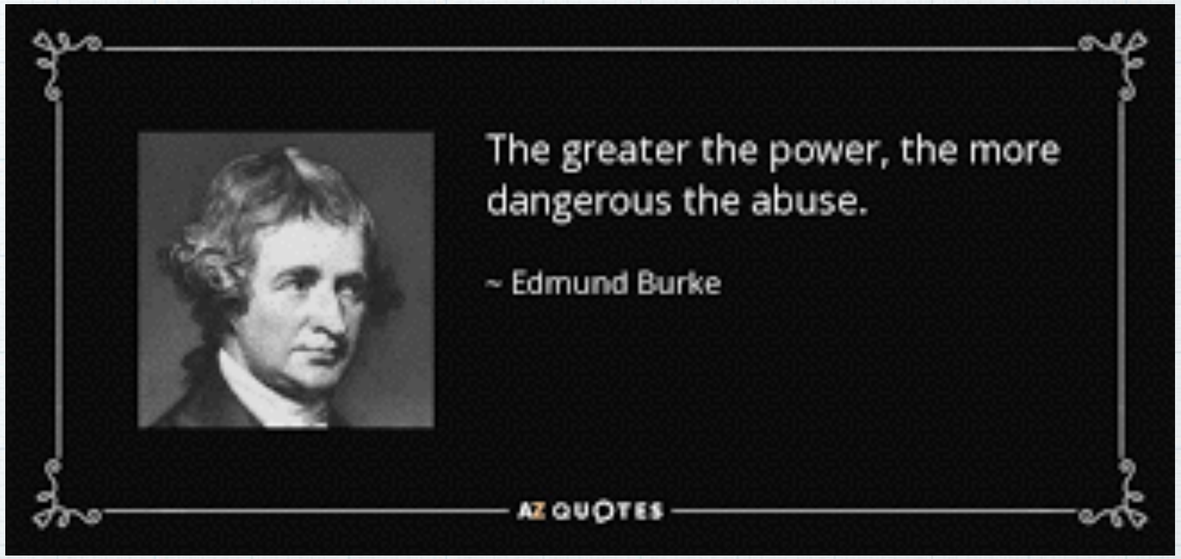
* Why not?
    * The design might be obsolete in many ways, replaced by better ones
    * The design might simply be their way of telling you what their values are, but not really intended as an unconditional demand from them to you.
    * The the design might have negative side effects that they are unaware of, and would not accept.
    * The design might cost far more sooner or later, than the Powers know about and would accept.
    * The Power behind the Power might have changed: a new Chief Technical Officer in power, might like to be consulted, on their ineffective predecessors’ bad decisions.

* We can treat the design request with respect. But we need, to be ‘design responsible’,
    * to be the professional designers they employ; to shed light on it,
    * both for the Power,
    * and for our own reputation and responsibility’s sake.

* Consequently we need to do the following Design Request analysis.

* Are there any clearly superior alternatives now available, which might have been overlooked or have recently emerged?
* What are the measurable expected side effects of this design request on any and all value requirements?
* What is the estimated consumption of our budgeted resources (time, money, maintenance costs, conversion costs, training, ...)
* Are there, for the design request, any potential violations of legal or other constraints, in any potential area of deployment?

* A written analysis should be submitted to the Power, with a recommendation about what to do, and requesting their written answer. Put the monkey on their back, or the monkey is still on yours.



‘We always use that design’

- Well that is fairly safe.
- A conservative option.
- But the times they are a changing, and
  - a designer is responsible
  - to their client
  - to be aware of interesting new options,
  - and to at least point them out.

- A table comparison of the options
  - would be one way to explain the differences,
  - and to give the Powers a real and responsible choice.

Requirements	DATABASE DESIGN	Symbology	Language Script	COPYRIGHT REVIEW...	Ape Apple Usability	Sum
Usability Status: 0 → Wish: 95 % ... % of [Users] able to complete a [De... [Users = All, ...] 18th June 2022	???? ± 0 0 % of ... 0 ± 0 % 0 0 % (x 0.0) ????	65 ± 25 65 % of ... 68 ± 26 % 68 7 % (x 0.1) 88%	70 ± 25 70 % of ... 74 ± 26 % 142 7 % (x 0.1) 74%	-2 ± 2 -2 % of ... -2 ± 2 % 140 -1 % (x 0.3) -2%	65 ± 25 65 % of ... 68 ± 26 % 208 7 % (x 0.1) 68%	✔ Σ±%: 208 ± 80 %
Copyright Law Compliance Status: 0 → Wish: 100 %... % [National Copyright Legislation]... [National Copyright Legislation = 16th July 2019	15 ± 5 15 % [N... 15 ± 5 % 15 11 % (x 0.7) 15%	22 ± 0 22 % [N... 22 ± 0 % 37 0 % (x 0.0) 22%	2 ± 0 2 % [N... 2 ± 0 % 39 0 % (x 0.0) 2%	95 ± 20 95 % [N... 95 ± 20 % 134 29 % (x 0.3) 95%	-30 ± 10 -30 % [N... -30 ± 10 % 104 -3 % (x 0.1) -30%	⚠ Σ±%: 104 ± 35 %
Technical Debt Management Status: 0 → Wish: 50 Min... % [Cost] spent on [Technical Mainten... [Cost = Total Technical Budge...] 22nd June 2018	100 ± 15 100 Minima... 200 ± 30 % 200 120 % (x 0.6) 200%	60 ± 10 60 Minima... 120 ± 20 % 320 96 % (x 0.8) 120%	10 ± 1 10 Minima... 20 ± 2 % 340 20 % (x 1.0) 20%	0 ± 20 0 Minima... 0 ± 40 % 340 0 % (x 0.2) 0%	55 ± 10 55 Minima... 110 ± 20 % 450 0 % (x 0.0) 110%	✔ Σ±%: 450 ± 112 %
Database Security Status: 0 → Wish: 95 Hac... % of [Unauthroised Access] to [Syst... [Unauthroised Access = All 1st June 2022	90 ± 50 90 Hackers 95 ± 53 % 95 19 % (x 0.2) 95%	40 ± 0 40 Hackers 42 ± 0 % 137 0 % (x 0.0) 42%	45 ± 20 45 Hackers 47 ± 21 % 184 14 % (x 0.3) 47%	10 ± 0 10 Hackers 11 ± 0 % 195 4 % (x 0.4) 11%	65 ± 10 65 Hackers 68 ± 11 % 263 0 % (x 0.0) 68%	✔ Σ±%: 263 ± 85 %
Sum Of Values: Σ%:	310 ± 88 % 310	252 ± 46 % 562	143 ± 49 % 705	104 ± 62 % 809	216 ± 67 % 1025	
Worst Case: Σ±%:	222 %	206 %	94 %	42 %	149 %	
Credibility - adjusted: Σ?%:	150 %	103 %	42 %	32 %	4 %	
Worst Case Cred. - adjusted: Σ±?%:	117 %	84 %	31 %	18 %	0 %	

A VALUE DECISION TABLE. DESIGNS ARE TOP COLUMNS (‘DATABASE DESIGN’) AND THEY ARE RATED AGAINST 4 VALUES, AS TO % ABILITY TO MEET THE VALUE WISH ON TIME. WHICH DESIGN DO YOU THINK IS THE BEST OVERALL OPTION ? LOOK AT THE ‘SUM OF VALUES’ NUMBER. BUT CONSIDER UNKNOWNS (‘????’) <- TECHNOSCOPES.

Semantic Analyzer Of Internet News - Phone App

<- Design Tag

Type: Solution Idea, Labels: - Edit

Is Part Of: Bullshit Indicator

Summary:

Description: by **gilbguest4** - Nov 27th 2018, 14:33 0

1. **Automated** analysis of the text marked via phone app
2. text scanned is saved in the cloud and is **Public**
2. Uses **Key Words** and **Semantic Analysis**
3. Accesible in the phone app

> **Automated:** defined as:
> Done by application independently
>
> **Key Words:** defined as:
> List of key words: lie, blame, false, true, truth, good, bad ...
>
> **Public:** defined as:
> **Free** to use by any **Users** and application
>
> **Semantic Analysis:** defined as:
> Analysis done on the basis of the soltuion presented in the paper http://jehsjd.efef.sefc

Source:

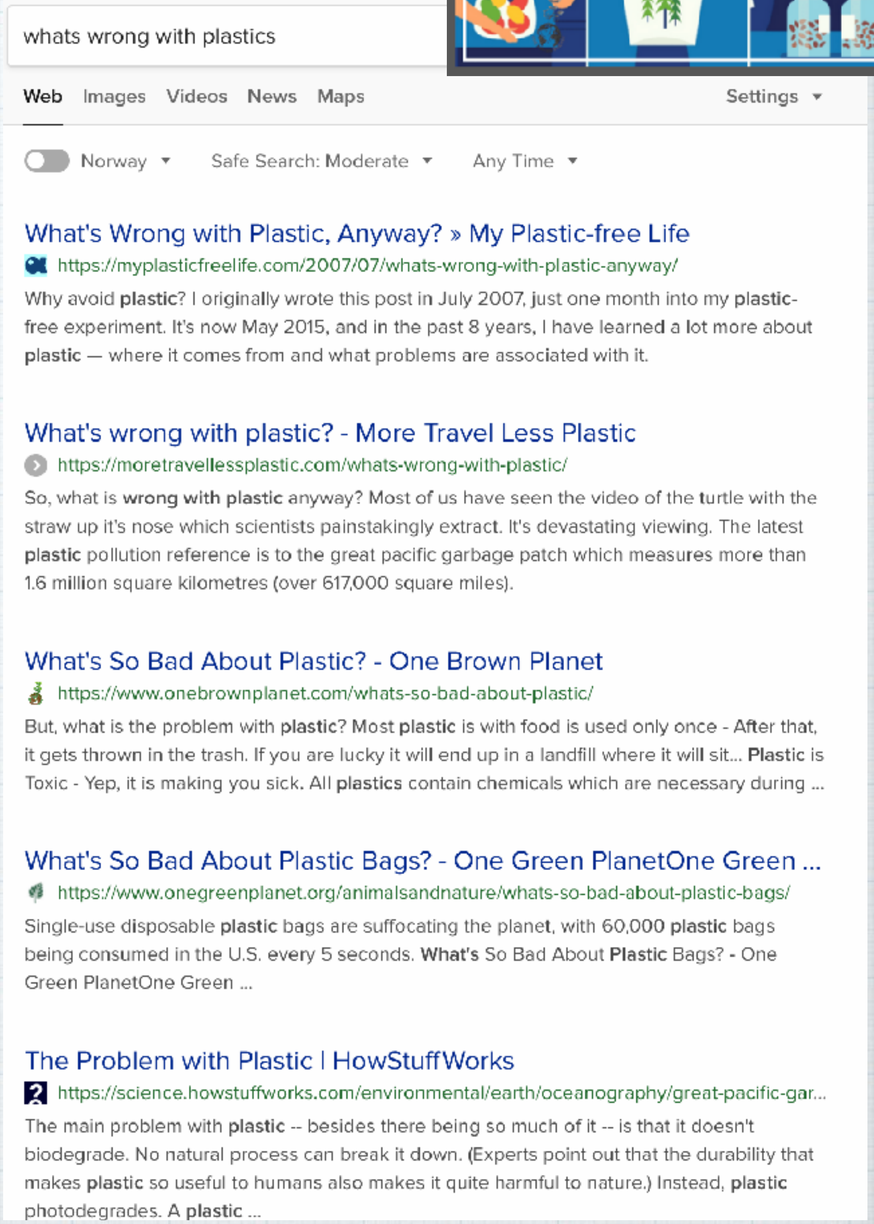
Marta and Daniel

Implementation Instance: Main Outside supplier Comarch. Contract framework signed,

Design Specification Example

‘We do not know of any other design’

- There is, you should safely assume, always some better design out there.
- If you cannot think of near alternatives even, then you can also assume your design competence is low.
- To avoid being found out as incompetent, you need to find someone who has better ideas.
- The simplest and cheapest way to do that initially is an internet search.
- Look for criticisms of the ‘one design’ you have, which will probably contain suggestions for better designs.
- Try the search key words: “what’s wrong with xxxxx’



'The supplier recommends this design'

- Sometimes suppliers have really valid reasons for recommending certain products or services.
  - In which case we should be able to ask questions and get good answers:

- Exactly why are you recommending this design to us?
- Are there better alternatives, if we pay more, from you or others?
- Can you give us some references from people who have used your recommendation
- Are there any weaknesses with your recommendation that you do not think we care about ?
- What would you recommend if we said, we do not care about the initial cost as long as we get lifetime good quality from it?
- Are you willing to give a total refund guarantee if we are unhappy with the design?
- How many years have you been using that design?
- Have you recommended this because we asked for a low bidder fixed price?

- You and I know that a supplier, might have unspoken reasons for recommending a design.
  - And for not telling us about better alternatives.
  - Forgive them for that.
  - You might act the same in their shoes.

- But there are many ways of finding out if there are some designs you might actually like better, even at a price, or maybe even a lower price.

12 Tough questions questions paper
http://www.gilb.com/dl24

'Our architect has specified this design'

* There are many kinds of 'architects'.
  * Some have proper qualifications and ethics.
  * Some do not.

* The architects are a type of supplier, and deserve the same analysis as for suppliers above.

* In a business I wander in, high tech, and IT, there are some people who went on a 4 day Enterprise Architecture course, and title themselves Enterprise Architects.
  * Needless to say they cannot be compared with proper architects with years of formal education, and practice in architecture firms.

* They have no training and discipline in quantified values and costs at all.
  * They talk about it in print and ignore it in practice.
  * The opposite of this book/Talk/Course where we get quite serious about values and costs in practice, and measurably.
  * Not with nice-sounding platitudes.

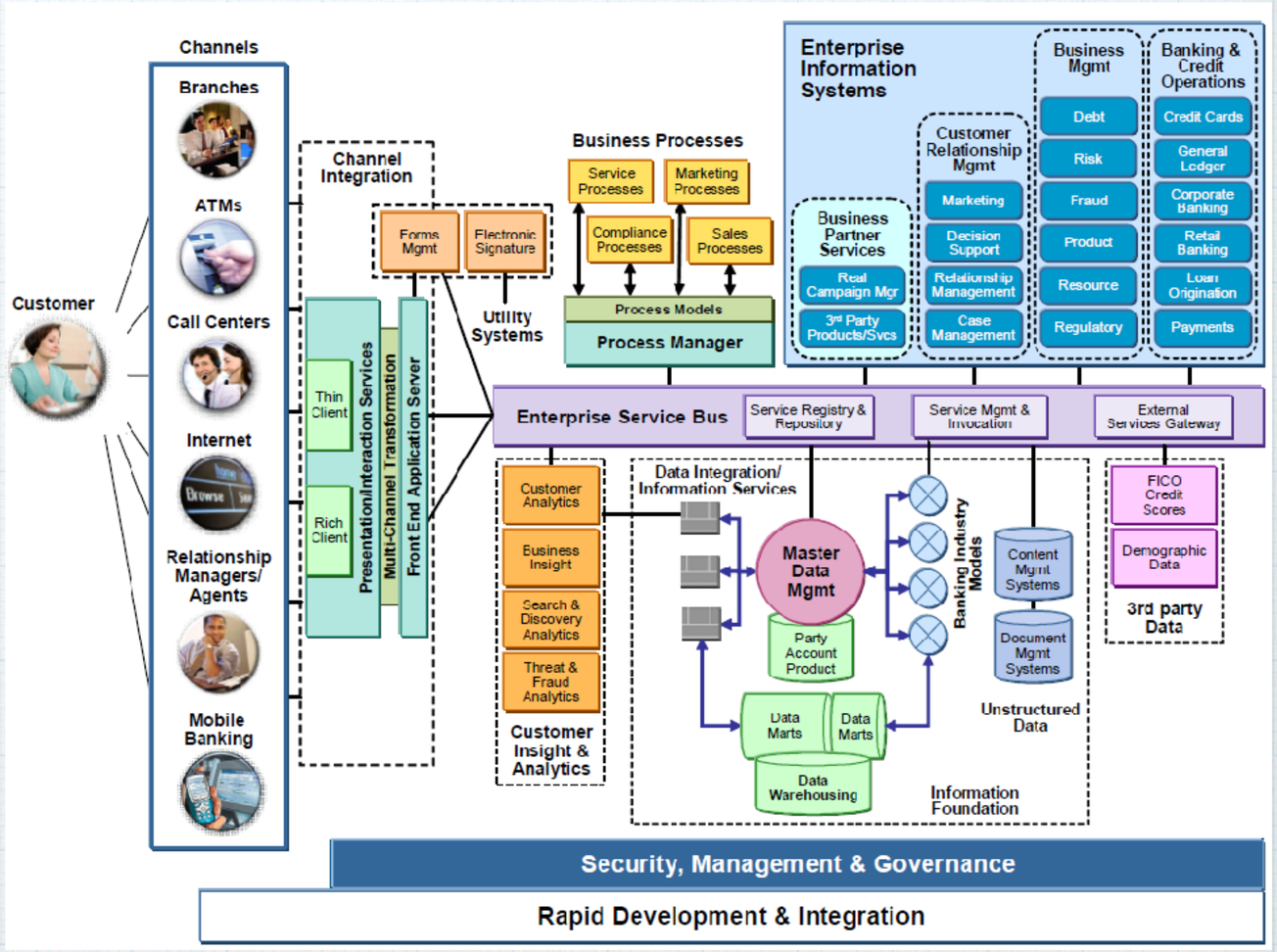
* But I have seen with my own eyes that they recommend extremely expensive designs, in the 1 million to 1 billion category (who cares currency),
  * and these design options are part of some very expensive and public failures:
  * for which the 'architect' takes no blame, or legal or financial responsibility.

* This makes me very angry, as a taxpayer, and sometimes angry as a services (health, tele, finance) consumer.

* I have concluded that I should not blame the pseudo-architects.
  * I should blame the management who employs and listens to them.
  * Who do not ask enough hard questions about their decisions.
  * The MBAs, Mean to Blame the Architects gang.

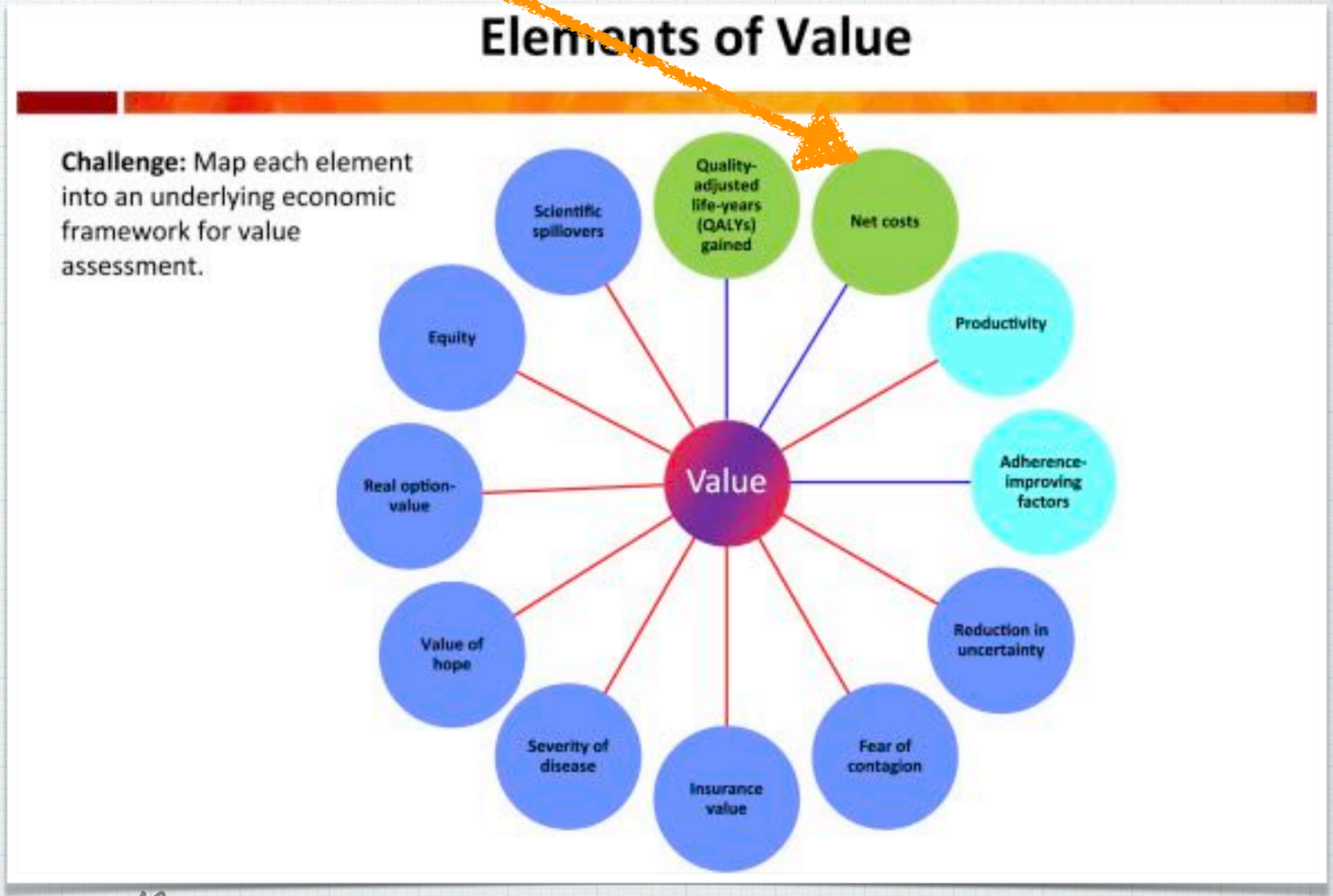
* Get yourself a proper Architecture Engineer! Not a buzzword 'architect'.

* What is that? This book.



'It is the cheapest option'

- Well, cheap is good,
  - all other factors being the same.
- But we need to analyze
  - that all other value and cost factors
  - that we consider critical,
  - are really 'the same'.
- Assume *not*, until proven otherwise,
  - might be a safe way
  - to make decisions.



'Other designs will take too long to implement'

* Maybe, maybe not.
  * If this is true, and it is critical
  * then that might be a good reason.

* but surely you are not so innocent that you simply believe such a claim as an oral meeting answer?

* How do you know how long it will really take to implement in practice?

* What guarantees do you have in place for that?
  * "Can we have it totally free if you are delayed more than 1 week for any reason?" No?? Let's talk about why?"

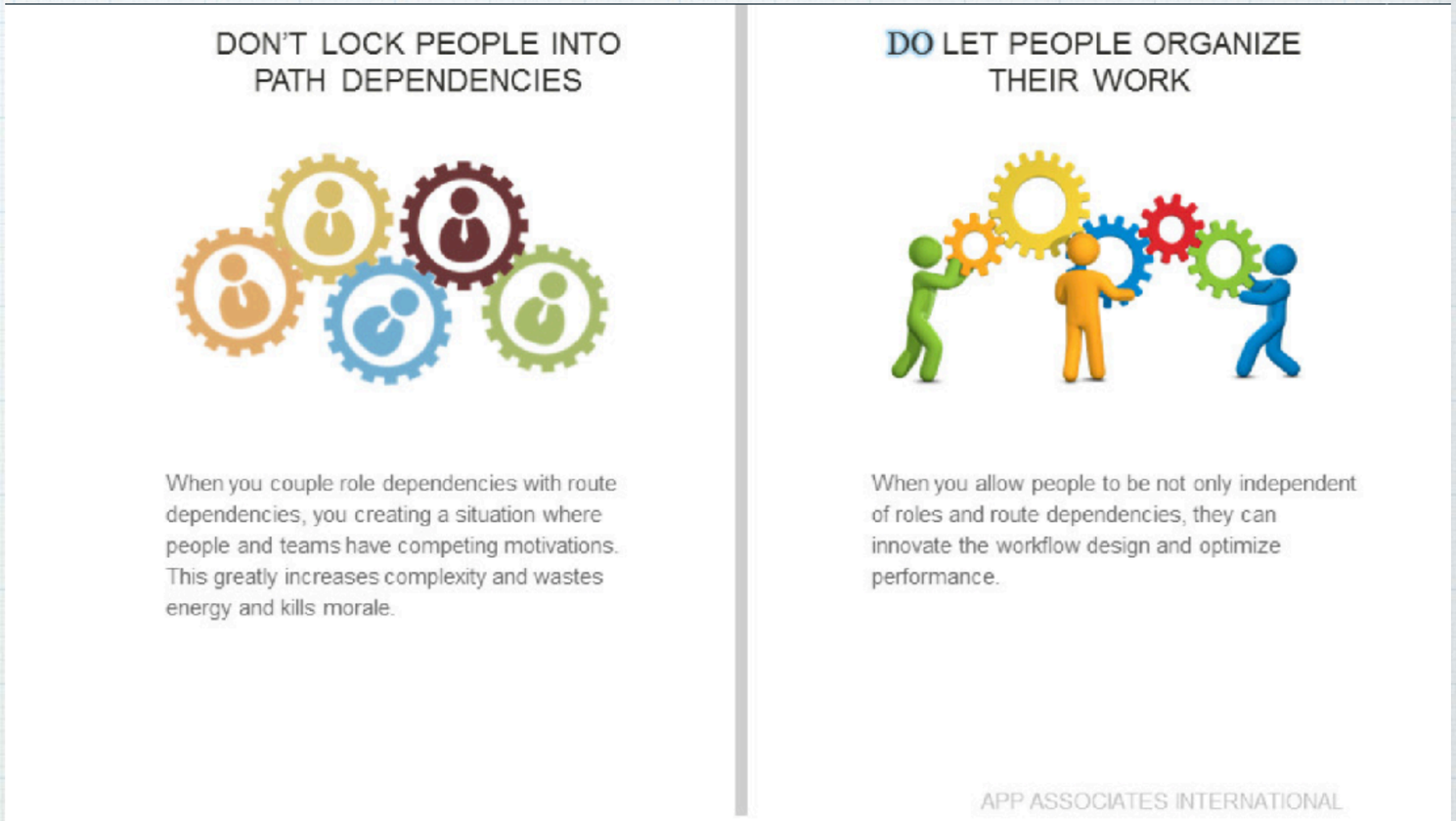
* And those other designs, maybe from their competitors,
  * have you checked with them about *their* guarantees,
  * and experiences with delivery times?
  * Is there an Express Delivery option, at a price?

* It may be worth it to dig deeper into the delivery time question.

* It is rarely simple, and things are far too often delayed, with *any* design.

‘This design is necessary because of another design’

- Dependencies.
- Maybe that ‘other design’ can be replaced with a better design
- which does not have such dependencies attached to them.



See decomposition based on independence of implementation
Value Planning book, Ch 5 Decomposition by Value” in my Dropbox:
https://tinyurl.com/VPDecomposition or VP book gilb.com

Part 3.

Simple ways to evaluate and compare designs

An Impact Opinion Table (IOT)

* An Impact Opinion Table (IOT):
    * looking at ideas from several value points of view

* Before we do go into more detail
    * (with the best and most advanced design methods),
    * let's look at some potentially useful ways of evaluating, and prioritizing, any design.
    * At least for simple problems.

* This is an Impact Table.
    * In fact let me be blunt and call it a ***Subjective*** 'Impact ***Opinion*** Table'.

* It is a simplified way to look at our Value Requirements and our Designs for attaining them.

* I will comment on it in more detail just below.

Designs -> Value Requirements v	Diet	Invest-ment	Sharing	Startup	Continu-ous Learning
Health	??	?	?	?	?
Wealth	?	??	?	?	?
Happiness	?	?	??	?	?
Career	?	?	?	??	?
Education	?	?	?	?	??

* FIGURE:
* A BASIC VALUES VERSUS DESIGNS
* TABLE STRUCTURE,
* INITIALLY WITH UNKNOWNS

Multidimensional Design Evaluation

* **Multi-dimensional Design Evaluation**
* Let me declare at this point, that evaluating one design, on one value-dimension, is *never* going to be a safe evaluation.

* We always have two concurrent needs:
    * Understanding side-effects on all other values and costs, of any design
    * Comparing one design with *other* designs
        * *Complementary* designs, to see which should be used first
        * *Competitive* designs, to pick a winner

* For both purposes, we need a *table* structure. Values versus designs.

* We need to look at any set of designs, that interest us.

* We need to evaluate all interesting designs,
    * with respect to any set of values and resources that interest us.

* That multi-dimensional design-evaluation process, will give us a more-complete view of the design situation, so that we can:
    * select the best mutually-exclusive architectures and designs.
    * prioritize delivery of design components,
    * so we can build a value-stream early, continuously
        * and with good cost-effectiveness.
*

Designs -> Value Requirements v	Diet	Invest-ment	Sharing	Startup	Continu-ous Learning
Health	??	?	?	?	?
Wealth	?	??	?	?	?
Happiness	?	?	??	?	?
Career	?	?	?	??	?
Education	?	?	?	?	??

Knowing what you do not know

*Here is an explanation of the ‘starting point’ (all unknowns) for the Table above.

*I inserted a ?? at the intersection between a Requirement and the Main Design for each Requirement.

*Then I inserted a simple '?' for all the *other potential* impacts, which that Main Design *might* cause, like it or not; ‘?’ for all the *other* Requirements: the '*Side* Effects'.

*This ?-filled table is our 'initial state of ignorance', *until* we begin to estimate the potential impacts of our ideas on each value objective.

*'??' means we do not understand anything about the main effect of our Design on the main value, which it was *chosen* for 'good impact' on.
    *or, at least, we have not shared and documented our understanding yet
    *Nobody else knows what we pretend to know

*'?' means we do not yet pretend to know anything about the ***side-effects*** of that idea, on all our *other* values.
    *These side-effects could, theoretically, be so bad as to ruin our life or project!

Designs -> Value Requirements v	Diet	Invest-ment	Sharing	Startup	Continu-ous Learning
Health	??	?	?	?	?
Wealth	?	??	?	?	?
Happiness	?	?	??	?	?
Career	?	?	?	??	?
Education	?	?	?	?	??

* *Confucius said: "True wisdom is knowing what you don't know."*

*The Impact Opinion Table, with ??s, is a tool for helping you see what you do not know, yet. This ?? Cultures extends to more-advanced design evaluation tools. Discussed later. (VDT or IET).

Getting some rough idea of what we think we know about ideas: filling out the estimates without pushing your grey cells too much.

- Your Designs can have *positive* impacts (+), *negative* impacts (-) or *no* impacts (0).
- Your Designs can have *minor* impacts (+), *medium* impacts (++) or *strong* impacts (+++).
- But this rough method is not good enough for all large systems,
  - such as health service, transportation, communication, government operation, international products services and trade.
- Now, at this point (in methods presentation) I am just looking for a 'rough feeling' for 'which one' of the Designs is *significantly* better than the others.
  - That's the one you probably want to *start* 'value deliveries' with.

Designs -> Diet Requirements v		Investment	Sharing	Startup	Continuous Learning
Health	+++	-	+	--	+
Wealth	-	++	-	-	+
Happiness	-	--	+++	-	+
Career	?	+	++	++	++
Education	0	++	+	++	+++
Net Advantage	3-2 =1 & ?	5-3 =2	7-1 = 6	4-4=0	8

Weaknesses with this simple Impact Table

1. **Ambiguous Value Specs:** There is no guarantee of rigorous value definition with a scale of measure. The detail might be nice sounding words ('We need to get a competitive level of market flexibility!').
   This implies that nobody can be sure of what the value requirement level actually is, and all evaluations are ambiguous. Worthless.

2. **Ambiguous Design Specs:** There is no guarantee that this design name points to a sufficiently clear and detailed design specification. Sometimes the name is all you have got, which is too common.
   This means that the 'impact evaluator person' will be at liberty to interpret the design specification in any different ways. And they will not tell you which misinterpretation they decided on.

3. **Very Rough +++ evaluations:** The + ++ and +++ are rough impressions, but what exactly do they mean ?
   Does +++ mean you will meet your success targets? Does ++ mean you will deliver above the worst acceptable case? Or is that the + ?
Is there any significant

4. **Evidence Please**? On what basis did the estimator decide on their +++ ? Were there some known experiences, or facts, as the basis? Can we be let in on this interesting secret? What is the source or these facts, and is is a credible source at all, or is it just SWAG (Scientific Wild-Assed Guess)?

5. **Responsibility:** if the evaluations are wrong, but we act on them in a large project. Who is to blame? Will these consultants give us our money and honor back? Are they even named here?

Designs -> Diet Requirements v		Invest-ment	Sharing	Startup	Contin-uous Learning
Health	+++	-	+	--	+
Wealth	-	++	-	-	+
Happi-ness	-	--	+++	-	+
Career	?	+	++	++	++
Education	0	++	+	++	+++
Net Advantage	3-2 =1 & ?	5-3 =2	7-1 = 6	4-4=0	8

The Impact Table 'over-simplifications' include:

- No 'explicit' definitions,
  - let alone 'quantified' definitions of your *objectives*.
  - You *think* you know what you are thinking about.
  - I'd be careful about that assumption!
- No detailed definition of the design ideas, or strategies,
  - you know *roughly* what you mean by them.
  - It is surprising what happens, when you actually define them in writing!
- No *exact* numbers,
  - to rate the expected impacts, just simple + ++ +++ etc.
- No necessity to back up your subjective ratings with *evidence*.
  - It's just your *gut feel*.
- No systematic look at *cost aspects*, or consequent calculation of *value for resources*.
- No information about any critical stakeholder constraints (laws, contracts, policies)

- If you are looking for deeper *truths*, good *risk* analysis, and *clear* communication: then *that* list above amounts to *too many weaknesses*.

- But there are always simpler, smaller, local, individual problems that *can benefit* from a simpler tool.

- We do not always need to use an electron microscope, when a magnifying glass will satisfy our needs.
- Binoculars see the starry night amazingly well,

Designs -> Requirements v	Diet	Investment	Sharing	Startup	Continuous Learning
Health	+++	-	+	--	+
Wealth	-	++	-	-	+
Happiness	-	--	+++	-	+
Career	?	+	++	++	++
Education	0	++	+	++	+++
Net Advantage	3-2 =1 & ?	5-3 =2	7-1 = 6	4-4=0	8

So, here are the positive (++) aspects of the Impact Table:

- It gives us a look at *multiple objectives (ends)* simultaneously (avoiding over-focus on *single* objectives).
  - This encourages us to look at side-effects, and the larger picture.
- It clearly considers a *set* of complimentary, or optional, Design Ideas (means), with some ability to see their differences.
- It registers someone's subjective opinion (the + 0 -), about the relationships,
  - and sometimes they can explain their judgement in practical, or experiential, terms if asked.
  - what if 2 or more people independently built Impact Tables, and we compared their opinions?
- It can help us distinguish 'very promising' Design Idea options, from total 'losers'
- So the Impact Table can be used as a rough quick draft, preliminary to a possible better analysis of the designs.
- In simple terms, if your entire family's lives were at stake, would you like to use this simple tool to evaluate your options in the hostage situation?

Designs -> Requirements v	Diet	Invest-ment	Sharing	Startup	Contin-uous Learning
Health	+++	-	+	--	+
Wealth	-	++	-	-	+
Happi-ness	-	--	+++	-	+
Career	?	+	++	++	++
Education	0	++	+	++	+++
Net Advantage	3-2 =1 & ?	5-3 =2	7-1 = 6	4-4=0	8

Part 4.
More-powerful ways to evaluate design options

Ways of Using the Impact Estimation Table aka Value Decision Table

1. Value Decision Making: analyzing how to get the Values you want at costs you can afford. We call this use Value Decision Table (VDT)

2. Presentation: presenting analysis, decision arguments in meetings, papers, books,

3. As a database for graphical presentation: the structures, and numeric data structured and defined, in tables and spreadsheets invite a variety of simplified graphical presentations.

4. For selection of winners, or top contenders, of mutually exclusive options (architectures, designs)

5. For determining the priority of delivery, of design subsets, for incremental value delivery

6. For documenting, and logically considering, risks of evaluation: best and worst case, source credibility.

7. For tracking incremental progress towards value targets: project management. Is the project going in the right direction, at the right rate of speed, at the right costs?

8. For managing dynamic design to cost: early and frequent (2% steps) proof-of-design cost-effectiveness, and adjustment immediately.

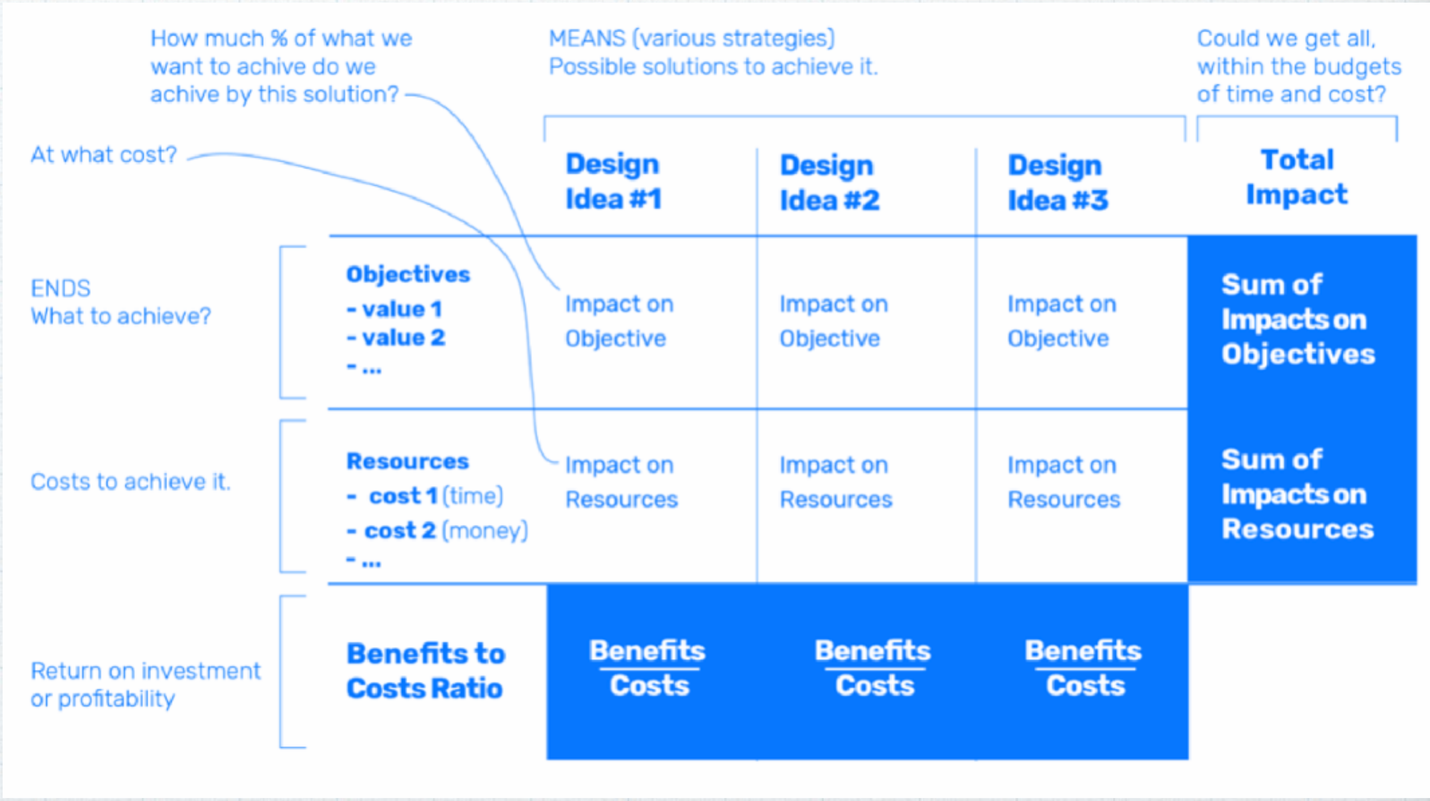
9. For modeling complex systems, especially values, costs and designs: at related hierarchical levels.

10. For *forcing* complete analysis of all values and costs, and recognizing incomplete design thinking

11. As documentation for decisions made earlier. Lawsuits, Contracting basis, organizational learning.

12. Relationship analysis: for seeing and presenting complex system relationships, up and down a hierarchy, and side-ways, at any given level.

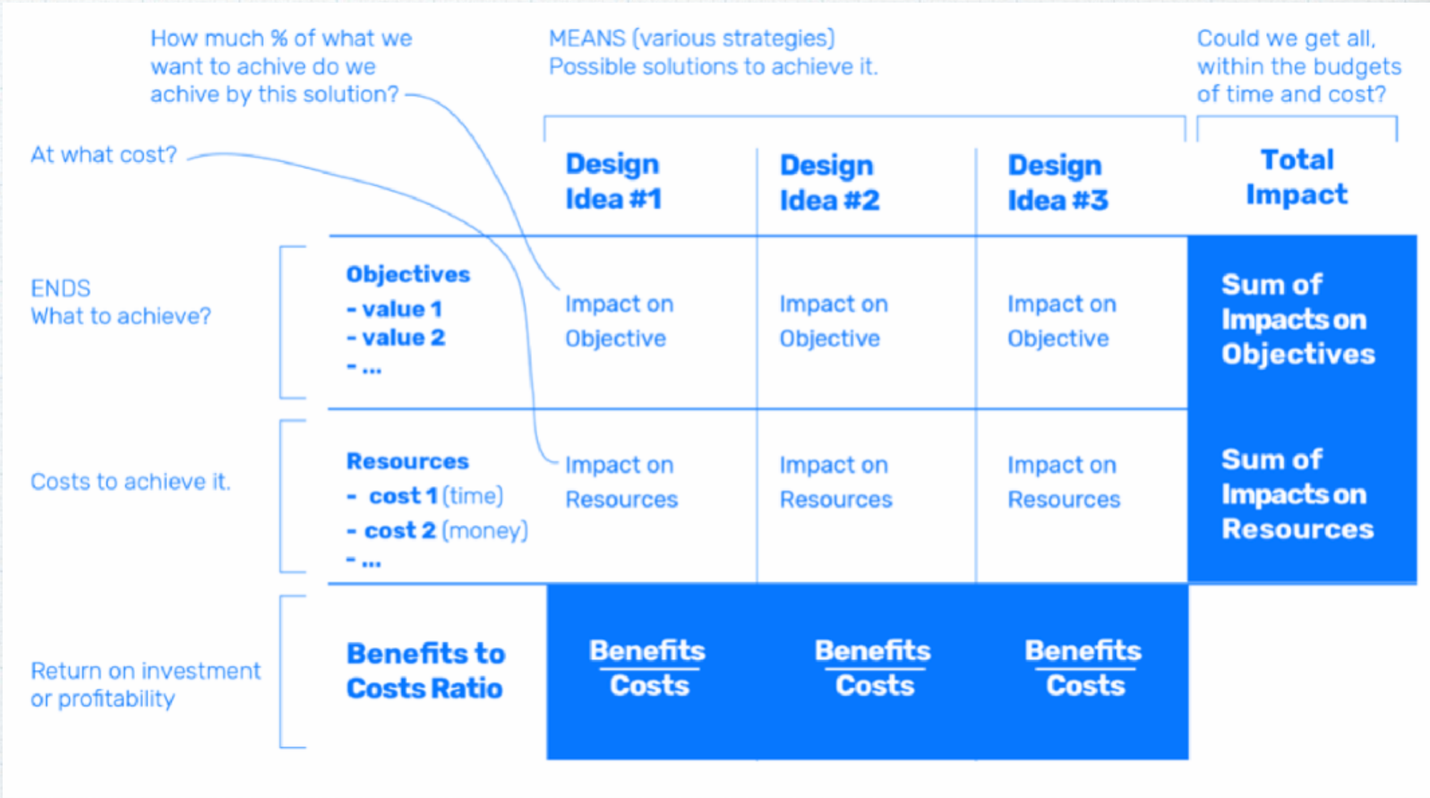
Impact Estimation (IE)
Design —> Requirements



The Impact Estimation Table aka Value Decision Table Structure

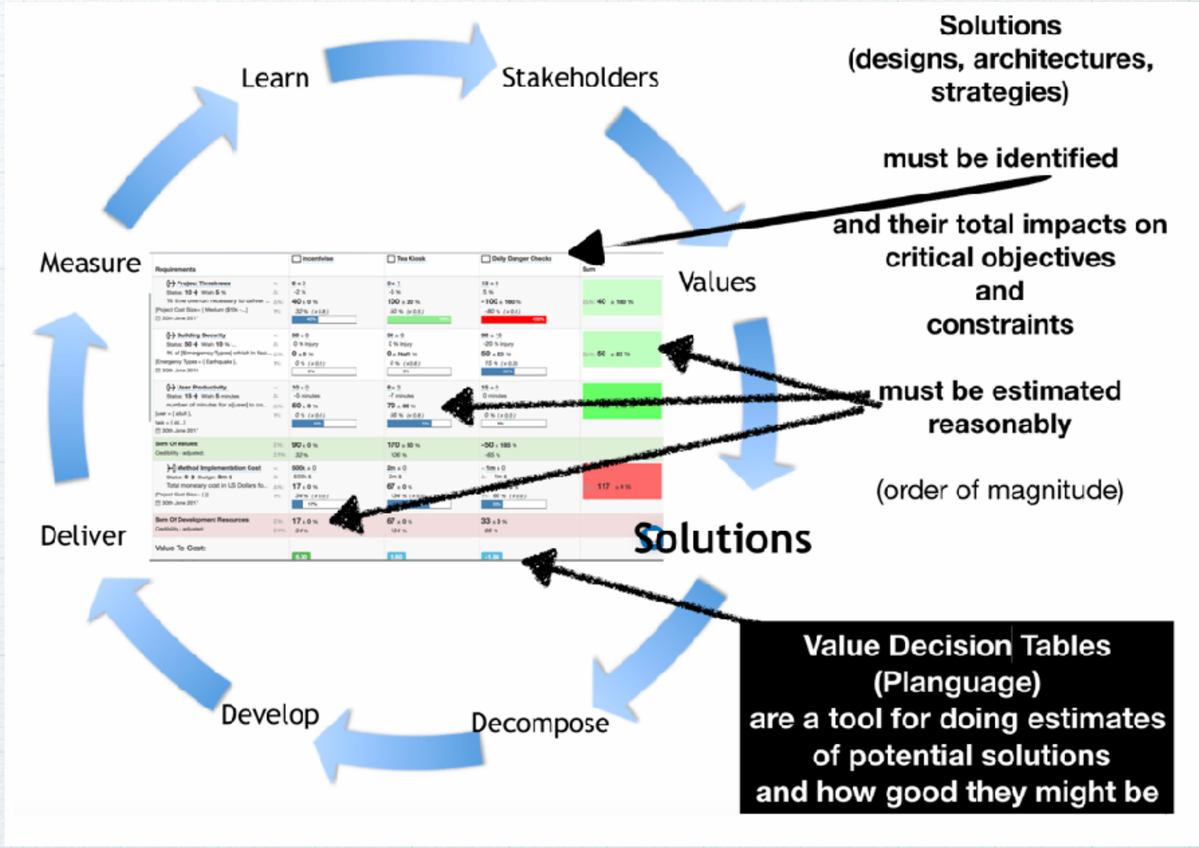
- Any number and type of designs can be considered.
- Any number of Value requirements can be considered
- Any number of resource requirements can be considered
- Normally a table is about 10x10
  - and if it gets larger, than a page or a screen,
  - we can decompose it into smaller detail tables,
  - or summary tables
- The Name Tags of Designs and Values and Costs on the table, *refer* to any 'quantity of detail' in the corresponding specification objects.
- The 'real Scale' numbers are recomputed to a % impact
  - (100% meets planned level on time)
  - which makes possible '% summaries',
  - which would be impossible, when Scales differ in nature.
- The table can use any kind of 'table software',
  - or be drawn.
  - But spreadsheets allow some automation and presentation advantages.
- Specialized Planguage tools like ValPlan.net permit more automation, presentation, and integrations than spreadsheets.

Clarity, Relations, Quantification , Overview



The IE Table tool ties several process steps together

* THE TABLE IS THE BASIC TOOL FOR DETERMINING IF A DESIGN HAS GOOD VALUES AND COSTS,
* AND IF A DESIGN FITS IN WITH THE OTHER DESIGNS,
* MEANING ‘THIS DESIGN MOVES US IN THE DIRECTION OF THE PLANNED VALUE LEVELS,
* WITHOUT USING TOO MUCH RESOURCE’



Real IE Table example: simple and small

HEALTHCARE SYSTEM IMPACT ESTIMATION

	Automate Rules	Web Self Service	Decision Support	Total Impacts
Increase Transmission of Requests *(30 minutes → 10 minutes)*	10 minutes 100%	3 minutes 100%	-	200%
Decrease Number of Errors Occurring *(353 per week → 30 per week)*	100 errors 80%	< 50 90%	-	170%
Decrease Time for Processing of Requests *(70 minutes → 10 minutes)*	35 minutes 70%	-	< 10 minutes 90%	160%
Decrease Time to Learn process *(1 day → 1 hour)*	-	1 hour 100%	10 minutes 103%	203%
TOTAL DESIGN REQUIREMENT IMPACT	250%	290%	193%	

SIMPLE REAL UK NHS VALUE DECISION TABLE

a real Health service, UK, table, successful project (source: http://www.gilb.com/dl582). Man-Chie Tse and Ravinder Ravi Singh Kahlon

* The right-hand column keeps track of the 'sum of all currently evaluated designs', for any one horizontal row of values or costs.
* In theory you need a total of 100% for your set of design ideas, to have a chance of meeting your requirements.
* But, like the good engineering practice of having a *safety factor*,
  * we usually aim for a sum of at least 200%,
  * twice the design we need in theory,
  * 'to be on the safe side'.
  * Color codes warn of 'too much cost' or 'too little value'.
* The estimates are necessarily rough.
  * There are dozens of known factors which influence the real outcome, and they are not included here.

Confirmit Example

IET, project step planning and accounting: using an Impact Estimation Table

- IET for MR Project – Confirmit 8.5
- **Solution:** Recoding
  - Make it possible to recode variable on the fly from Reportal.
  - Estimated effort: 4 days
  - Estimated Productivity Improvement: 20 minutes (50% way to Goal)
  - actual result 38 minutes (95% progress towards Goal)

http://www.gilb.com/DL32
Confirmit Case paper

	A	B	C	D	E	F	G	BX	BY	BZ	CA
1											
2		Current Status	Improvements		Goals			Step9			
3								Recoding			
4								Estimated impact		Actual impact	
5		Units	Units	%	Past	Tolerable	Goal	Units	%	Units	%
6					Usability.Replacability (feature count)						
7		1,00	1,0	50,0	2	1	0				
8					Usability.Speed.NewFeaturesImpact (%)						
9		5,00	5,0	100,0	10	15	5				
10		10,00	10,0	66,0	20	15	5				
11		40,00	0,0	0,0	40	30	10				
12					Usability.Intuitiveness (%)						
13		0,00	0,0	0,0	0	60	80				
14					Usability.Productivity (minutes)						
15		20,00	45,0	112,5	65	35	25	20,00	50,00	38,00	95,00
20					Development resources						
1			101,0	91,8	0		110	4,00	3,64	4,00	3

© 2008 Kai Gilb

EVO Plan Confirmit 8.5

4 more product areas were attacked concurrently

Impact Estimation Table: Reportal codename "Hyggen"

Current Status	Improvements		Reportal - E-SAT features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Intuitivness (%)		
75,0	25,0	62,5	50	75	90
			Usability.Consistency.Visual (Elements)		
14,0	14,0	100,0	0	11	14
			Usability.Consistency.Interaction (Components)		
15,0	15,0	107,1	0	11	14
			Usability.Productivity (minutes)		
5,0	75,0	96,2	80	5	2
5,0	45,0	95,7	50	5	1
			Usability.Flexibility.OfflineReport.ExportFormats		
3,0	2,0	66,7	1	3	4
			Usability.Robustness (errors)		
1,0	22,0	95,7	7	1	0
			Usability.Replacability (nr of features)		
4,0	5,0	100,0	8	5	3
			Usability.ResponseTime.ExportReport (minutes		
1,0	12,0	150,0	13	13	5
			Usability.ResponseTime.ViewReport (seconds)		
1,0	14,0	100,0	15	3	1
			Development resources		
203,0			0		191

Current Status	Improvements		Survey Engine .NET		
Units	Units	%	Past	Tolerable	Goal
			Backwards.Compatibility (%)		
83,0	48,0	80,0	40	85	95
0,0	67,0	100,0	67	0	0
			Generate.WI.Time (small/medium/large seconds)		
4,0	59,0	100,0	63	8	4
10,0	397,0	100,0	407	100	10
94,0	2290,0	103,9	2384	500	180
			Testability (%)		
10,0	10,0	13,3	0	100	100
			Usability.Speed (seconds/user rating 1-10)		
774,0	507,0	51,7	1281	600	300
5,0	3,0	60,0	2	5	7
			Runtime.ResourceUsage.Memory		
0,0	0,0	0,0		?	?
			Runtime.ResourceUsage.CPU		
3,0	35,0	97,2	38	3	2
			Runtime.ResourceUsage.MemoryLeak		
0,0	800,0	100,0	800	0	0
			Runtime.Concurrency (number of users)		
1350,0	1100,0	146,7	150	500	1000
			Development resources		
64,0			0		84

http://www.gilb.com/DL32
Confirmit Case paper

Current Status	Improvements		Reportal - MR Features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Replacability (feature count)		
1,0	1,0	50,0	14	13	12
			Usability.Productivity (minutes)		
20,0	45,0	112,5	65	35	25
			Usability.ClientAcceptance (features count)		
4,4	4,4	36,7	0	4	12
			Development resources		
101,0			0		86

Current Status	Improvements		XML Web Services		
Units	Units	%	Past	Tolerable	Goal
			TransferDefinition.Usability.Efficiency		
7,0	9,0	81,8	16	10	5
17,0	8,0	53,3	25	15	10
			TransferDefinition.Usability.Response		
943,0	-186,0	######	170	60	30
			TransferDefinition.Usability.Intuitiveness		
5,0	10,0	95,2	15	7,5	4,5
			Development resources		
2,0			0		48

Technical Debt Engineering – "green" week

Current Status		Improvement	Goals			Step 6 (week 14)		Step 7 (week 15)	
	Units		Past	Tolerable	Goal	Estimated Impact	Actual Impact	Estimated Impact	Actual Impact
	100,0	100,0	0	80	100			100	100
Speed									
	100,0	100,0	0	80	100	100	100		
Maintainability.Doc.Code									
	100,0	100,0	0	80	100	100	100		
InterviewerConsole									
NUnitTests									
	0,0	0,0	0	90	100				
PeerTests									
	100,0	100,0	0	90	100			100	100
FxCop									
	0,0	10,0	10	0	0				
TestDirectorTests									
	100,0	100,0	0	90	100			100	100
Robustness.Correctness									
	2,0	2,0	0	1	2	2	2		
Robustness.BoundaryConditions									
	0,0	0,0	0	80	100				
Speed									
	0,0	0,0	0	80	100				
ResourceUsage.CPU									
	100,0	0,0	100	80	70	70			
Maintainability.Doc.Code									
	100,0	100,0	0	80	100	100	100		
SynchronizationStatus									
NUnitTests									

Green Week Confirmit Case
The Green Week: Reducing Technical Debt by Engineering
http://www.gilb.com/dl575
May 2013

© 2008 Kai Gilb

A Detailed IE Table

- A MORE DETAILED TABLE.
- THERE IS AN UNDERLYING SET OF DATA FOR EVERY ESTIMATION,
  - AND EVERY VALUE REQUIREMENT SPECIFICATION.
- SO WE CAN CHOOSE TO DISPLAY MORE OR LESS
  - AS OUR PRESENTATION NEEDS DICTATE.

Requirements	DATABASE DESIGN	Symbology	Language Script	COPYRIGHT REVIEW...	Ape Apple Usability	Sum
Usability Status: **0** → Wish: **95** % ... % of [Users] able to complete a [De... [Users = **All**, ...] 18th June 2022	**????** ± 0 0 % of ... **0** ± 0 % 0 *0 % (x 0.0)* ????	**65** ± 25 65 % of ... **68** ± 26 % 68 *7 % (x 0.1)* 68%	**70** ± 25 70 % of ... **74** ± 26 % 142 *7 % (x 0.1)* 74%	**-2** ± 2 -2 % of ... **-2** ± 2 % 140 *-1 % (x 0.3)* -2%	**65** ± 25 65 % of ... **68** ± 26 % 208 *7 % (x 0.1)* 68%	✔ ΣΔ%: **208** ± 80 %
Copyright Law Compliance Status: **0** → Wish: **100** %... % [National Copyright Legislation]... [National Copyright Legislation = 16th July 2019	**15** ± 5 15 % [N... **15** ± 5 % 15 *11 % (x 0.7)* 15%	**22** ± 0 22 % [N... **22** ± 0 % 37 *0 % (x 0.0)* 22%	**2** ± 0 2 % [N... **2** ± 0 % 39 *0 % (x 0.0)* 2%	**95** ± 20 95 % [N... **95** ± 20 % 134 *29 % (x 0.3)* 95%	**-30** ± 10 -30 % [N... **-30** ± 10 % 104 *-3 % (x 0.1)* -30%	⚠ ΣΔ%: **104** ± 35 %
Technical Debt Management Status: **0** → Wish: **50** Min... % [Cost] spent on [Technical Mainten... [Cost = **Total Technical Budge...**] 22nd June 2018	**100** ± 15 100 Minima... **200** ± 30 % 200 *120 % (x 0.6)* 200%	**60** ± 10 60 Minima... **120** ± 20 % 320 *96 % (x 0.8)* 120%	**10** ± 1 10 Minima... **20** ± 2 % 340 *20 % (x 1.0)* 20%	**0** ± 20 0 Minima... **0** ± 40 % 340 *0 % (x 0.2)* 0%	**55** ± 10 55 Minima... **110** ± 20 % 450 *0 % (x 0.0)* 110%	✔ ΣΔ%: **450** ± 112 %
Database Security Status: **0** → Wish: **95** Hac... % of [Unauthroised Access] to [Syst... [Unauthroised Access = **All** 1st June 2022	**90** ± 50 90 Hackers **95** ± 53 % 95 *19 % (x 0.2)* 95%	**40** ± 0 40 Hackers **42** ± 0 % 137 *0 % (x 0.0)* 42%	**45** ± 20 45 Hackers **47** ± 21 % 184 *14 % (x 0.3)* 47%	**10** ± 0 10 Hackers **11** ± 0 % 195 *4 % (x 0.4)* 11%	**65** ± 10 65 Hackers **68** ± 11 % 263 *0 % (x 0.0)* 68%	✔ ΣΔ%: **263** ± 85 %
Sum Of Values: Worst Case: Credibility - adjusted: Worst Case Cred. - adjusted:	**310** ± 88 % 310 *222 %* *150 %* *117 %*	**252** ± 46 % 562 *206 %* *103 %* *84 %*	**143** ± 49 % 705 *94 %* *42 %* *31 %*	**104** ± 62 % 809 *42 %* *32 %* *18 %*	**216** ± 67 % 1025 *149 %* *4 %* *0 %*	
Capital Cost £	**500k** ± 60k	**????** ± 0	**45** ± 30	**256k** ± 32k	**90** ± 1	

Detail of estimates, uncertainty, evidence, source (managing risks of designs)

Requirements	Incentivise	Tea Kiosk	Daily Danger Checks	Su
Project Timeliness =:	8 ± 0	5 ± 1	15 ± 8	
Status: **10** → Wish: **5** % Δ:	-2 %	-5 %	5 %	
% time overrun necessary to deliver ΔΔ%	**40** ± 0 %	**100** ± 20 %	**-100** ± 160 %	ΣΔ
[Project Cost Size = { Medium ($10k -...] ?%:	*32 % (x 0.8)*	*50 % (x 0.5)*	*-80 % (x 0.8)*	
30th June 2017	40%	100%	-100%	
Building Security =:	50 ± 0	50 ± 0	30 ± 10	
Status: **50** → Wish: **10** % I... Δ:	0 % Injury	0 % Injury	-20 % Injury	
% of [Emergency Types] which in fact ΔΔ%	**0** ± 0 %	**0** ± NaN %	**50** ± 25 %	ΣΔ
[Emergency Types = { Earthquake }, ?%:	*0 % (x 0.0)*	*0 % (x 0.6)*	*15 % (x 0.3)*	
30th June 2018	0%	0%	50%	
User Productivity =:	10 ± 0	8 ± 3	15 ± 0	
Status: **15** → Wish: **5** minutes Δ:	-5 minutes	-7 minutes	0 minutes	
number of minutes for a [user] to co... ΔΔ%	**50** ± 0 %	**70** ± 30 %		
[user = { adult }, ?%:	*0 % (x 0.0)*	*56 % (x 0.8)*		
task = { dri...]	50%	70%	0%	
30th June 2017				
Sum Of Values: Σ%:	**90** ± 0 %	**170** ± 50 %	**-50** ± 185 %	
Credibility - adjusted: Σ?%:	*32 %*	*106 %*	*-65 %*	
Method Implementation Cost	500k ± 0	2m ± 0	1m ± 0	
Status: **0** → Budget: **3m** $ Δ:	500k $	2m $	1m $	
Total monetary cost in US Dollars fo... ΔΔ%	**17** ± 0 %	**67** ± 0 %	**33** ± 0 %	ΣΔ
[Project Cost Size = { }] ?%:	*34 % (x 0.0)*	*134 % (x 0.0)*	*66 % (x 0.0)*	
30th June 2017	17%	67%	33%	
Sum Of Development Resources: Σ%:	**17** ± 0 %	**67** ± 0 %	**33** ± 0 %	
Credibility - adjusted: Σ?%:	*34 %*	*134 %*	*66 %*	
Value To Cost:				

Selected Impact Target

Row: User Productivity
Col: Tea Kiosk
Scale: number of minutes for a [user] to complete a [task]

Value Impact: Change...

Estimate: minutes
Δ -7 ± 3

Actual: minutes
Δ scale val ± 0

Credibility: 0.8

In-house measurements of design / strategy correlate to external sources

Evidence:

we have used tea kiosks and several competitors have which save about seven minutes for users

Source:

https://www.tripadvisor.com/ShowUserReviews-g154995-d4871495-r475327934-McDonald_s-London_Ontario.html

Add Comment...

Content Review System. **Bullshit Busters Youtube Channel**

SPEC-LDD0CXQ
v 0.0.1 **New**
by **gilbguest12** - Nov 27th 2018, 15:24

Type: Solution Idea, Labels: -

Is Part Of: Content Review System

Summary:

1. setup a business account on YouTube
2. start a casting for a good **[Presenter]**
3. prepare recording equipment
4. measure how many fake news are present in a **[Newspaper]** and prepare material for recording
5. record an episode about the last recent fake news found in the **[Newspaper]**
6. release the episode on the YouTube
7. measure and test the value against the scale
8. if below Tollerable, repeat 5, 6, 7, 8. If Goal reached, proceed 4 - 8 with another **[Newspaper]**

Newspaper: defined as:

Gazeta Wyborcza, Rzeczpospolita, Sosnowiec Dzisiaj, Warszawskie Słoiki

Presenter: defined as:

A man or woman speaking in the episode

Source:

Iza, Kasjan, Paweł

Description: [Youtube Channel] with [Fake News Analysis] and [Guide For Evaluating] [News] in the terms of [Reliability] an...

Design Spec example

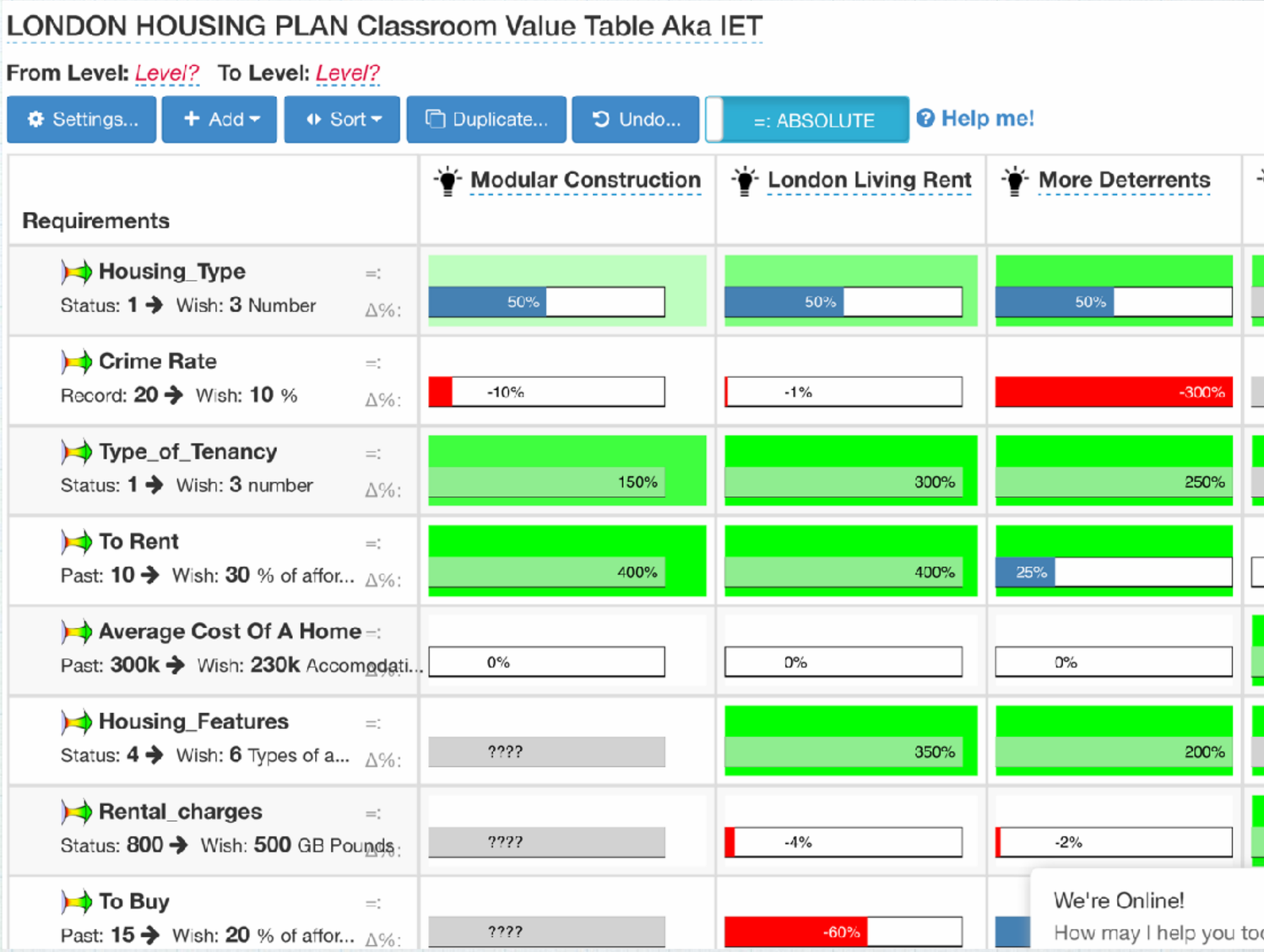
Is there enough info Here

to

Estimate impacts And costs

Value Presentation to/for/by the Architecture process

- The Table is a basic tool for
  - Developing architecture
  - Evaluating Architecture
  - Presenting Architecture
  - Tracking architecture delivery impact
- The Architect's Tasks
  - find a set of designs (the 'architecture') which arguably will deliver the planned Value Levels, within the resources budgeted.
  - document your ideas with estimates and evidence, on an Impact Estimation Table (CE, VP) also called Value Decision Table.



Resource Budgets

Here are some interesting resource categories:

Up Front Costs: *before* producing Value

- Capital Costs Money
- Calendar Time
- Work Hours

Lifecycle Costs

- Money for recurrent lifetime operational costs
- System Maintenance, bug fix, port to new platforms
- Software Licenses
- Premises and Hardware rental
- Training Costs
- Internet Costs

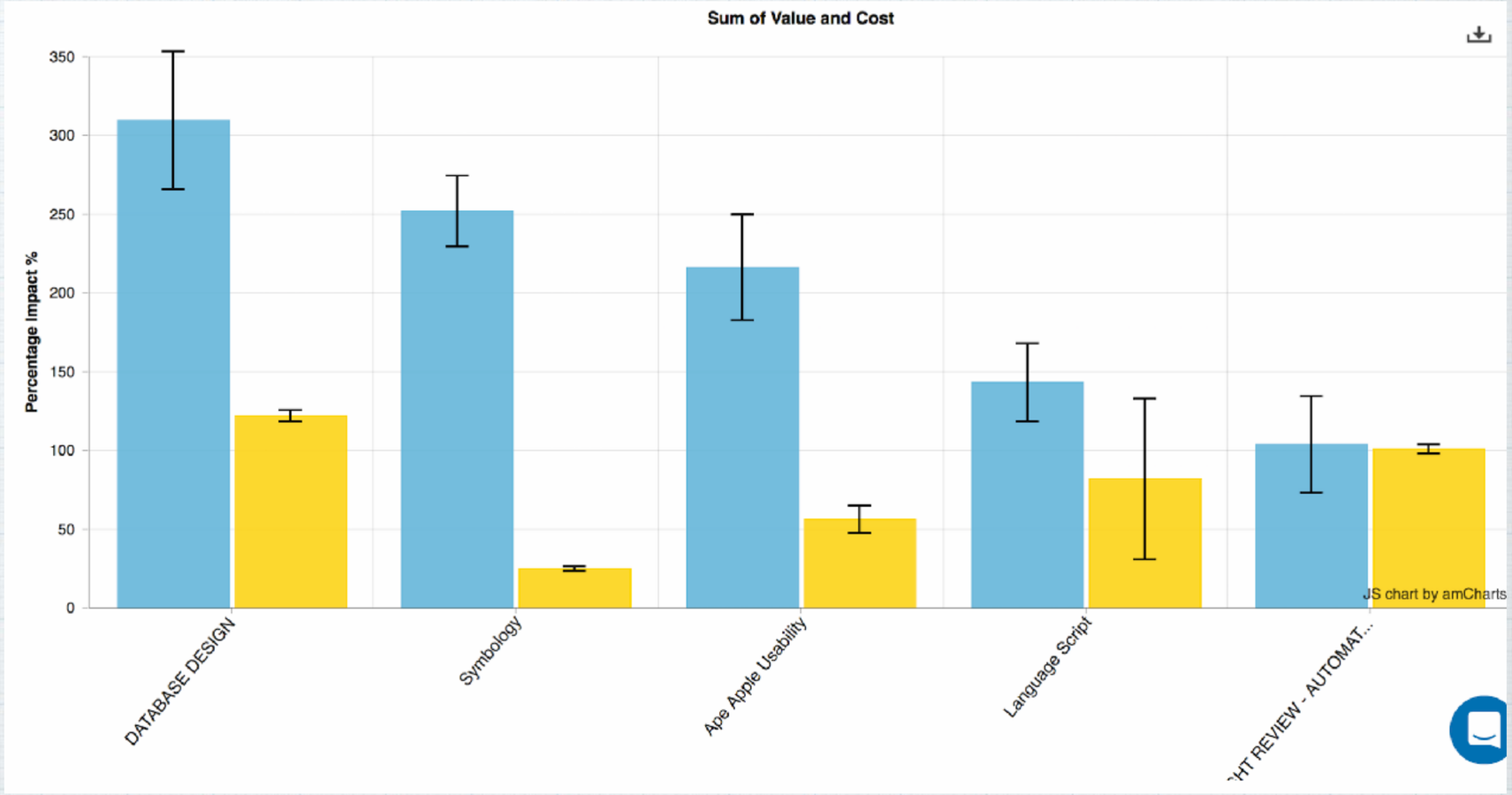
Design Cost Est.

Capital Cost £ Status: **0** → Budget: **3.584m** £ Pounds to deliver the initial set of... *No qualifiers* 31st December 2020	Δ: =: Δ%: ?%:	**500k** ± 60k 500k £ **14** ± **2** % 14 *21 % (x 0.5)* 14%	**????** ± 0 0 £ **0** ± **0** % 14 *0 % (x 0.0)* ????	**90** ± 1 90 £ **0** ± **0** % 14 *0 % (x 0.0)* 0%	**45** ± 30 45 £ **0** ± **0** % 14 *0 % (x 0.0)* 0%	**256k** ± 32k 256k £ **7** ± **1** % 21 *11 % (x 0.4)* 7%	✔ ΣΔ%: **21** ± **3** % Show Sideb
Calendar Cost, Days Status: **0** → Budget: **1k** days days, the time taken to implement [D... [Defined Tasks = **All**...] 30th June 2022	Δ: =: Δ%: ?%:	**1k** ± 50 1k days **100** ± **5** % 100 *150 % (x 0.5)* 100%	**????** ± 0 0 days **0** ± **0** % 100 *0 % (x 0.0)* ????	**60** ± 5 60 days **6** ± **1** % 106 *12 % (x 0.0)* 6%	**400** ± 200 400 days **40** ± **20** % 146 *80 % (x 0.0)* 40%	**60** ± 20 60 days **6** ± **2** % 152 *12 % (x 0.0)* 6%	✔ ΣΔ%: **152** ± **28** %
Full Time Equivalents 4.5 Status: **140k** → Budget: **200k** Pounds £ *No qualifiers* 2018	Δ: Δ%: ?%:	**5k** ± 500 145k Pounds **8** ± **1** % 8 *12 % (x 0.5)* 8%	**15k** ± 2k 155k Pounds **25** ± **3** % 33 *38 % (x 0.5)* 25%	**30k** ± 10k 170k Pounds **50** ± **17** % 83 *100 % (x 0.0)* 50%	**25k** ± 50k 165k Pounds **42** ± **83** % 125 *84 % (x 0.0)* 42%	**50k** ± 1k 190k Pounds **83** ± **2** % 208 *166 % (x 0.0)* 83%	⚠ ΣΔ%: **208** ± **106** %
Maintenance Costs £K Status: **0** → Budget: **1m** ann.. £ cost per Year *No qualifiers* 2022	Δ: =: Δ%: ?%:	**0** ± **0** 0 annual... **0** ± **0** % 0 *0 % (x 0.5)* 0%	**200** ± 7 200 annual... **0** ± **0** % 0 *0 % (x 0.2)* 0%	**13** ± 5 13 annual... **0** ± **0** % 0 *0 % (x 0.0)* 0%	**220** ± 10 220 annual... **0** ± **0** % 0 *0 % (x 0.0)* 0%	**52.8k** ± 10k 52.8k annual... **5** ± **1** % 5 *8 % (x 0.4)* 5%	✔ ΣΔ%: **5** ± **1** %
Sum Of Development Resources: Worst Case: Credibility - adjusted: Worst Case Cred. - adjusted:	Σ%: Σ±%: Σ?%: Σ±?%:	**122** ± **8** % 122 *130 %* *183 %* *65 %*	**25** ± **3** % 147 *28 %* *38 %* *14 %*	**56** ± **18** % 203 *74 %* *112 %* *0 %*	**82** ± **103** % 285 *185 %* *164 %* *0 %*	**101** ± **6** % 386 *107 %* *197 %* *6 %*	

Simplifying the Values/Costs Design Options From complex table to bar charts

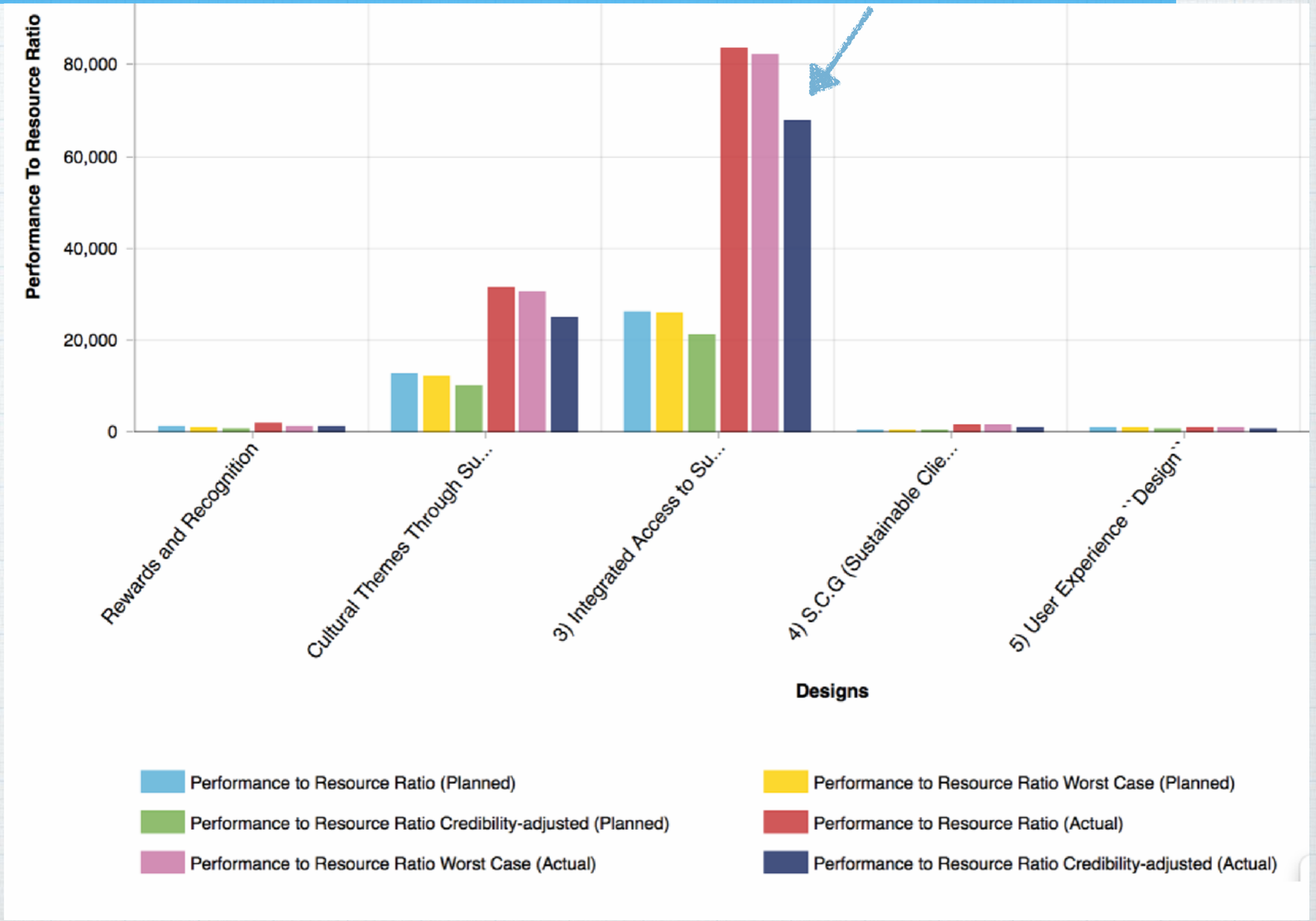
Do you think it is a good practice to implement the most credible designs before the less credible designs?

Do you think we should use the lowest value estimates and the highest cost estimates as a basis for our priorities?



Many 'risk level' options for 'Design Efficiency' (Cost Effectiveness)
Optimist, Uncertainty, Credibiliy, Both (U+C), Actual Result

- **HERE ARE A VARIETY OF PLANNING LEVELS FOR THE DESIGNS, COUPLED WITH THE FEEDBACK FROM ACTUAL IMPLEMENTATION.**
- We can see how good we are are understanding designs, and estimating their worth.
- In this case things are working out better than estimated initially. That is why the 3 columns on the right side are much taller.
- Some designs are far better than others, and this makes that fact visually clearer.
- Notice that his presentation is not just for effectiveness, but for cost-effectiveness.



Part 5.
Starting a larger project, Planning Optimisation Week

DAILY AGENDA

1. Quantify the critical values
2. Draft the best designs to reach the values
3. Build a Table to see if you have pretty good design for the values.
4. And - next part of this book - decompose the designs into weekly do-able value increments
5. Get approval from the ‘Powers That Be’, to start rolling out results, for r4eal, next week.

Top-Level Planning Week

* This process can be shortened to 2 days and even 1 day if you need to.
  * But my experience is that it is then too hectic.
  * You get what you pay for here.
  * The full week gives people time to learn, buy in, discuss, argue, and feel pretty good about the proposals.
  * A week is a small investment to get a big project started better.

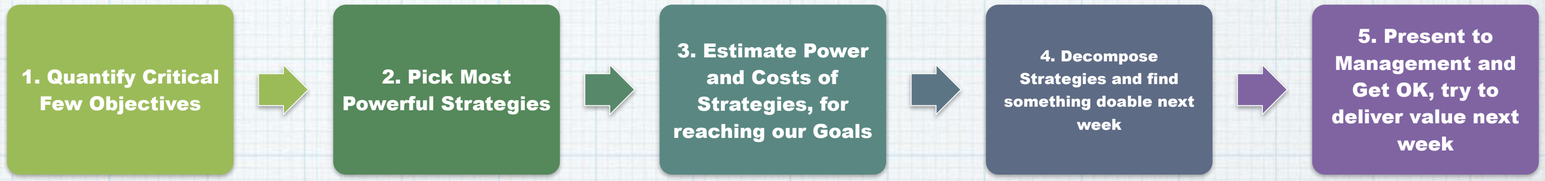
* We build a top-level critical model of our project.
  * We get a balanced idea of the key values to aim for, and the key constraints to respect.

* This top level model, with updates, will become the primary control center for the project.
  * It is for the project management level, and all levels they report to.
  * The essentials of project control on a one page control panel.

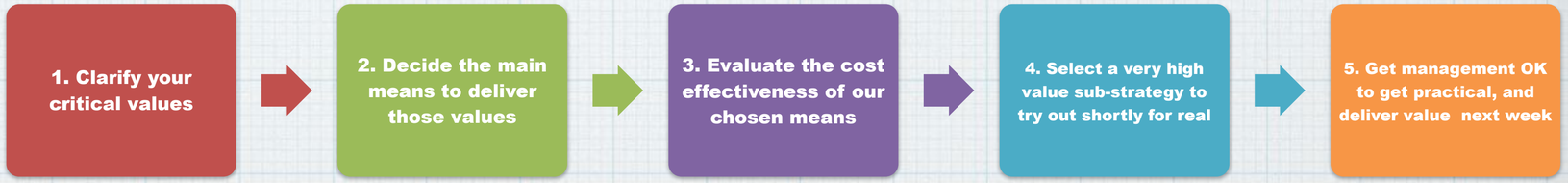
* There is only one essential question: *<u>are we delivering values as planned, for budgets and deadlines we planned?</u>*

* We do not use 'yellow stickies': we digitize the planning,
  * even just in spreadsheets,
  * so we can build on it,
  * as we detail the planning,
  * and progress the value delivery and learning process.

* No 'Infra-structure only',
  * just incremental improvements to previous incremental status
    * in the plans
    * and in the real systems.



The Planning Week Schedule



Purposes of each days tasks

Learning by Doing

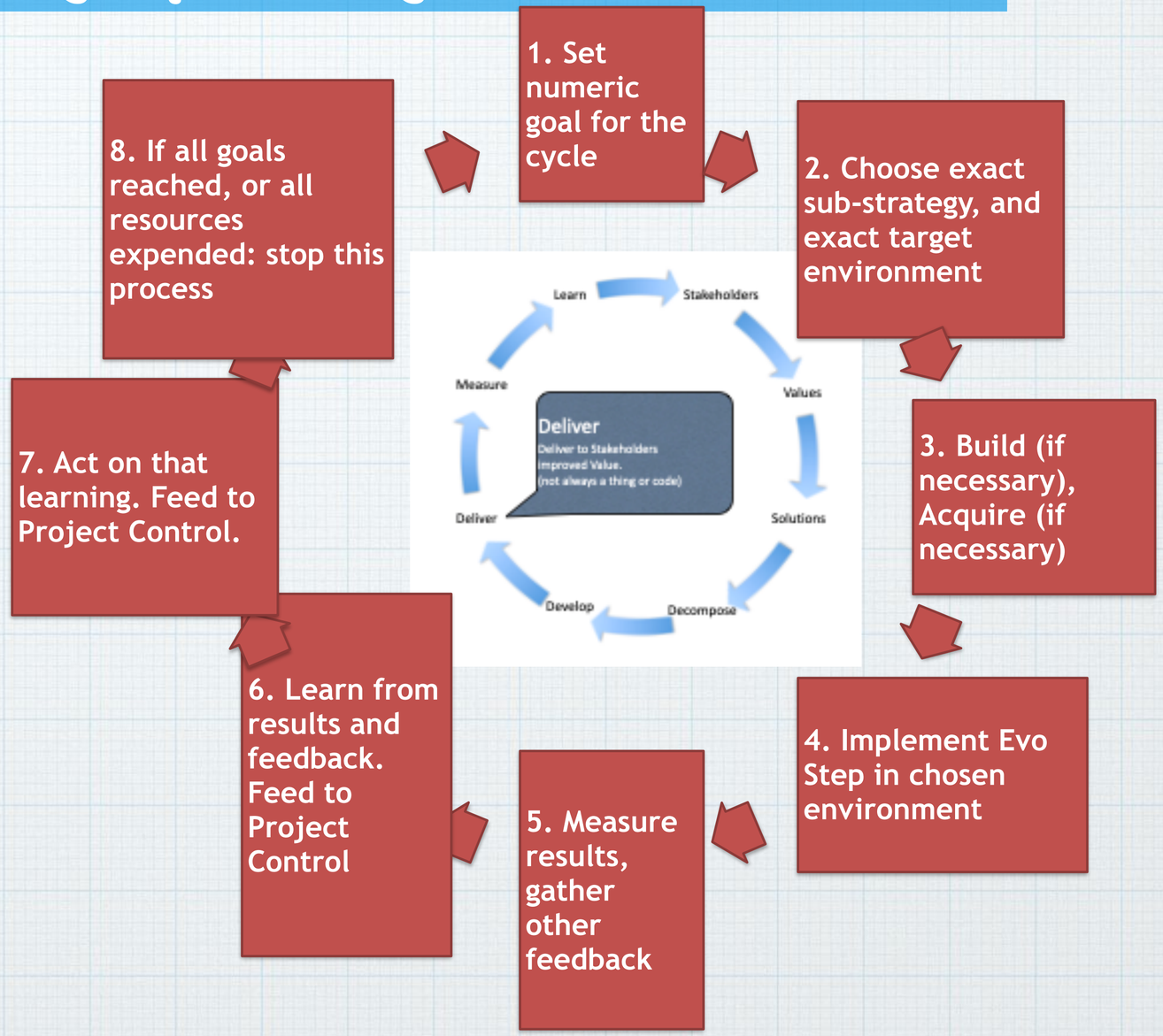
- Participants learn the Planguage methods, on the fly, by doing it.
  - No other training necessary.
  - But a competent coach is necessary,
    - someone who knows what is in this book! You.

- I have personally coached 5 real project teams at once in the same week, and repeated the feat 5 different weeks (= 25 projects) at McDonnell-Douglas Aircraft (now part of Boeing), for aircraft design projects.
  - They liked the results so much they commissioned me to train their coaches and certify them as competent.
  - We always got approval to deliver measurable results from the next week and onwards.
  - What manager could resist?
  - There are many more case studies of the 'startup week' method.
    - (like Ericsson, HP, JP Morgan Bank, DoD)

- We do not build prototypes or mockups of our design.
  - We test our design ideas by *implementing them*
    - *on real existing systems*:
    - but usually on a small scale,
    - a week's work,
    - before we scale up.

- I personally do not trust mockups and prototypes at all.
  - Not for large projects.
  - I do not believe they give us credible enough information.
  - They certainly do not deliver any <u>real value</u> to stakeholders.

- Real and small increments cost roughly the same as prototypes and mockups:
  - but they deliver much more credible <u>feedback</u> from the real world,
  - and above all, they deliver real and measurable <u>value</u>.



'Weekly' 'sprints': Deliver Value and Learn

The big trick to use real, small, value-delivery steps, on large projects is knowing:

- How to *decompose design* into small implementable delivery steps (architecture -> sub-designs)

- How to safely deliver these small steps to real live existing systems, products, services.

- Before you 'get skeptical on me', let me inform you that
  - Elon Musk increments real assembly-line productionTesla cars,
  - with average 20 incremental changes (half hardware, half software) ***weekly***.

  - And he makes a damned fine vehicle for me, too.

  - This is the same method I am talking about. Here

- Safest car in the world, one of my 'Very personal' values!

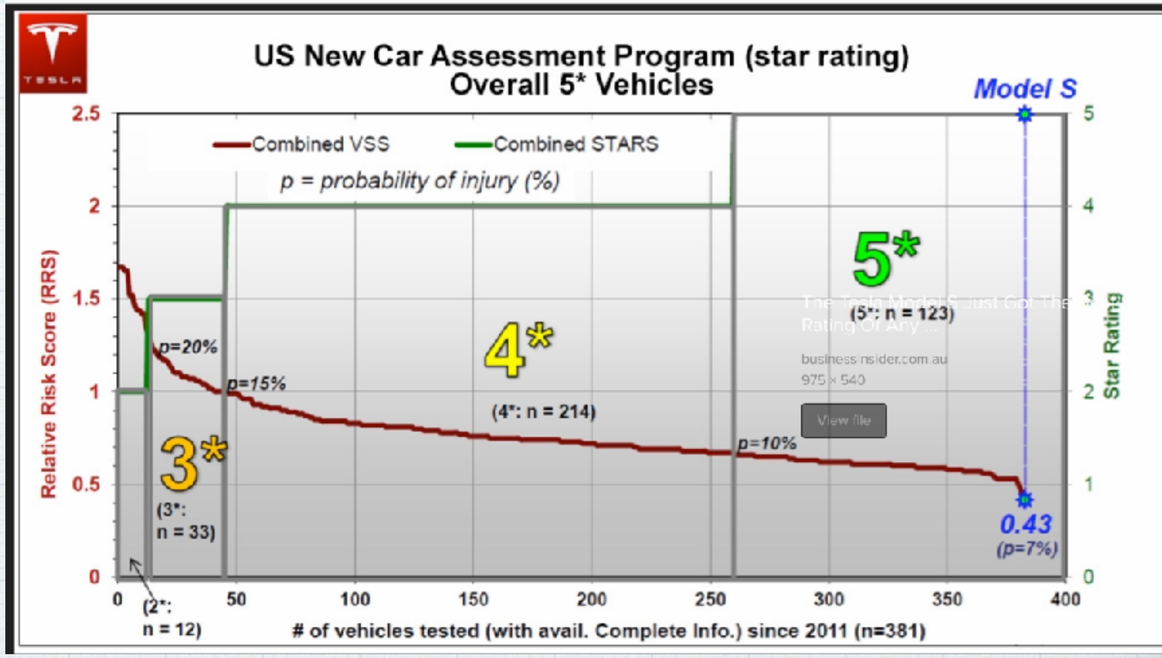


FIGURE: TESLA S SAFETY, DESIGNED IN INCREMENTALLY, AND MODEL 3 GOT BETTER.
NOT BAD FOR A BEGINNER IN CAR INDUSTRY.

WE should NEVER REALLY 'BUILD A NEW SYSTEM' FROM SCRATCH
(no matter how radical the vision and architecture)

- It is worth mentioning that this (Evo, with POW start) is not a process which always assumes we are starting from scratch.

- I have often used it for a major upgrade of *existing* systems, several years old.
  - For example the 8 years old US DOD Persinscom system.

- I normally can assume that the previous system/product/service is out there, right now, in the field, being used by real people.

- I can also assume that the old system badly needs value improvements now, and that is why we are 'starting this project'.

- Your project is not, *ever*, to '*create* a *new* system/product/service'.
  - The *real* project is always, without exception, to *improve the critical values*, of the 'old' system.
  - But this will be a cultural shift for many, and require leadership.
  - "Building and spending are not the game, real value delivery is the scoring mechanism!"

- We can therefore exploit this reality (of existing systems) for *these* purposes:
- As a realistic playground for experiments in design: see how well things *really* work.
- As a possibility to actually improve the 'old' system *immediately*, in critical priority areas. Put design to immediately-useful value improvement. Prove you know how to design usefully.
- As a major risk management strategy, where we do things in small steps, and get feedback before committing more resources. Big failure is impossible with this method.

- Of course there are all kinds of things that are bad and not cost-effective, with the old system.

- And there are all kinds of new improved designs that need to be put in place. But these can both be done, in their own time. Perhaps as an increment, and hopefully a cost-effective increment.

- But there is no need to do major investments in system *replication*, before proving that you can design for real value quickly, when and where it counts.

US DoD. Persinscom **Impact EstimationTable**:

Designs

Requirements

Estimated Impact of Design -> Requirements

Design Ideas ->	*Technology Investment*	*Business Practices*	*People*	*Empowerment*	*Principles of IMA Management*	*Business Process Re-engineering*	*Sum Requireme*
	50%		5%	5%	5%	60%	185%
Availability 90% <-> 99.5% Up time	50%		5–10%	0%	0%	200%	265%
Usability 200 <-> 60 Requests by Users			5–10%	50%	0%	10%	130%
Responsiveness 70% <-> ECP's on time	50%	10%	90%	25%	5%	50%	180%
Productivity 3:1 Return on Investment	45% 50%						303% 251%
Morale 72 <-> 60 per month on Sick Leave							
Data Integrity 88% <-> 97% Data Error %	42%						177%
Technology Adaptability 75% Adapt Technology	5%						160%
Requirement Adaptability ? <-> 2.6% Adapt to Change	80%						260%
Resource Adaptability 2.1M <-> ? Resource Change	10%	80%	5%	50%	50%	75%	270%
Cost Reduction FADS <-> 30% Total Funding	50%	40%	10%	40%	50%	50%	240%
Sum of Performance	*482%*	*280%*	*305%*	*390%*	*315%*	*649%*	
Money % of total budget	15%	4%	3%	4%	6%	4%	36%
Time % total work months/year	15%	15%	20%	10%	20%	18%	98%
Sum of Costs	*30*	*19*	*23*	*14*	*26*	*22*	
Performance to Cost Ratio	*16:1*	*14:7*	*13:3*	*27:9*	*12:1*	29.5 :1	

The Unity Method 111111
for decomposition into iterative value delivery steps
http://www.gilb.com/DL451

The 'Greedy Conspiracy' Will Destroy Your Success.
How to make sure your designs are worth paying for!

* Surely you know there are self-interested powerful forces that want to rebuild the system anew, from scratch.
  * They are internationally experienced at seducing weak managers,
  * and your own professionals to believe that his major swap-out is 'absolutely necessary'; the only option.

* This is their opportunity
  * to make a pile of money,
  * selling you hardware and services,
  * you really do not need, in order to deal with improving ***your*** critical values quickly.

* There is a way of testing their sincerity.
  * Offer them a pure 'no cure (Values) no pay' contract.

* Offer them 50% more than their fixed price offer,
  * but only if they deliver your quantified critical value levels on time.
  * Watch them run away and make invalid or false excuses.

LLOYD'S STANDARD FORM OF SALVAGE AGREEMENT

(Approved and Published by the Council of Lloyd's)

NO CURE - NO PAY

1. Name of the salvage Contractors: (referred to in this agreement as "the Contractors")

2. Property to be salved: The vessel: her cargo freight bunkers stores and any other property thereon but excluding the personal effects or baggage of passengers master or crew (referred to in this agreement as "the property")

3. Agreed place of safety:

4. Agreed currency of any arbitral award and security (if other than United States dollars)

* They want your money up front, as they build,
  * even when no critical value results are visible.
  * They want to keep that money, even when the project is a total failure,
  * or the value is a great disappointment.
  * Do you really want to be their next victim?

* The greedy irresponsible suppliers are bad enough,
  * but the willing victims, usually using someone else's money (taxpayers money),
  * are a reflection of poor management and business school training.

* They are not Value focussed, I have found.
  * Money Money Money.
  * They do not learn to quantify and manage most critical values.

* To top it all off, your own professional staff is keen to support the idea of a ***completely new system***,
  * since they will get updated skills,
  * so they can leave you when the project fails, for better pay.

* If you need a refresher on Big Bang project failure,
  * google it.
  * It is pervasive. (Failure, reports, cases, analysis of failures)

No Cure Paper

http://www.gilb.com/DL38

Twelve Tough Questions:
Tools to fight the greedy suppliers

A way to sum up this paper for managers!

1. Numbers

Why isn't the improvement *quantified*?

2. Risk

What is the degree of *risk* or uncertainty, and *why*?

3. Doubt

Are you *sure*? If not, w*hy* not?

4. Source

Where did you get *that* information? How can *I* check it out?

5. Impact

How does *your* idea effect *my goals and budgets*, measurably? **6. All critical factors**

Did we *forget anything critical* to survival?

7. Evidence

How do you *know* it works that way? Did it '*ever*'?

8. Enough

Have we got a *complete* solution? Are *all* requirements satisfied?

9. Profitability first

Are we planning to do the '*profitable* things' first?

10. Commitment

Who is responsible for failure, or success?

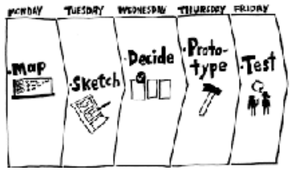
11. Proof

How can we be *sure* the plan is working, *during* the project, *early*?

12. No cure, no pay

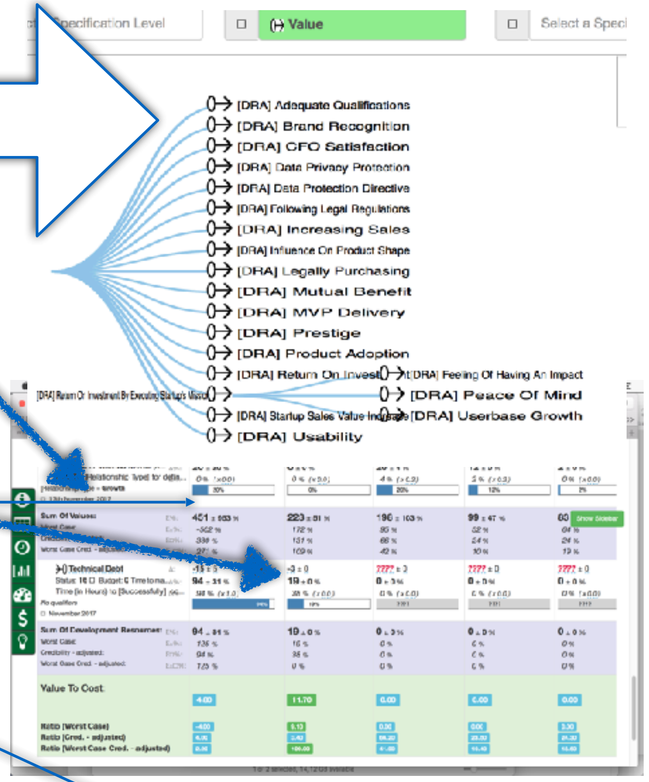
Is it '*no cure, no pay*' in a contract? Why not?

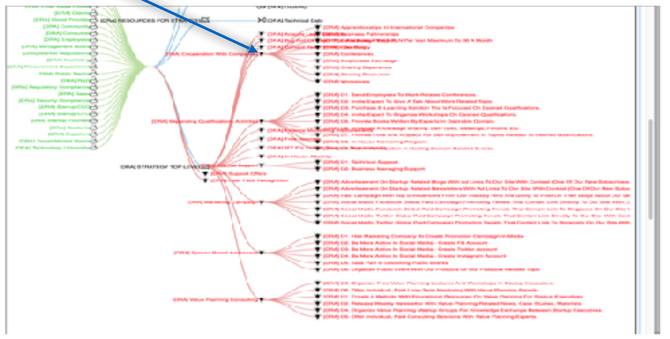
http://www.gilb.com/dl24

An advanced ‘Design 

Sprint’ for grownups.

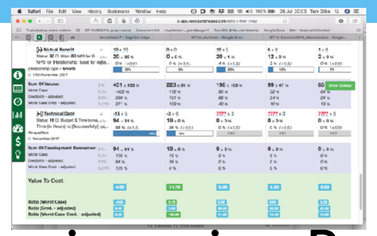
- The Startup Week*. Agile Value Delivery **
- Monday
  - Quantify critical stakeholder values
- Tuesday
  - Identify top 10 strategies or designs to each the values
- Wednesday
  - Rate strategies versus values and costs, and risks on an Impact Table
- Thursday
  - Decompose best strategy, and rate value/costs of details to choose next week’s value delivery
- Friday
  - meet with managers to get OK
- Next week (and every week later)
  - deliver some measurable stakeholder value
  - measure results, costs
  - learn about problems early
  - adjust designs for future



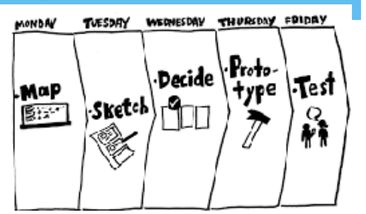


- * source is ‘Polish Export’ examples in ‘Innovative Creativity’ book (gilb.com) chapter 9. Done over 2 days with 60 people in 20 teams. Warsaw, at Startberry (startup Incubator)
- ** http://www.gilb.com/dl812, gilb.com/dl568
- DL812: extensive slides, DL568: short paper, see ‘Presenter Notes ‘in this slide.

‘Project Startup’ versus ‘Design Sprint’



Planguage Evo



- Engineering Based
- Systems Applicable (UX)
- All Values Quantified
- Risk Mgt (±.Cred, Prty)
- Scale-Free
- Decades of Experience
- Research Published: HP
- Many publ.Case Studies
- AI Prioritization Val/€
- Design estimates V&€
- Actual incr. measures
- Digital Planning Long Term

- Programming Craft
- Software and UI Limited
- Values Not Quantified
- No Explicit Risk Mgt.
- Not proven large scale
- Hot new idea
- No known research
- Can’t find cases, yet
- Role player decides pri.
- No estimates
- Dodgy Prototype
- Yellow Sticky Culture

gilb.com/dl568

See Presenter Notes for references

Design Sprint 'Claimed Benefits' <-Jake
(of course YOU are skeptical, and know this.)

"8 incredible Design Sprint benefits for your business"

"Here are the 8 amazing Design Sprint benefits you get in your business by employing this methodology of Google:

1. Design Sprint helps you save time and money

Design Sprint is designed to work quickly and intensely to get a solution to a business problem through design.

By using Design Sprint you reduce the time you spend on the design process and the process of defining your product, going from months to days

This is a great benefit because you save a lot of time and money and allows you to define a validation plan based on the feedback from your users.

2. Design Sprint Quickly Reduces Product Development Cycles

Derived from the above, development times are dramatically reduced, as Design Sprint work on a connecting problem with the solution. This helps you to test whether an idea works or not, without developing products with very long production cycles (Idea, Design, Approve, Develop, Launch and Validate).

With the Design Sprint you become a more agile organization

Before investing in the development of your product or a new functionality that requires an expensive process you can dedicate 5 days so that the team understands the problem that your company is facing, designing the solutions, creating a functional prototype and validating your ideas in a matter of hours. Becoming a more agile organization.

3. Real feedback with Design Sprint

Knowing the feedback of your product is fundamental to developing successful products. Many times when we get this information is when we have finished the project.

With the Design Sprint, you know firsthand and quickly the real feedback from your customers. This feedback is crucial because it helps you improve your product or service at the same time you design it

On the other hand, your team is actively working on the process, as the production cycle involves different sources of information within your organization.

4. Validate your business ideas with Design Sprint

Without validation, it is difficult for ideas and products to work. That is precisely what you will do on the last day of the Sprint in a very concrete way.

Through Design Sprint you can design the validation plan of the business idea or functionality of your product

Being clear how the process will be, the time you are going to invest and the type of results with which we can continue the process of transferring your product to the market.

5. Generates business and innovation.

Design Sprint gives your team a way of working to solve complex problems in a week.

So you can achieve a new approach to the project that would have taken months, even years

6. Align expectations with your team

Making all departments share knowledge, needs, and strategy so that the result is a solution that satisfies and meets needs.

Being able to make your step to deploy is a cycle of continuous product integration

7. Help you measure

The sprint design uses measurement processes in the different phases that the methodology uses.

What allows you to measure the results obtained at the end of the process, as well as the impact of the same on your business and on the equipment and surplus generated during the process

8. An agile and fast methodology that you can apply to your business

Once you internalize the Design Sprint methodology you can use it and coordinate it with other processes that you already have established in your project or business.

Typically, the first time you make a Sprint Design is tiring and difficult.

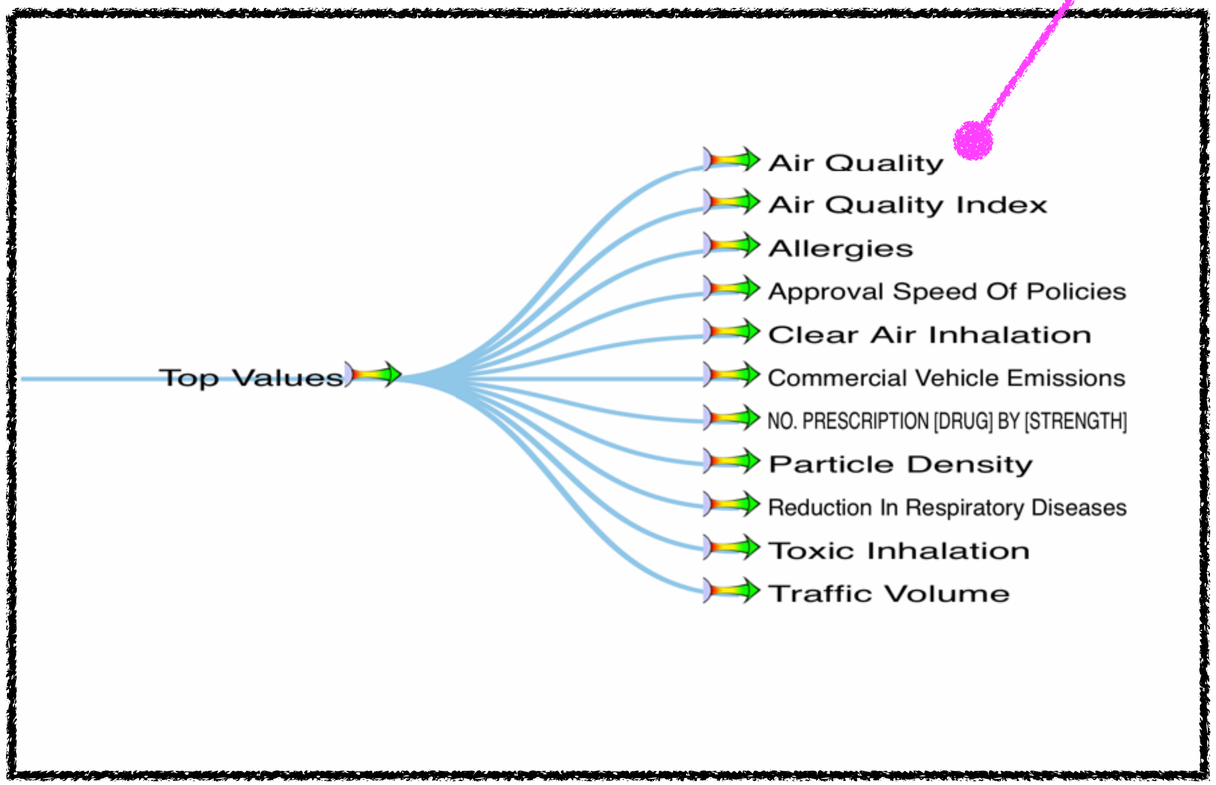
We recommend that you count with the help of a Sprint Master Certified to achieve these incredible results"

Skeptical Observations <-TSG

- These claims are made by a **seller** of 'Design Sprint' training and certification service (letshackity.com)
- Most of the terms and concepts have **poor definition**, and are *highly ambiguous* (examples)
  - Design, Align Expectations, Investing (Product Dev), Complex Problems, measure the results, agile methodology, validation, and many more.
- Not one single number is offered to indicate the magnitude of improvements
- No clear baseline (who is going to get improved) is indicated
- No references to real case studies with results, costs, problems
- No comparison with any other known methods
- No links or references to anything
- Lots of causal assertions, none proven
- "This feedback is crucial because it helps you improve your product or service at the same time you design it"
- No indication or example of the types and magnitude of the costs for the individual, the project, and the organization for learning and maintaining the Design Sprint method
- No glowing references from real people or customers
- No information about how things went after the first week, to tell us how good or bad the week was.
- Constant implication: Google is successful, therefore this method is good

http://www.letshackity.com/en/design-sprint-benefits-business-innovation/

Day 1: The Top Ten (or 11) Critical Stakeholder Values Quantified on a Page



Air Quality

Level: Stakeholder, **Type:** Value, **Labels:** critical Edit

Is Part Of: Top Values

Status	Wish	Status	Wish	Stretch	Tol
9.5k	950	1.5k	150	0	

Status [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] @ 2016 : **9.5k** People <- *Sadiq Khan, Mayor of London Website, Press Release Toxic air*

AQ.Ambition Level: Drastically improve air quality in London to acceptable legal levels as stated in the Paris ..

Scale: Number of [Persons] who reside in London Boroughs dying from exposure over [Time] to [Pollution] pe..

Status: 9.5k People [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] When 2016 1

ALL2020.Wish: 950 People [Persons = **<All>**, Pollution = **<All>**, Area = **<All>**] When 2020

Status: 1.5k People [Persons = **Senior**, Pollution = **NO2**, Area = **<All>**] When 2016

Wish: 150 People [Persons = **Senior**, Pollution = **NO2**, Area = **Greater London**] When 2020

Stakeholders: Mayor Of London, Mayor of London

Assumption: IssueActionStatistic on number of deaths from pollution per year from Mayor of London press r..

Assume1.Assumption: IssueActionDemographic distribution for population of London matches demographi..

Authority: On behalf of the Mayor of London, responsible for making London a better place

Authority: The ruling of the Paris Accords

Authority: The voting population of London Boroughs (as can vote in the London Mayoral elections)

Dependencies: Air Quality Index An improvement in this requirement will directly impact the number of peopl..

Issue: IssueAction

Intended Readership: House of Lords, London Assembly, General Electorate, Parliament

Meter: Recent Hospital records from London hospitals for deaths by Pollution Exposure related illness

Mission: Improve Air Quality in London and reduce the number of deaths related to long term exposure to se..

Rationale: People dying is bad. This is the tip of the iceberg, the worse sufferers from pollution.

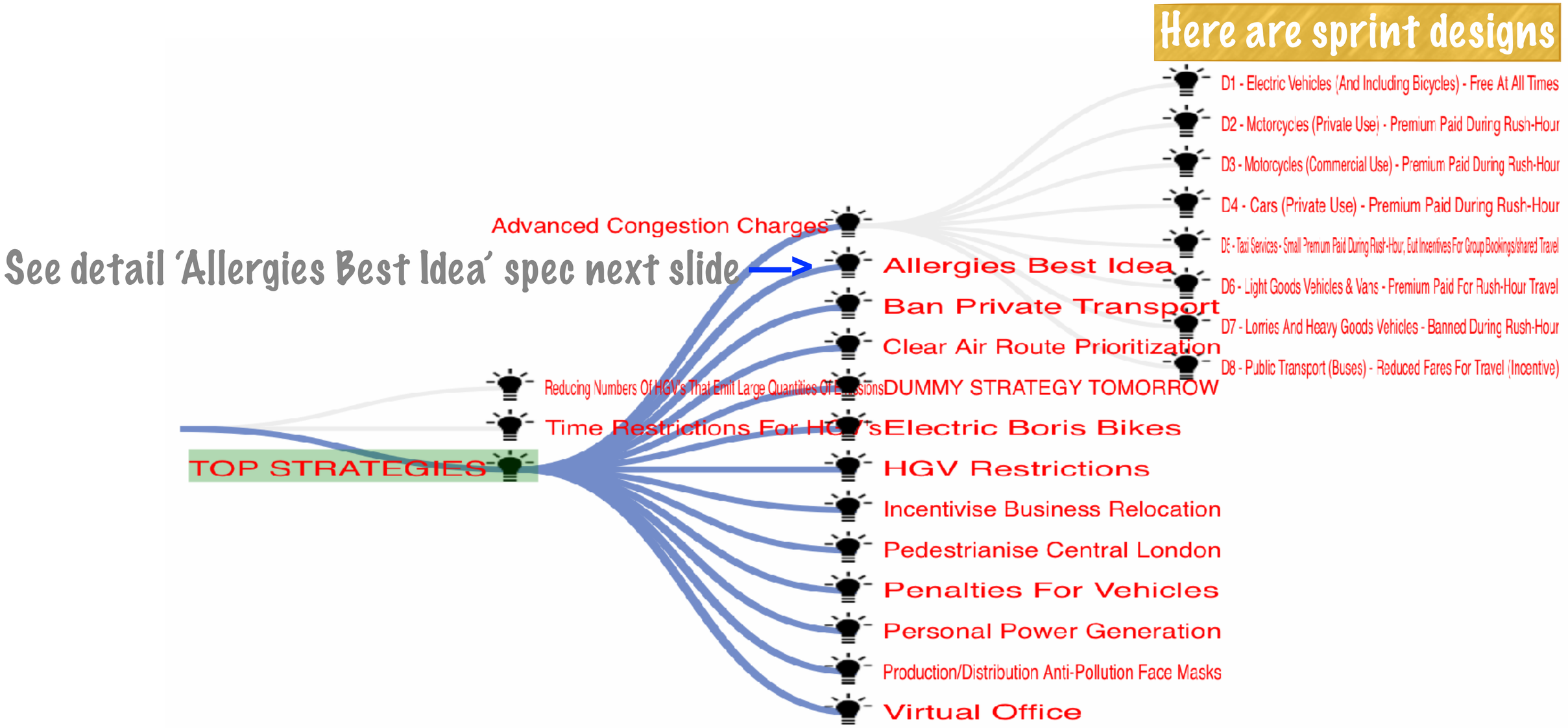
Risk: There is a risk that Brexit changes the status-quo.

Stretch: 0 People [Persons = **Senior**, Time = **5 years**, Pollution = **NO2**, Area = **Greater London**] When 2022

Tolerable: 200 People [Persons = **Senior**, Time = **5 years**, Pollution = **NO2**, Area = **Greater London**] When 2017

Trend: 350 People [Persons = **Senior**, Time = **5 years**, Pollution = **NO2**, Area = **Greater London**] When 2017

Day 2: The Top-Ten Best Designs: the architecture to deliver the values



Allergies Best Idea

Stakeholder Solution Idea *Empty* (by - a few seconds ago)

Permalink 0.0.1

Is Part Of: TOP STRATEGIES

Summary: reduce emission in congestion zone

Description: Change... (by **tomgilb** - a few seconds ago) 0

1. No charge for electrical vehicles
2. Reduce charges by 50% for low emission vehicles
3. increase congestion charge for high emission vehicles
4. introduce congestion charges on weekends for high emission vehicles.
5. introduce special congestion zone around schools

Source:

group 2

Add Comment...

Value Impact: (by **gilbguest6** - 4 months ago) 0

Scale Level Impact (Estimate): = **-18800** ± 50 People (Credibility: **0** Wild guess, no credibility)

Air Quality

Number of **[Persons]** who reside in London Boroughs dying from exposure over **[Time]** to **[Pollution]** per year in **[Area]**

Status: 9500 People [Persons = <All>, Time = , Pollution = <All>, Area = <All>]

Wish: 150 People [Persons = Senior, Time = , Pollution = NO2, Area =

Day 2
Sample Design idea
For reducing pollution in London
BCS Course

London Pollution Planning
BCS 2016

TOP STRATEGIES.**Advanced Congestion Charges**

SPEC-QF5A3AZ
v 0.0.1 **Draft**
by **tomgilb** - Mar 7th 2017, 14:38

Level: Stakeholder, **Status:** Not Determined **Type:** Solution Idea, **Labels:** no labels

Is Part Of: TOP STRATEGIES

Consists Of: D1 - Electric Vehicles (And Including Bicycles) - Free At All Times D2 - Motorcycles (Private Use) - Premium Paid During Rush-Hour D3 - Motorcycles (Commercial Use) - Premium Paid During Rush-Hour D4 - Cars (Private Use) - Premium Paid During Rush-Hour D5 - Taxi Services - Small Premium Paid During Rush-Hour, But Incentives For Group Bookings/shared Travel D6 - Light Goods Vehicles & Vans - Premium Paid For Rush-Hour Travel D7 - Lorries And Heavy Goods Vehicles - Banned During Rush-Hour D8 - Public Transport (Buses) - Reduced Fares For Travel (Incentive)

Summary: Advanced congestion charges reflected by the grouping and categorisation of vehicles (as specified in …

Description:

Advanced congestion charges reflected in the following groups/categories:-

D1 - Electric vehicles (and including bicycles) - free at all times

D2 - Motorcycles (private use) - premium paid during rush-hour

D3 - Motorcycles (commercial use) - premium paid during rush-hour

D4 - Cars (private use) - premium paid during rush-hour

D5 - Taxi services - small premium paid during rush-hour, but incentives for group bookings/shared travel

D6 - Light goods vehicles & vans - premium paid for rush-hour travel

Lorr banned during rush-hour

Pub ares for travel (incentive)

We're Online!
How may I help you today?

Day 2

An Architecture Defined As 8 sub-designs

ValPlan | Dashboard | Canvas | Tables | More... | Create | LONDON CONGE... | tomgilb

LONDON CONGESTION WRT 'AIRQUALITY' / Value Decision Tables / London Pollution Planning

London Pollution Planning

From Level: *Level?* **To Level:** *Level?*

Settings... | Add | Help me! | Show VDT Sideb

equirements	Personal Power Ge...	Allergies Best Idea	Advanced Congesti...	Penalties For Veh...	Clear Air Route P..	HGV Restrictions	Sum
Air Quality Index Past: **135** → Wish: **67** µg/m³	29 %	51 %	74 %	88 %	0 %	-7 %	Σ Δ%: **235** %
Air Quality Status: **9.5k** → Goal: **150** People	-30 %	303 %	37 %	43 %	0 %	0 %	Σ Δ%: **353** %
Allergies Status: **10** → Wish: **1** number of ...	33 %	33 %	44 %	22 %	33 %	-56 %	Σ Δ%: **109** %
Approval Speed Of Policies Status: **6** → Goal: **3** Mnths	0 %	67 %	0 %	0 %	0 %	-33 %	Σ Δ%: **34** %
NO. PRESCRIPTION [DRUG] BY ... Status: **1k** → Wish: **100** NUMBER	22 %	22 %	39 %	50 %	11 %	0 %	Σ Δ%: **144** %
Clear Air Inhalation Status: **20** → Wish: **70** %	30 %	40 %	36 %	50 %	40 %	4 %	Σ Δ%: **200** %
Particle Density Status: **1k** → Wish: **300** Number of ...	26 %	86 %	54 %	50 %	0 %	0 %	Σ Δ%: **216** %
Reduction In Respiratory Di... Status: **1k** → Wish: **100** PATIENTS	33 %	56 %	17 %	78 %	17 %	0 %	Σ Δ%: **201** %
Toxic Inhalation Status: **100** → Wish: **10** Max Mg Pol...	28 %	56 %	61 %	50 %	0 %	-1 %	Σ Δ%: **194** %
um Of Values: Σ%:	**171** %	**714** %	**362** %	**431** %	**101** %	**-93** %	
LABOUR EFFORT Status: **0** → Budget: **1k** WORK MONTHS	2 %	6 %	30 %	30 %	10 %	10 %	Σ Δ%: **88** %
£ CAPITAL COSTS Status: **0** → Budget: **1m**	0 %	24 %	30 %	50 %	30 %	3 %	Σ Δ%: **137** %
um Of Development Resources: Σ%:	**2** %	**30** %	**60** %	**80** %	**40** %	**13** %	
alue To Cost:	85.50	23.80	6.00	5.40	2.50	-7.20	
atio (Worst Case)	44.00	16.80	4.20	5.10	1.70	-7.20	
atio (Cred. - adjusted)	-3.80	4.90	0.30	0.10	0.70	0.00	
atio (Worst Case Cred. - adjusted)	-32.90	6.00	23.50	7.90	41.00	0.00	

Day 3

LONDON CONGESTION Simplified Impact TABLE

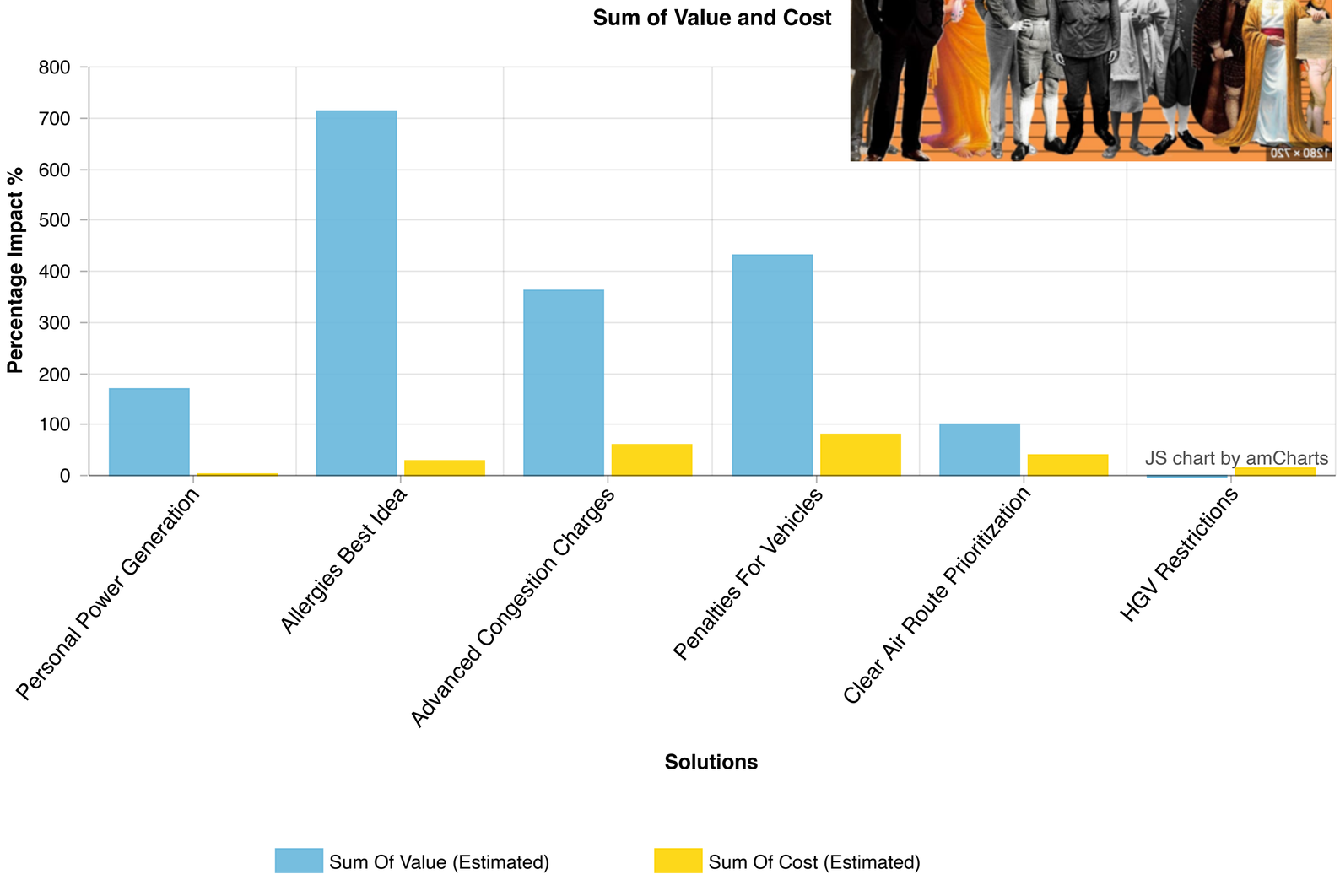
UNSORTED

WITH VALUES TO COSTS CALCULATED (at bottom)

London Pollution Planning

From Level: *Level?* **To Level:** *Level?*

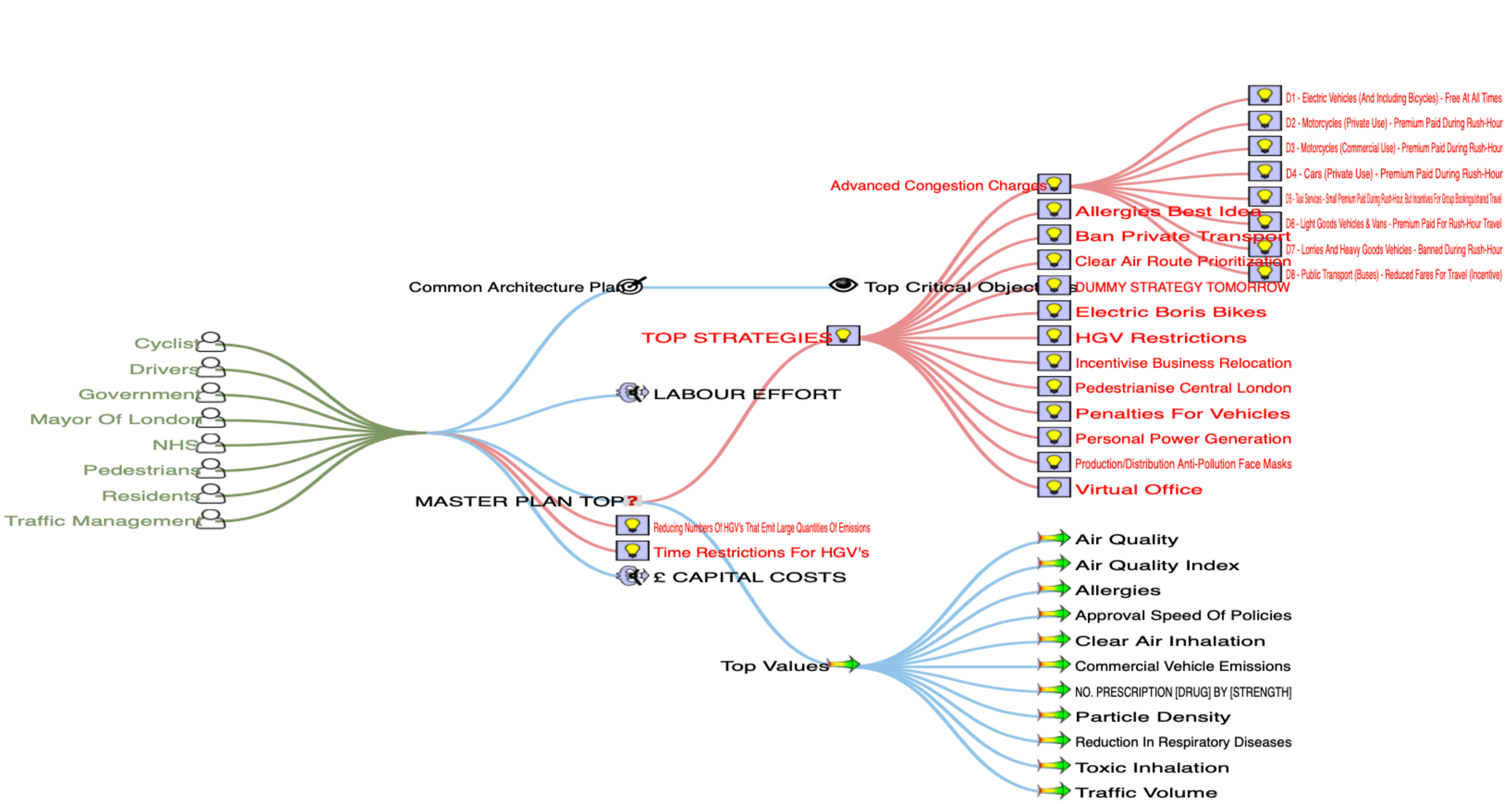
Settings... | Add | Help me!



Day 3

LONDON CONGESTION BAR CHART VALUE TO COST HIGHEST TO LOW LEFT TO RIGHT QUEUE

London Pollution case top level diagram, with 2 level design



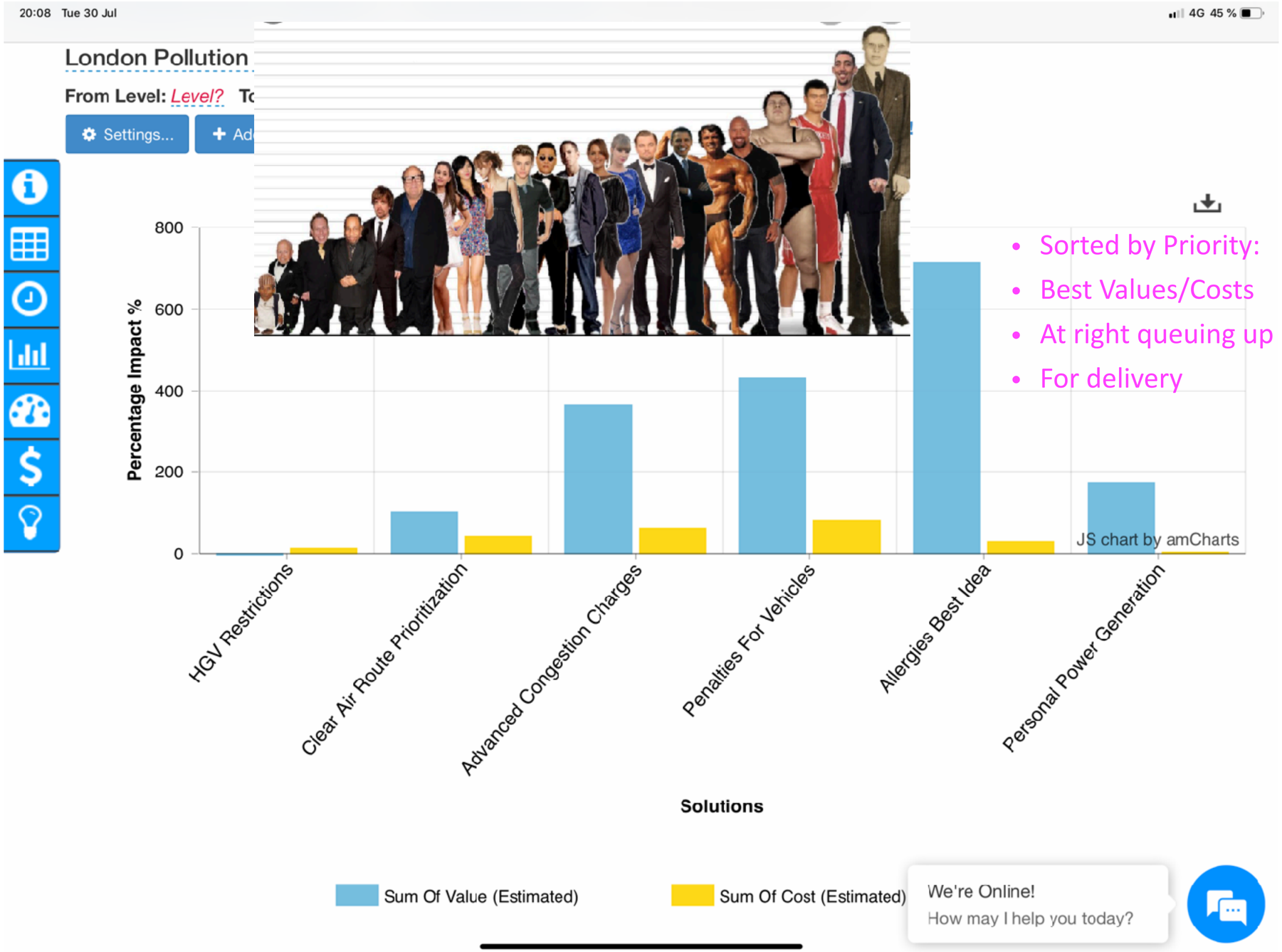
Day 3:
Value Table: estimate how cost-effective your designs are

- See next slide
- For
- Simplification
- Priority Design
- Bar Chart

Day 3:
Value Table: estimate how cost-effective your designs are

Requirements	HGV Restrictions	Clear Air Route P...	Advanced Congesti...	Penalties For V
Air Quality Index Past: 135 → Wish: 67 µg/m³	-7%	0%	74%	88%
Air Quality Status: 9.5k → Goal: 150 People	0%	0%	37%	43%
Allergies Status: 10 → Wish: 1 number of ...	-56%	33%	44%	22%
Approval Speed Of Policies Status: 6 → Goal: 3 Mnths	-33%	0%	0%	0%
NO. PRESCRIPTION [DRUG] BY ... Status: 1k → Wish: 100 NUMBER	0%	11%	39%	50%
Clear Air Inhalation Status: 20 → Wish: 70 %	4%	40%	36%	50%
Particle Density Status: 1k → Wish: 300 Number of	0%	0%	54%	50%
Reduction In Respiratory Di... Status: 1k → Wish: 100 PATIENTS	0%	17%	17%	78%
Toxic Inhalation Status: 100 → Wish: 10 Max Mg Pol	-1%	0%	61%	50%
Sum Of Values:	-93 %	101 %	362 %	431 %
LABOUR EFFORT Status: 0 → Budget: 1k WORK MONTHS	10%	10%	30%	30%
£ CAPITAL COSTS Status: 0 → Budget: 1m	3%	30%	30%	

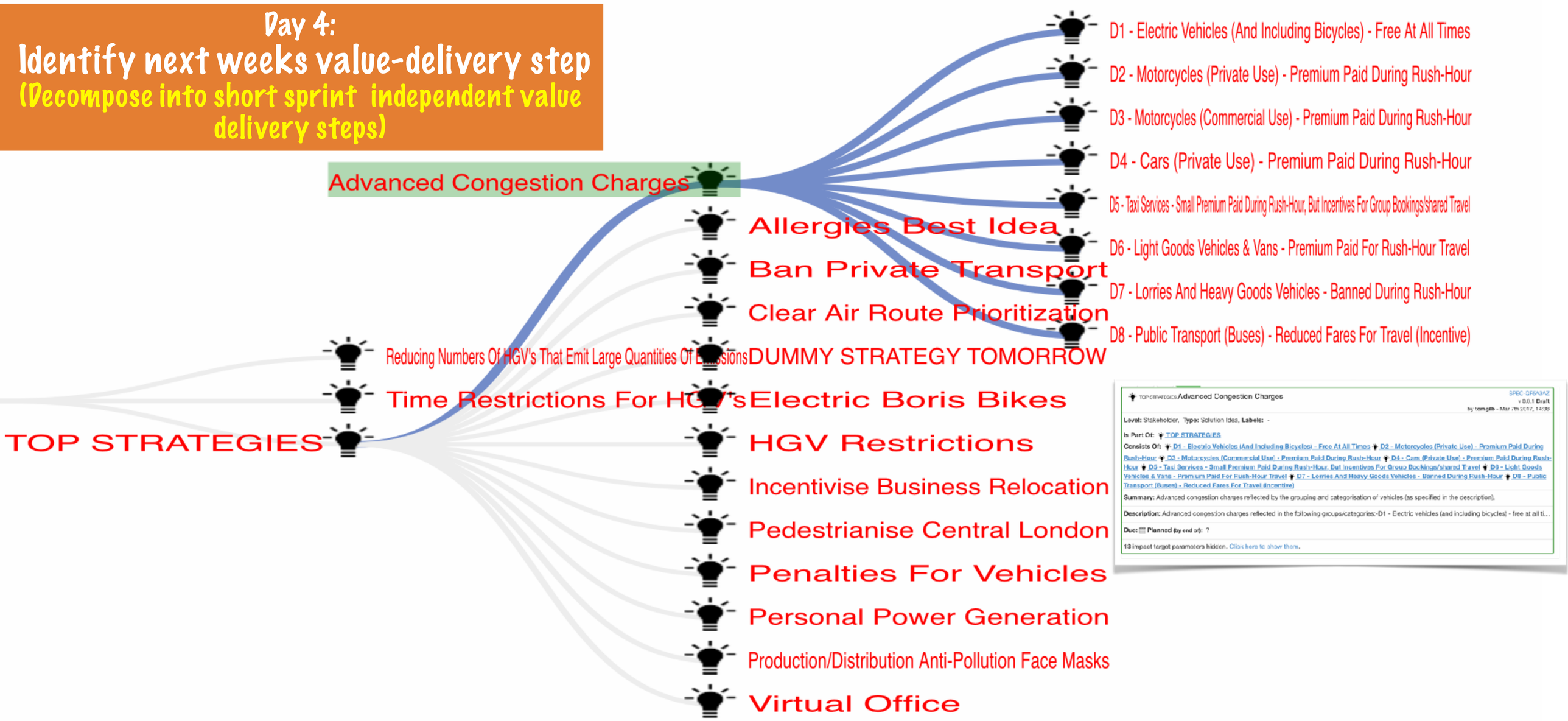
We're Online!
How may I help you today?



- Sorted by Priority:
- Best Values/Costs
- At right queuing up
- For delivery

We're Online!
How may I help you today?

Day 4:
Identify next weeks value-delivery step
(Decompose into short sprint independent value delivery steps)



London Pollution Planning BCS 2016

Day 4:

Identify next week's value-delivery step.

Sort the 'sprint sized' value delivery designs by values/costs delivery priority

Expanding Qualification Activities Value Table

From Level: *Level?* To Level: *Level?*

Settings... | Add | Sort | Duplicate... | Undo... | =: ABSOLUTE | Help me! | Show Sidebar

uirements	D3. Purchase E-L...	D2. Invite Exper...	D1. Send Employe...	D4. Invite Exper...
Adequate Qualifications — Status: 39 → Wish: 0 % [Employe... — % [Employees] have acquired [Skill Le...	97%	38%	51%	23%
Adequate Qualifications — Status: 61 → Wish: 0 % [Employe... — % [Employees] have acquired [Skill Le...	93%	26%	2%	67%
Adequate Qualifications — Status: 0 → Wish: 50 % [Employe... — % [Employees] have acquired [Skill Le...	60%	36%	30%	44%
Adequate Qualifications — Status: 0 → Wish: 35 % [Employe... — % [Employees] have acquired [Skill Le...	6%	31%	14%	34%
Adequate Qualifications — Status: 0 → Wish: 15 % [Employe... — % [Employees] have acquired [Skill Le...	0%	13%	7%	8%
n Of Values:	246 %	144 %	104 %	228 %

FIGURE: HERE, FROM ANOTHER PLAN, IS A VALUE TABLE FOR DECIDING WHICH ONES OF THE SUB-DESIGNS ARE TO BE PRIORITIZED NEAR TERM (SOURCE POLISH EXPORT PLAN)

•

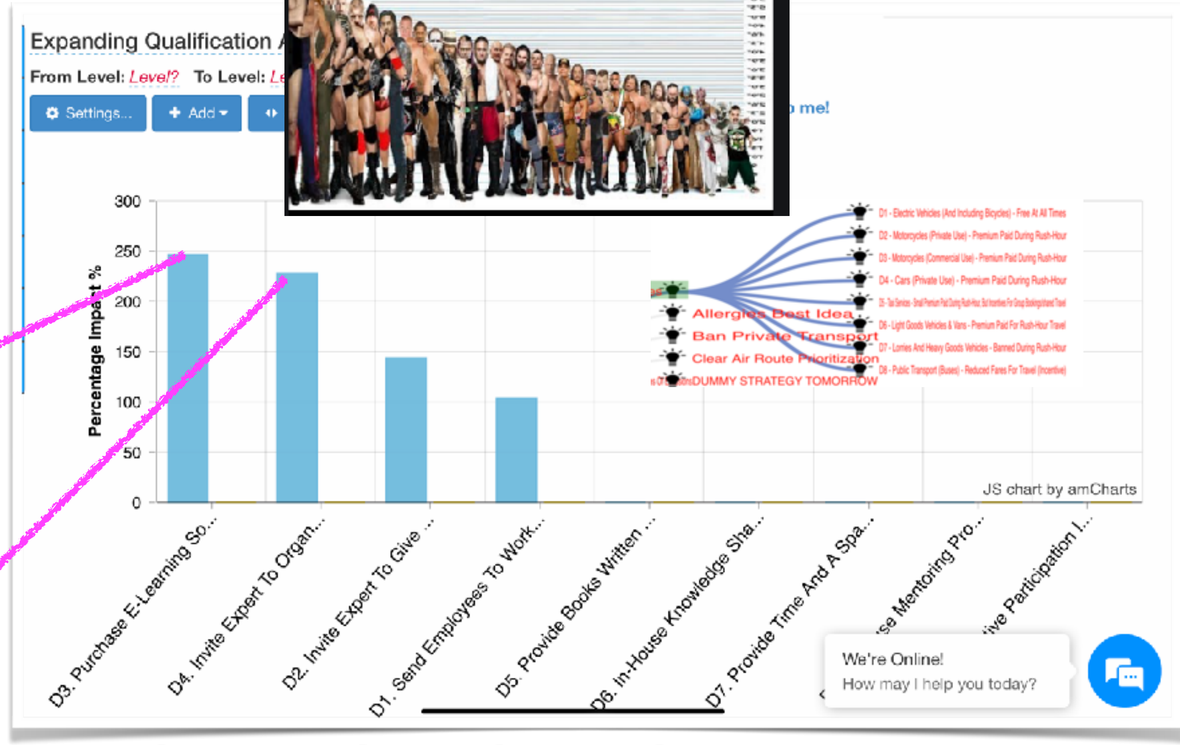


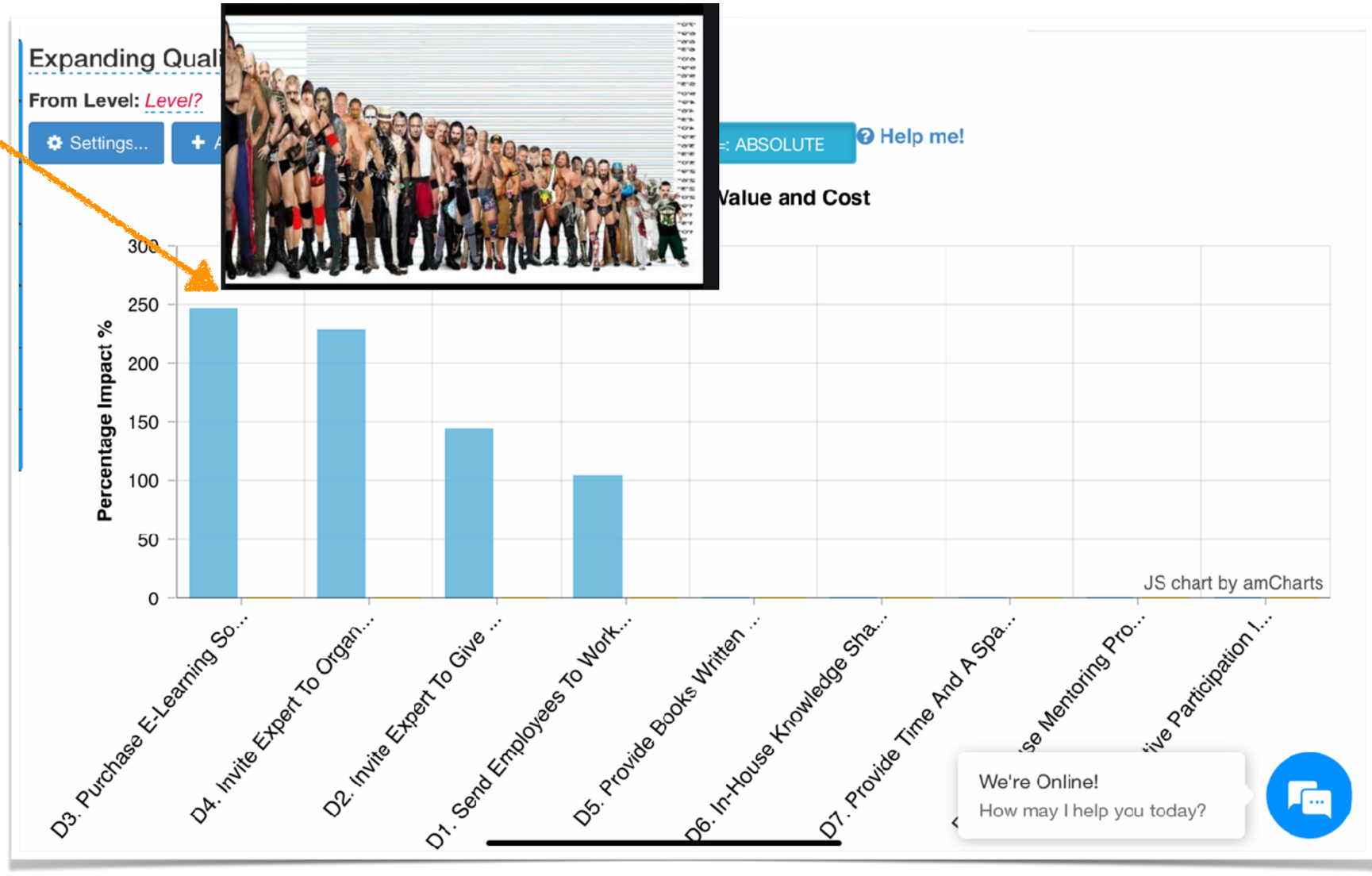
FIGURE: THIS BAR CHART IS EXTRACTED FROM THE TABLE AT LEFT, WE ASKED VALPLAN.NET TO SORT BY IMPACT TOTAL ON ALL VALUE REQUIREMENTS.

<u>LEFT-SIDE IS HEAD OF VALUE DELIVERY QUEUE</u>

THIS IS 'AUTOMATIC PRIORITIZATION OF DESIGN'. (SOURCE POLISH EXPORT PLAN)

Day 5:
Present Plans to Management, ask for approval to deliver the value.

- “Sub-Design D3 gives best overall stakeholder value delivery
- And takes 1 sprint week
- Shall we follow this value-delivery process?
  - Weekly ?
- Would you like a weekly report on incremental value delivery?
- Or would you prefer to look at costs and risks too?”



Part 6.

Decomposing designs, some more ideas

* The set of the top ten most-effective designs, can be viewed as ‘the architecture’ of your system.

* That means that each top-level design is going to consume approximately 10% of your total spend for the project.
  * What if a design is a disaster?
  * Well, you lose 10%,
  * and I think that is too much to gamble away
  * just to ‘test out a design’.
  * Not to mention the disturbance, caused by the failed design
    * being inserted in a live system, even on a small scale.

* So, your question to yourself is, “Am I feeling lucky today, punk?”

* Me? I do not like large and unnecessary risks.
  * Risks are unavoidable with large complex system renewal.
  * But I prefer small, easily-reversible risks.
  * So small, that nobody really is aware that my design has failed.
  * You can hide small failures more easily than large failures.

* After decades of experience I have landed on my personal standard
  * for ‘how big’ a design increment insertion,
  * that I will live with, on average.

* There are two ways of expressing this idea: ‘weekly’, and ‘2%’

* *Weekly* is very tempting
  * because there is a wonderful built-in way of coping with problems,
  * without being too late.
  * Work the weekend!
  * Then you can hide that fact that you were late.

Decompose down to 2% of resources Value increments

Not Construction increments

* But another way of expressing this is as a 2% increment of the budget (money, or calendar time to big deadline).

* Do 2% of spend.
  * Make sure it works and delivers real value. +V
  * Then, you have ‘permission’ to proceed,
  * and spend the next 2%.
  * Otherwise, why should I trust you with that money?

* I have discovered that you can always divide your design up, +V
  * into almost arbitrarily small increments:
  * no matter how large,
  * or how seemingly indivisible, the design idea might be.
  * You can <u>always find</u> Value delivery steps +V

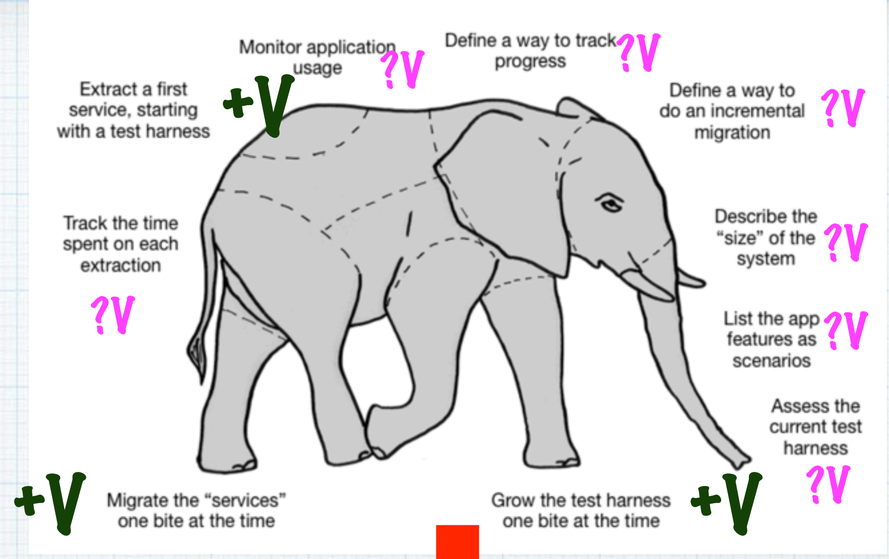
* Assuming I am right, and assuming I can teach you how to do this in practice, when all around you say it is ‘impossible’; your largest bet is now 2%, at one time.

* That is in simplistic terms, the most you can lose, if it does not work at all.
  * Dump it, and revert to where you were, before inserting any new design increment.

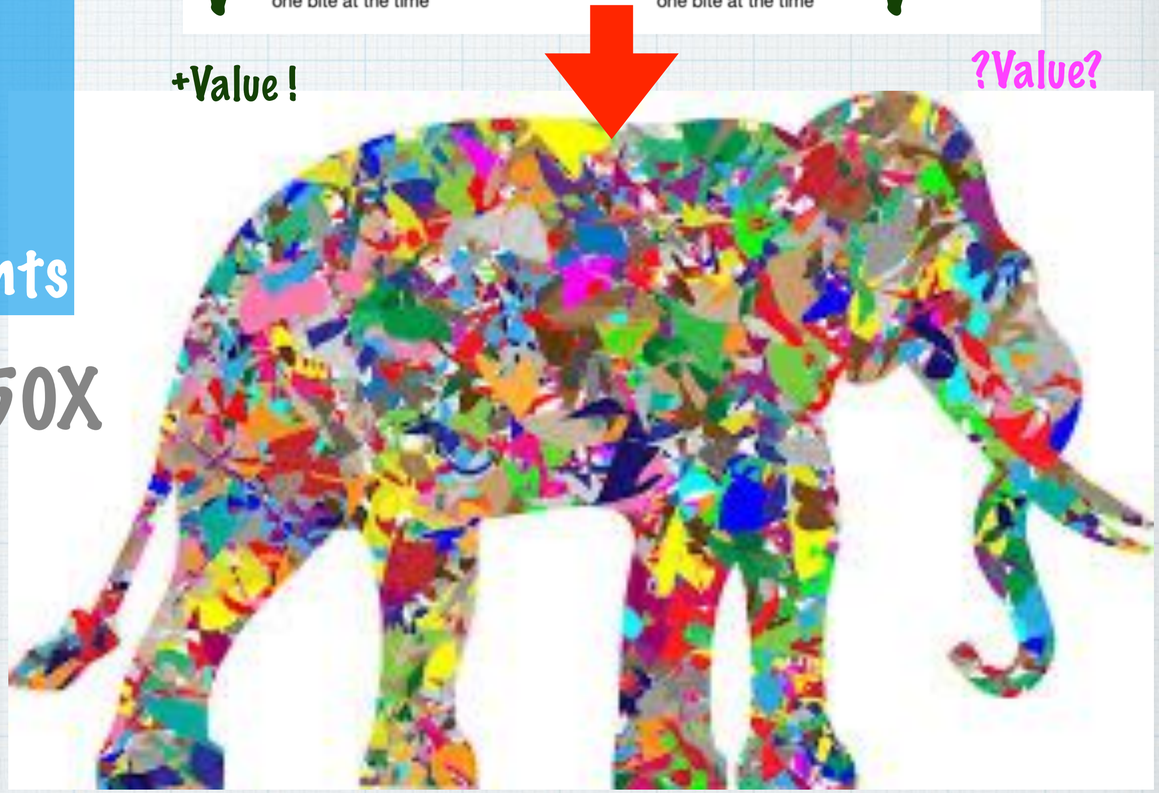
* Of course, part of your design has to be to plan for the possibility that the increment fails
  * : to make sure it cannot do much damage when it does fail,
  * and that we can reverse-out the design easily.

* You cannot responsibly assume all will be well, and that rapid painless reversal will not be necessary.

10X



50X



How do you eat an elephant ?

The Distinction Between a Value Implementation Step, and a Building/Acquisition Step.

+V ?V

- Let me emphasize a subtle point right away.

- I am not talking about *construction* steps: *about building* the system designs.

- I am talking about *implementation* steps: ?V
  - 'inserting design which is *available*
    - (already built or acquired),
  - into a real system', to see if we can get value. +V

- To handle this distinction we need these two concepts:
  - Frontroom (REAL VALUE): where we deliver available design to the stakeholder and the system. +V
  - Backroom (POTENTIAL VALUE) where we construct, acquire, validate, and make ready, a design-component for delivery and integration, to an existing system. ?V

- A simple analogy: ?V
  - it might take the Master Chef 5 hours to completely ready a dessert (backroom)
  - but when the restaurant guest orders dessert, it is ready, in the refrigerator in minutes (frontroom) +V

- This means that I am NOT claiming that I know how to decompose the acquisition and construction steps for a design, to a week or 2% every time.

- I am claiming that once the design is ready to install, I can find a limited cost-or-time way to deliver its value to stakeholders..

- Sometimes it is quite simple,
  - and well-known experimentation, and statistical-sampling ideas can be used.
  - For example: 'lets just do this design on 2% of the population,
  - and scale up gradually if it succeeds,
  - but stop and fix it first if it fails, before scaling up'.

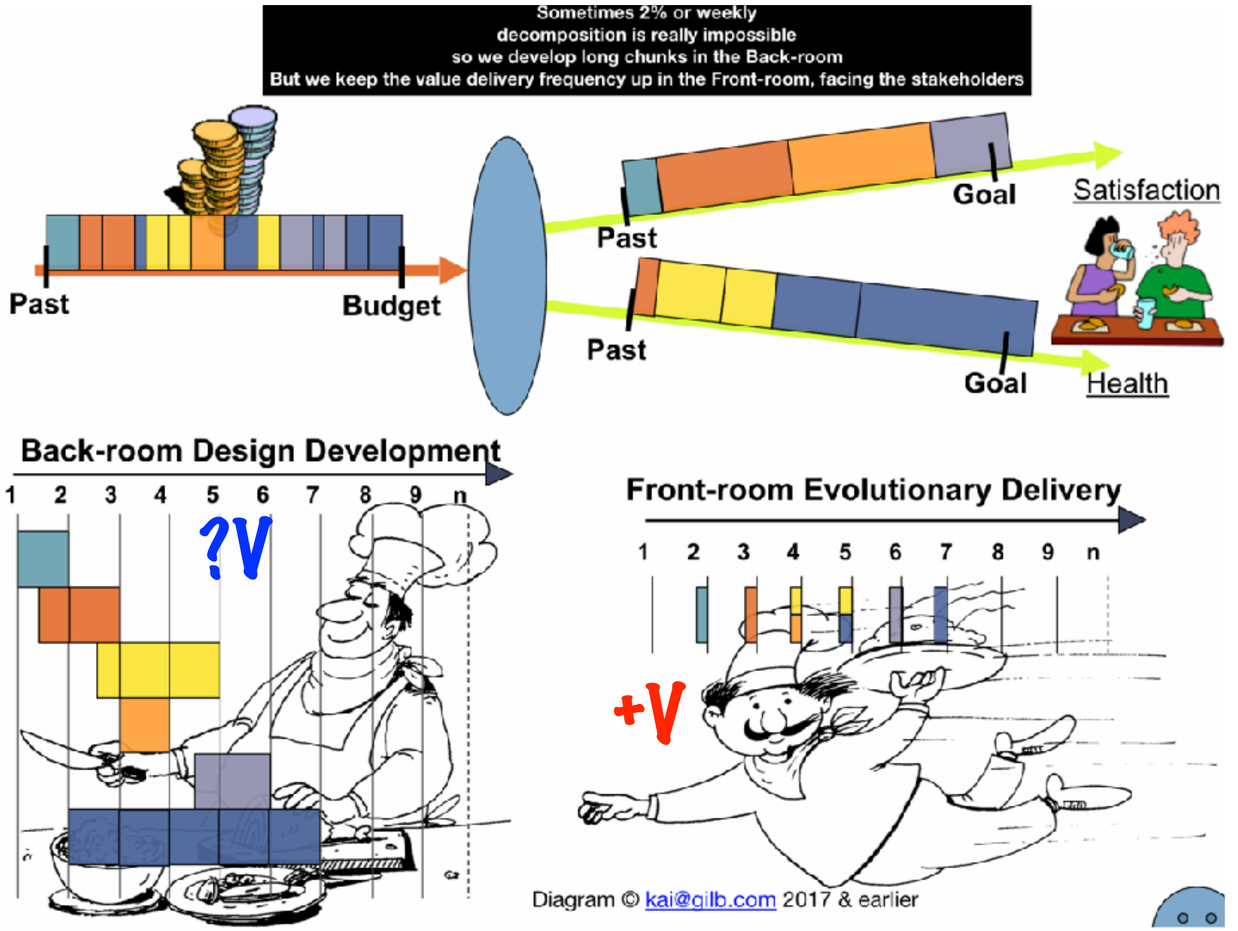


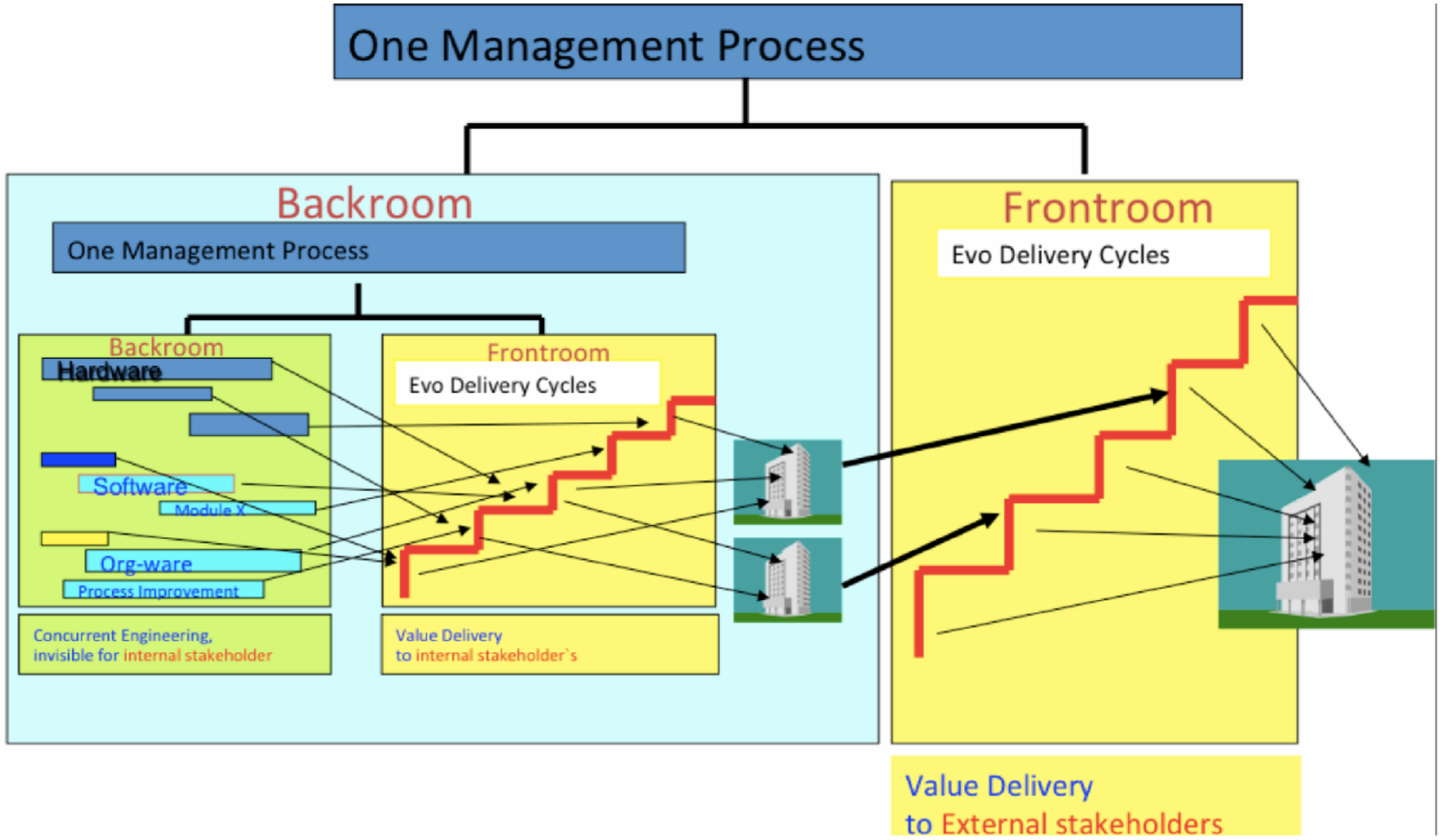
FIGURE: IT IS QUITE USEFUL (LOW RISK) TO DECOMPOSE YOUR VALUE DELIVERY INCREMENTS INTO SMALL, LOW RISK, DELIVERIES. IF THEY WORK, GREAT. IF NOT 'FULL REVERSE! AND LEARN!'
THIS IS NOT THE SAME PROBLEM AS DECOMPOSITION OF YOUR ARCHITECTURE FOR THE PURPOSE OF BIDDING, PURCHASING, BUILDING, VALIDATING AND QA APPROVAL (BACKROOM)
WHICH IS A PREREQUISITE TO THE INCREMENTAL VALUE DELIVERY STEPS.

Multiple Levels of Backroom/Frontroom
For large systems (Philips, NL)

FIGURE:
THE LARGER AND MORE COMPLEX YOUR SYSTEM,

THE MORE LIKELY THAT YOU WILL WANT TO USE MULTIPLE LEVELS OF BACKROOM AND FRONTROOM,

AT DIFFERENT LEVELS OF SUPPLIER AND MAJOR SUBSYSTEM.



Design Decomposition for Frontroom Value Delivery
Finding 'Agile Value Delivery Step' Designs

* It is unfortunately widespread that well-educated, smart professionals have a big problem with this 'decomposition into small steps'.

* For half the people I work with, they can simply 'decompose into independent value designs'
  * because I have asked for the decomposition in a clear way
  * (components must give *value*, and be *independently implementable*).

* The other half of my clients can be in strong denial.
  * "It cannot be decomposed! It is clearly impossible!
  * You don't understand our complex environment!
  * Nobody in our industry has even done this!"

Value stream Is always possible!

* Think of Elon Musk's Tesla production process mentioned above.
  * 20 production changes to the car, half hardware, half software, every week!
  * The other automakers don't do it that way!

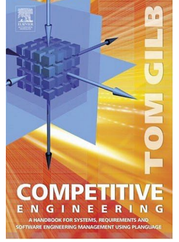
* Those 'design decomposition deniers' are normally wrong.
  * Using bad assumptions, bad logic,
  * and they do not know the decomposition methods, like those below.
  * They are usually making wrong assumptions
    * about what *kind* of decomposition I am asking for
      * (the 'build it' assumption they make, is wrong).

* This is very strong old style 'non-incremental' culture,
  * and we need to show respect for people going through this 'incremental value' learning process.
  * They are not stupid,
  * but they have *not* been educated and <u>worked</u> in a 'Value Driven Increment culture', at all.
  * They are still assuming some type of Big Bang Building culture

FIGURE:
HERE IS A REAL EXAMPLE OF DECOMPOSITION OF A TOP-LEVEL DESIGN IDEA ('ADVANCED CONGESTION CHARGES'),

INTO THE NEXT LEVEL OF INDEPENDENTLY IMPLEMENTABLE VALUE-DELIVERY DESIGNS.
(DETAIL OF EARLIER FIG. ABOVE)

If necessary each of these (D1.... D8) decomposed designs can be decomposed into even smaller delivery steps, for example D1 being implemented in specific London boroughs,
one at a time.



Gilb's Design Decomposition Principles

Inspired by CE book -2005, p. 314, and much-improved text detail (2019)

Based on many decades of decomposition practice.

1. **PERSIST: do not give up in 5 minutes, because you cannot think of useful decompositions.**
   1. **Assume there are many possibilities**
   2. **Think simple and small**
   3. **Don't eat drink or rest until you have solved the problem**
   4. **Ask some friends for good ideas, challenge them**
   5. **Ask what is the simplest thing we could do next week to improve that value level?**
2. **RULES: use these primary rules A. The decomposed design increment MUST deliver measurable planned value. B. The decomposed design increment must be implementable independently of all other potential increments (so that most value-effective increments can be done early, and not wait for others)**
   1. **These rules apply to all the top ten designs**
   2. **This work can be done in parallel**
   3. **First teams to get small increments can immediately put ideas into action**
   4. **If necessary, decompose more than once, to get to small 2% increments**
   5. **When you have several options, do the best value-to-cost options, first**
3. **OBSTACLES: it is OK to name and list potential obstacles and problems.**
   1. **But NOT OK to use problems as an excuse for inaction, for not finding good solutions.**
   2. **_Use_ these problem lists, and challenge the creative powers amongst you, to find solutions, to get rid of the obstacle 'by smart design'.**
   3. **Really good designers, can design their way past almost any obstacle**
4. **111111 DECOMPOSITION: be quite satisfied with _some_ real progress in value delivery, rather than _none_ (procrastination and excuses). Maybe just '1 function, 1 stakeholder, 1 value, 1 design, 1 week delivery cycle, 1% is better than 0%'.**
5. **MOTIVATION NUDGES: Do not divert your focus to the design ideas themselves, or the technical architecture. _re-focus on the values_.**
   1. **Sometimes you only need a small nudge, like a motivation, or a reward, to move the value to a better level.**
   2. **Nudges are simple and cheap, and you can experiment more easily, to find what works.**
6. **TEMPORARY SCAFFOLDING: Don't get stuck in the _new_ system design-increments**
   1. **Sometimes it is worth doing _temporary scaffolding_ to make progress.**
   2. **For example use old systems, temporary agreements, _borrowed_ systems or services.**
7. **ELEGANCE IS NOT THE POINT: Do not worry about elegance of design, it is only the value and cost results, that really matter.**
8. **THEY WANT RESULTS NOT DESIGN: Do not be afraid that stakeholders will not like your design itself:**
   1. **If it delivers the _value_ to them, they will like it.**
   2. **If no value, then even 'elegance' alone will not be of interest.**

A. DON'T WORRY GIVE VALUE: Do not procrastinate by worrying about 'what _might_ happen after we release this increment'.
- **A. You are not in full control of all potential future problems, anyway.**
- **B. Just focus on delivering value.**
- **C. Problems can be fixed, by design, as they occur.**
- **D. There may be a thousand things that _might_ go wrong, but only a few will really happen, so it pays off to wait, to sense them early, and deal with them quickly.**

B. DON'T GET _ANALYSIS PARALYSIS_; You cannot foresee everything that will happen when you release a design increment: don't try.
- **A. But make sure you can sense negative feedback quickly (design it)**
- **B. And make sure you can reverse the change as quickly, to limit damage (design it)**
- **C. Declare the negative feedback to be a scientific success in learning what NOT to do on a larger scale.**
- **D. Thank the participants profusely for their contribution, and ASK THEM what would be better!**
- **E. Tell them it is called 'Participatory Design Feedback Experiments'. And they are a vital part of the scientific process of making their own systems better. Be humble. They have really useful insights.**

C. REAL VALUE BREEDS COOPERATION: If you really constantly focus on delivering value to your stakeholders, then they will be quite forgiving of an occasional slip up.
- **A. They are not used to getting so much, real improvement, so fast.**
- **B. Make sure you seriously ask them, when you slip up, what you _should_ have done to improve _their_ value**
- **C. Make a big point of doing _something_ quite soon, to show you listened, and _can_ deliver**
- **D. That will teach the stakeholders that you are worth giving advice about value to.**

D. ACTION GIVES FACTS: If you just dive in, and do real stuff, you will more quickly learn what gives value, than if you discuss and plan in an office.
- **A. No 'modelling' beats the real thing, for trustworthy and useful insights in several dimensions**
- **B. Arguments based on observed real facts and experiences are more convincing**

E. HARNESS THE POSITIVE: Do early steps of a new design, before scaling up, with _positive, mature and supportive_ stakeholders.
- **A. That will help make things work better, after you scale up.**
- **B. Keep _local_ if you can, with insiders and friends.**
- **C. Let early adopters know that you really need their help and advice, to make early increments better, before you scale up.**
- **D. Credit their ideas in writing. In plans, and in 'the real thing'**

F. THE TROOPS KNOW WHERE THE BOOT PINCHES: If you have problems thinking of some valuable early small delivery steps, then maybe it is time to observe directly, and talk directly to your stakeholders.
- **A. Ask them for ideas. Their design ideas.**
- **B. Allocate time and energy to make some of those ideas come true; build _your_ credibility**
- **C. Build in (design in) simple feedback devices where stakeholders can at least display discomfort, and maybe suggest improvements in 'our' design**
- **D. This applies to get started early, but is also a continuous long-term design process**

G. DON'T REINVENT YOUR WHEEL: Do not be afraid to make use of the 'awful old system' initially, to get some real improvements: you can move the proven successful design ideas to the 'new system' when new components are incrementally in place.

H. OPEN-ENDED ARCHITECTURE: your basic architecture skeleton needs to be 'open ended'.
- **A. That means you need to plan specific architecture ideas which make it easy to change designs, when early design don't work well, into design which work better.**
  1. **There are many devices such as plug-and-play modularization, centralized architecture control , self-programming AI, and flexible contracts (not locked in, not over-specified in advance, pay-as-you-deliver?Value)**

Some practical organizational tactics to find good increments

* If some of your stakeholder's colleagues
  * have found good incremental solutions, and we can show these solutions, that might help the decomposition-resisters to get on board.

* Sometimes I (as a coach) manage to think of a simple example of incremental delivery for exactly their design.
  * It usually takes less than 5 minutes.
  * And I can then can ask the resisters, 'is that valid?'

* People can be *trained*
  * with a variety of specific practical principles,
  * to help them solve the problem,
  * of smaller incremental delivery.
  * See the above principles slide.

* I have observed that
  * viable solutions require a combination of *technical* expertise and domain expertise.
  * If these are not combined in one person,
    * you might need to make teams with this mix of skills.
  * Sometimes the technical expert cannot imagine the practical 'banking' or 'military' domain
  * and they do not dare guess at solutions which might be very simple for the domain expert to find.
  * And visa versa: the domain expert cannot alone imagine what the technical specialist in say 'Security' or Artificial intelligence' might be well aware of.

* Something my experience tells me, is
  * there are *always* good increment-options, which can be found same day.

* There do not seem to be known exceptions.
  * But some people are so confident that *they* are an exception,
  * that you can have fun, and make a bet they are wrong,
  * and can even be proven wrong today.

  * Easy money

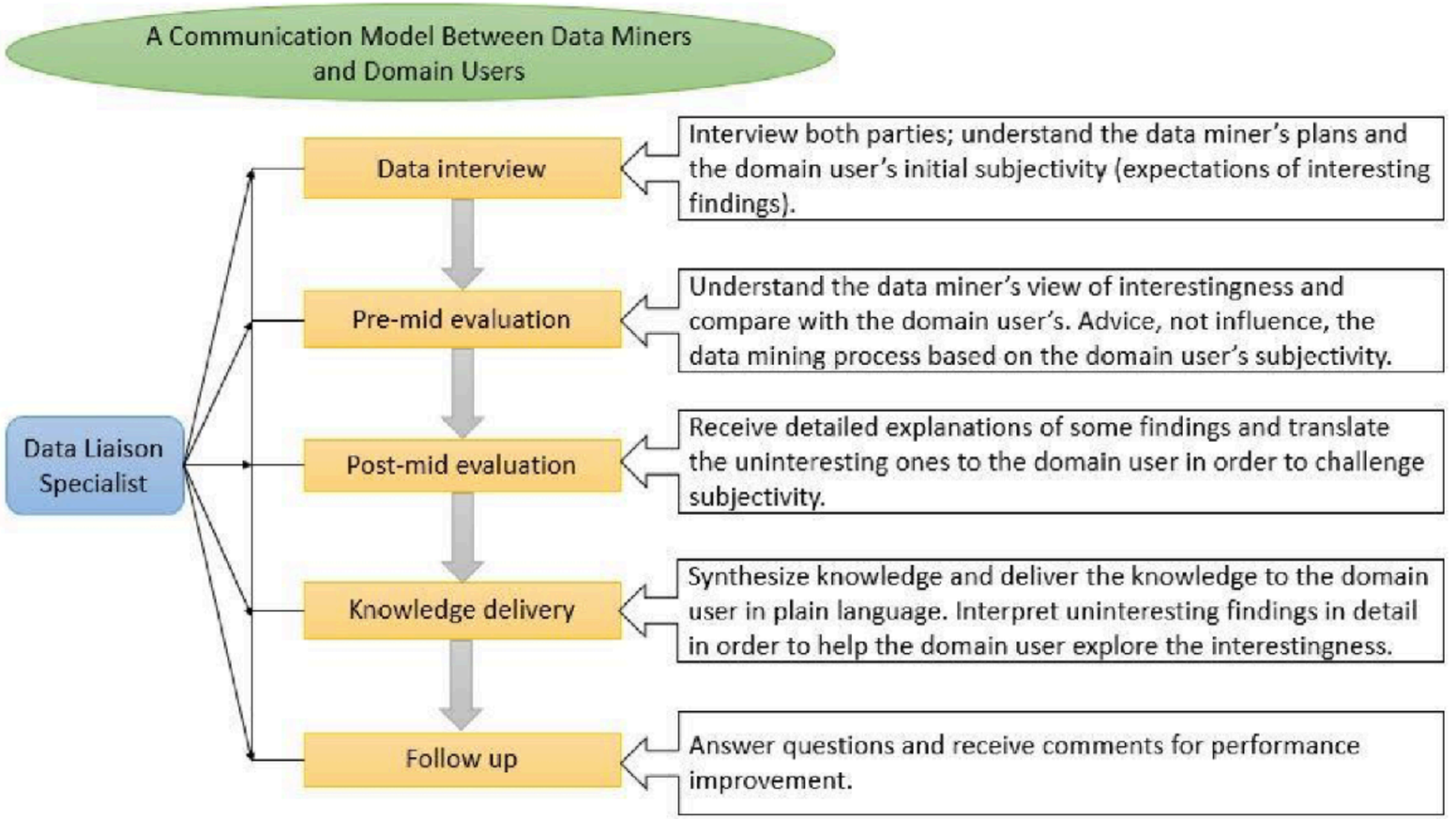
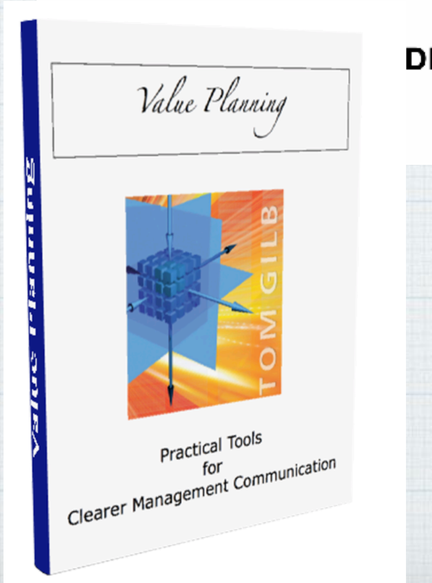


Figure 1. A communication model between data miners and domain users

The possibility of getting control of the value-to-cost ratio by decomposition; and then by prioritization of high-efficiency designs.

- By decomposing values, functions, and design components into a more-detailed set-of-things,
  - we can select a small fraction of the total product or system, to implement incrementally.
- This will give us real 'value delivery' to *some* stakeholders, for *some* value improvements.
  - But, at least as important, it will give us *measurable feedback* about a large number of factors,
  - so we can better see what we need to improve,
  - before we scale up, when implementing the rest of our agenda.

Value Decomposition Methods	Details		
Design Decomposition	Independently Implementable	Measurable Value Delivery ability	Urgent subset
Value Decomposition	Multiple Values	Scale Dimensions	Scale Dimension Sets
Deadline Decomposition	Date Due Start Date	Event Synchronization like 'Law in Force', 'Official Holiday'	Condition Synchronization like 'Interest Rates over 5%','Heat Wave'
Priority Type Decomposition	Wish (Desired Level)	Tolerable (Minimum Level)	Goal (Committed Level)



DIFFERENT TYPES OF DECOMPOSITION INTO COMPONENTS OR SUB-SETS

Source: Value Planning book, Chapter 5 Decomposition

Various decomposition methods,

so that we can learn reality early, and credibly,

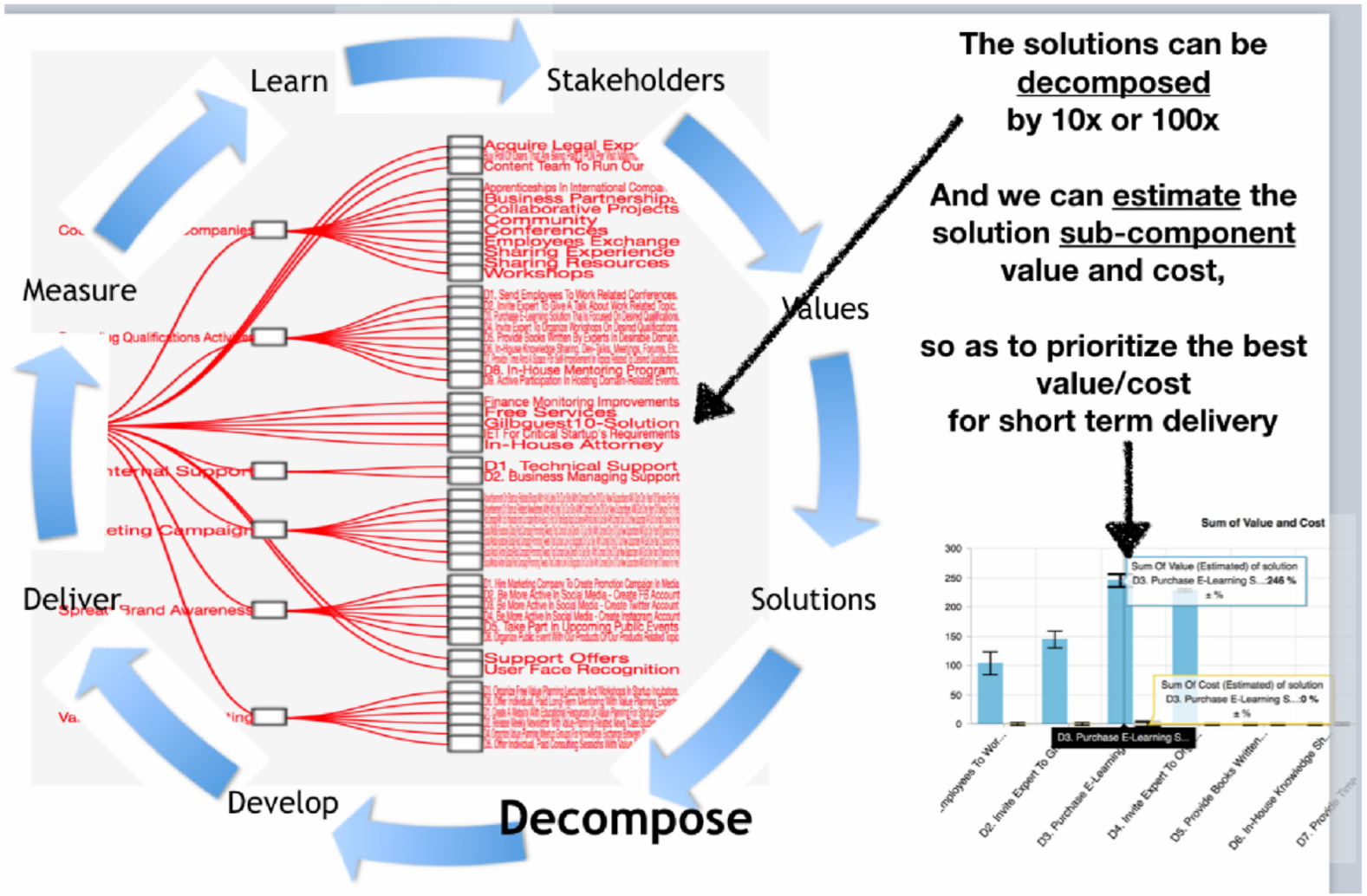
before scaling-up size or volume.

https://tinyurl.com/VPDecomposition

(for Value Planning detailed chapter)

Real Example of Decomposition of Technical Solutions to Control Cost-Efectivenss

Figure 1.83 : real planning example of decomposition and values/ costs estimation, of the decomposed design ideas.



24. HERE IS A REAL EXAMPLE OF MY CLIENT'S DECOMPOSITION OF DESIGN SOLUTIONS INTO INDEPENDENTLY IMPLEMENTABLE COMPONENTS. ON THE RIGHT LOWER CORNER IS AN ESTIMATE OF THE EFFECTIVENESS OF EACH SUB-COMPONENT, SO THAT WE CAN CHOOSE TO GO IMPLEMENT THE MOST EFFECTIVE ONE EARLY

Source: ADVANCED AGILE SOFTWARE ENGINEERING Turkey, April 2018, Polish Case

Figure 1.83 : real planning example of decomposition and values/costs estimation, of the decomposed design ideas.

These are the DECOMPOSITION conditions for sub-strategies that can actually can deliver value:

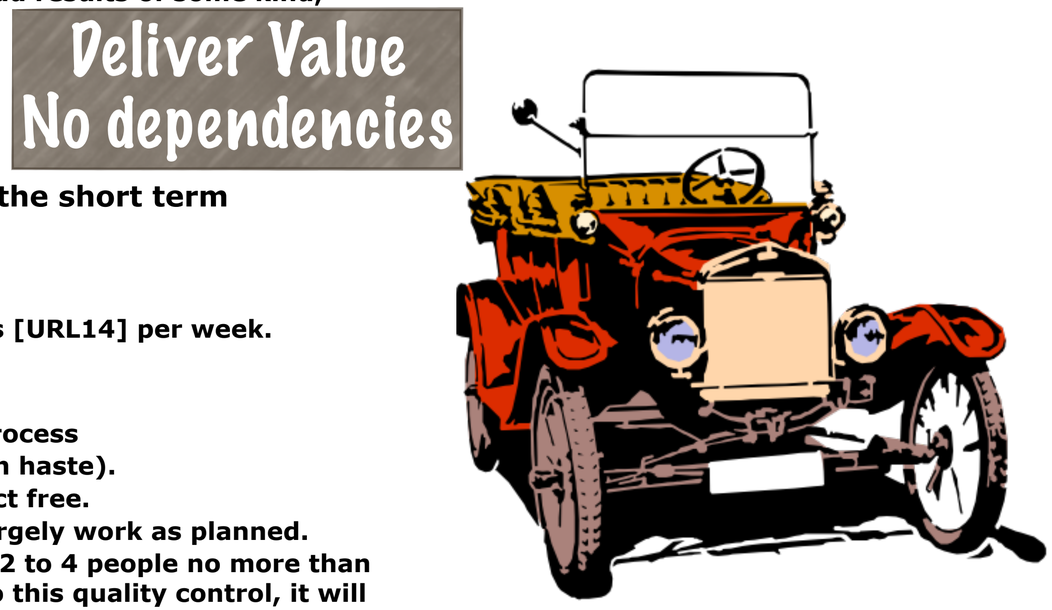
* Your expected primary **value improvement for this current cycle is estimated.**
  * Maybe with ± uncertainty, evidence, sources.
* **Side effects on other objectives, and on resources, are estimated**
* **We have plans for suitable Meters,**
  * ways of measuring what is happening in at least the primary value improvement objective.
* **There are no clear, unexpected, drastic negative side effects:**
  * such as people very disturbed, confused, getting bad results of some kind;
  * which disqualify the value delivery attempt,
  * even if 'our' value is delivered satisfactorily.
* **The value delivered is reasonably useful,**
  * even if not quite as good as initially estimated.
  * It should at least be clearly worth the cost.
* **We are prepared to learn, analyse, and change in the short term**
  * (during or immediately after this delivery cycle)
  * and the long term (in weeks ahead).
  * We plan the time to do this.
  * Even if it just a half-hour Defect Prevention Process [URL14] per week.
* **Critical plans are formally reviewed**
  * against our standards, and policies,
  * using a reasonably rigorous item by item review process
    * (not a meeting and a management sign-off in haste).
    * Plans do not Exit until they are close to defect free.
  * Plans do not have to be perfect, but they have to largely work as planned.
  * A review, for a weeks implementation, should take 2 to 4 people no more than one hour if organized well [URL23]. If you fail to do this quality control, it will normally cost you a lot more loss than that hour.

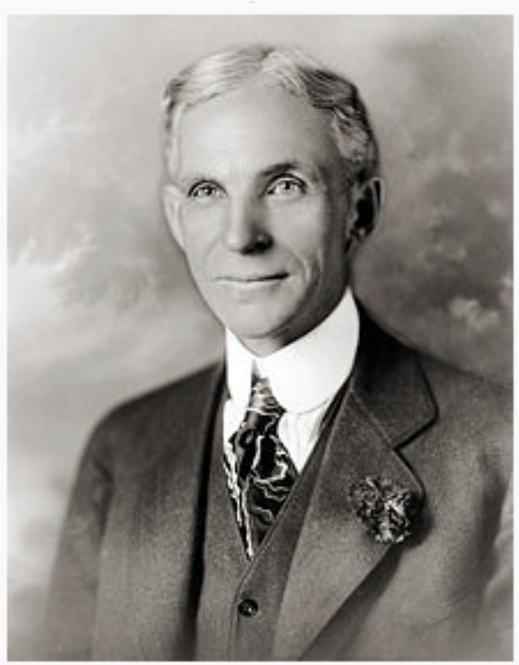
* Source Value Planning, https://tinyurl.com/VPDecomposition, VP 5.2

"Nothing is particularly hard if you divide it into small jobs."

"There are no big problems, just a lot of little problems"

"Obstacles are those frightful things you see when you take your eyes off your goal"
--Henry Ford, 1863-1947




Ford in 1919

Defining a structured scale of measure

[Scale Parameters] give 3 levels of decomposition

1. Scale
2. [Scale Parameter]
3. A Condition

% #Success Level# in

[Building]	[Resilience]	For	[Vulnerable]	in	[Situations]	to	[Shocks]
Economic Power	Avoiding		Poor		Individual Poverty		Climate
Health Power	Escaping		Physically Exposed		Family Poverty		Economic
Communications Ability	Resisting		Weak Health		Communal Poverty		Social
Recovery Speed	Recovering		No Network Fallback		National Poverty		[Environmental]
Relocation Capability	...		Insufficient Insurance		Epidemic Hit		- Earthquake - Flood - Avalanche - Fire
...			Insufficient Savings		...		
			...				...

Each condition can be defined

Each Scale Parameter is defined as a set of conditions

This graphic is designed by anna.maria.karlowska@gmail.com, 2019
For use in my book Sustainability Planning (about UN Sustainable Development Goals)
https://www.dropbox.com/sh/gc65fds9h0gv3cm/AABJvW4fwAnqVn25bPtY9bmia?dl=o

Defining a structured scale of measure

% #Success Level# in

[Building]	[Resilience]	For	[Vulnerable]	in	[Situations]	to	[Shocks]
Economic Power	Avoiding		Poor		Individual Poverty		Climate
Health Power	Escaping		Physically Exposed		Family Poverty		Economic
Communications Ability	Resisting		Weak Health		Communal Poverty		Social
Recovery Speed	Recovering		No Network Fallback		National Poverty		[Environmental]
Relocation Capability	...		Insufficient Insurance		Epidemic Hit		Earthquake Flood Avalanche Fire
...			Insufficient Savings		...		
			...				...

Each condition can be defined

Each Scale Parameter is defined as a set of conditions

[Scale Parameters] give 3 levels of decomposition

1. Scale
2. [Scale Parameter]
3. A Condition

Goal 42%

Building = Recovery Speed

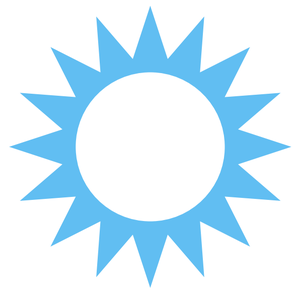
Resilience = Escaping

Vulnerable = Weak Health

Situation = Epidemic Hit

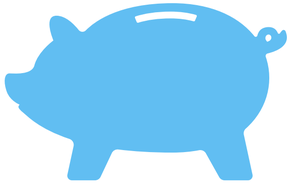
Shocks = Environmental.Fire

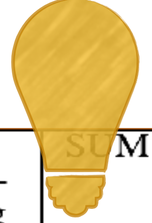
This graphic is designed by
anna.maria.karlowska@gmail.com, 2019
For use in my book Sustainability Planning
(about UN Sustainable Development Goals)
https://www.dropbox.com/sh/
gc65fds9h0gv3cm/
AABJvW4fwAnqVn25bPtY9bmia?dl=o



<——Design Decomposition——>

Value Decomp-osition





STRATEGIES ➔ OBJECTIVES	Technology Investment	Business Practices	People	Empow-erment	*Principles of IMA Management*	Business Process Re-engineering	SUM
Customer Service ?➔0 Violation of agreement	50%	10%	5%	5%	5%	60%	185%
Availability 90% ➔ 99.5% Up time	50%	5%	5-10%	0	0	200%	265%
Usability 200 ➔ 60 Requests by Users	50%	5-10%	5-10%	50%	0	10%	130%
Responsiveness 70% ➔ ECP's on time	50%	10%	90%	25%	5%	50%	180%
Productivity 3:1 Return on Investment	45%	60%	10%	35%	100%	53%	303%
Morale 72 ➔ 60 per mo. Sick Leave	50%	5%	75%	45%	15%	61%	251%
Data Integrity 88% ➔ 97% Data Error %	42%	10%	25%	5%	70%	25%	177%
Technology Adaptability 75% Adapt Technology	5%	30%	5%	60%	0	60%	160%
Requirement Adaptability ? ➔ 2.6% Adapt to Change	80%	20%	60%	75%	20%	5%	260%
Resource Adaptability 2.1M ➔ ? Resource Change	10%	80%	5%	50%	50%	75%	270%
Cost Reduction FADS ➔ 30% Total Funding	50%	40%	10%	40%	50%	50%	240%
SUM IMPACT FOR EACH SOLUTION	***482%***	***280%***	***305%***	***390%***	***315%***	***649%***	
Money % of total budget	15%	4%	3%	4%	6%	4%	
Time % total work months/year	15%	15%	20%	10%	20%	18%	
SUM RESOURCES	***30***	***19***	***23***	***14***	***26***	***22***	
BENEFIT/RESOURCES RATIO	***16:1***	**14:7**	**13:3**	**27:9**	**12:1**	**29.5 : 1**	

Impact Estimation: Value-for-Money Delivery Table

© Gilb.com

- Here are
  - some decomposition concepts
- **1 1 1 1 1 1 1 Unity**
  - **1% increase at least**
  - **1 stakeholder**
  - **1 quality/value**
  - **1 week delivery cycle**
  - **1 function focus**
  - **1 design used**

STRATEGIES ➔ OBJECTIVES	Technology Investment	Business Practices	People	Empow-erment	*Principles of IMA Management*	Business Process Re-engineering	SUM
Customer Service ?➔0 Violation of agreement	50%	10%	5%	5%	5%	60%	185%
Availability 90% ➔ 99.5% Up time	50%	5%	5-10%	0	0	200%	265%
Usability 200 ➔ 60 Requests by Users	50%	5-10%	5-10%	50%	0	10%	130%
Responsiveness 70% ➔ ECP's on time	50%	10%	90%	25%	5%	50%	180%
Productivity 3:1 Return on Investment	45%	60%	10%	35%	100%	53%	303%
Morale 72 ➔ 60 per mo. Sick Leave	50%	5%	75%	45%	15%	61%	251%
Data Integrity 88% ➔ 97% Data Error %	42%	10%	25%	5%	70%	25%	177%
Technology Adaptability 75% Adapt Technology	5%	30%	5%	60%	0	60%	160%
Requirement Adaptability ? ➔ 2.6% Adapt to Change	80%	20%	60%	75%	20%	5%	260%
Resource Adaptability 2.1M ➔ ? Resource Change	10%	80%	5%	50%	50%	75%	270%
Cost Reduction FADS ➔ 30% Total Funding	50%	40%	10%	40%	50%	50%	240%
SUM IMPACT FOR EACH SOLUTION	***482%***	***280%***	***305%***	***390%***	***315%***	***649%***	
Money % of total budget	15%	4%	3%	4%	6%	4%	
Time % total work months/year	15%	15%	20%	10%	20%	18%	
SUM RESOURCES	***30***	***19***	***23***	***14***	***26***	***22***	
BENEFIT/RESOURCES RATIO	***16:1***	**14:7**	**13:3**	**27:9**	**12:1**	**29:5**	

© Gilb.com

Part 7.

Risk management for value design

Risk in my opinion, means risk of things ‘going wrong’.

I have concluded that everything in a design process, every detail, is somehow related to risk. That means everything thus far, is *some* form of risk management.

For example:

1. Quantifying Value: reduces risk of misunderstanding of design requirements
2. Estimating design impacts on a value table, reduces the risk of putting too much effort too early into low value, high cost, dubious design ideas
3. Decomposition of design ideas into smaller incrementally implementable-in-real existing systems reduces risks of designs not working, and of value delivery being delayed
4. Open-ended architecture reduces risks of surprising new design-changes costing too much to implement.

“Risk comes from not knowing what you're doing”
— Warren Buffett (1930-)

Some Risk Postulates, and Truisms, for Design and Designers

1. Risk is inherent in all processes, all designs, and all planning elements.
2. Risk is present whether we acknowledge it explicitly or not.
3. Risk can be analyzed and understood, as a function of the design and its history.
4. Risk mitigation can be planned, but it might cost more than it is worth.
5. You can design risk *mitigation* to be worth more than it costs.
6. Your designs can consciously reduce specific risks.
7. Your design requirements can motivate designers to design, to reduce specific risks.
8. Total elimination of all risk is not feasible or economic: but systematic risk reduction and control is possible by design.
9. A designer can be trained to specialise in dealing with risk.
10. Standards for design specification, and for designs themselves, can be enforced, to help manage risks

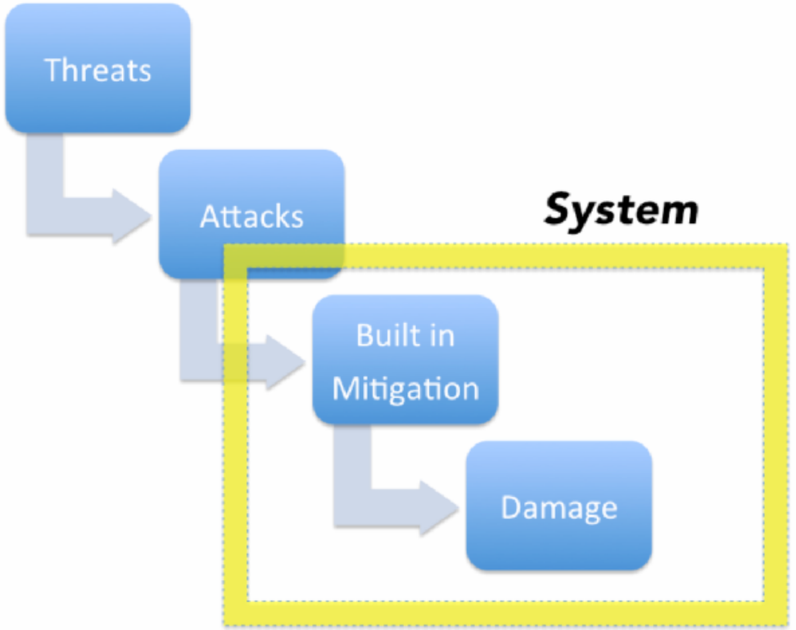
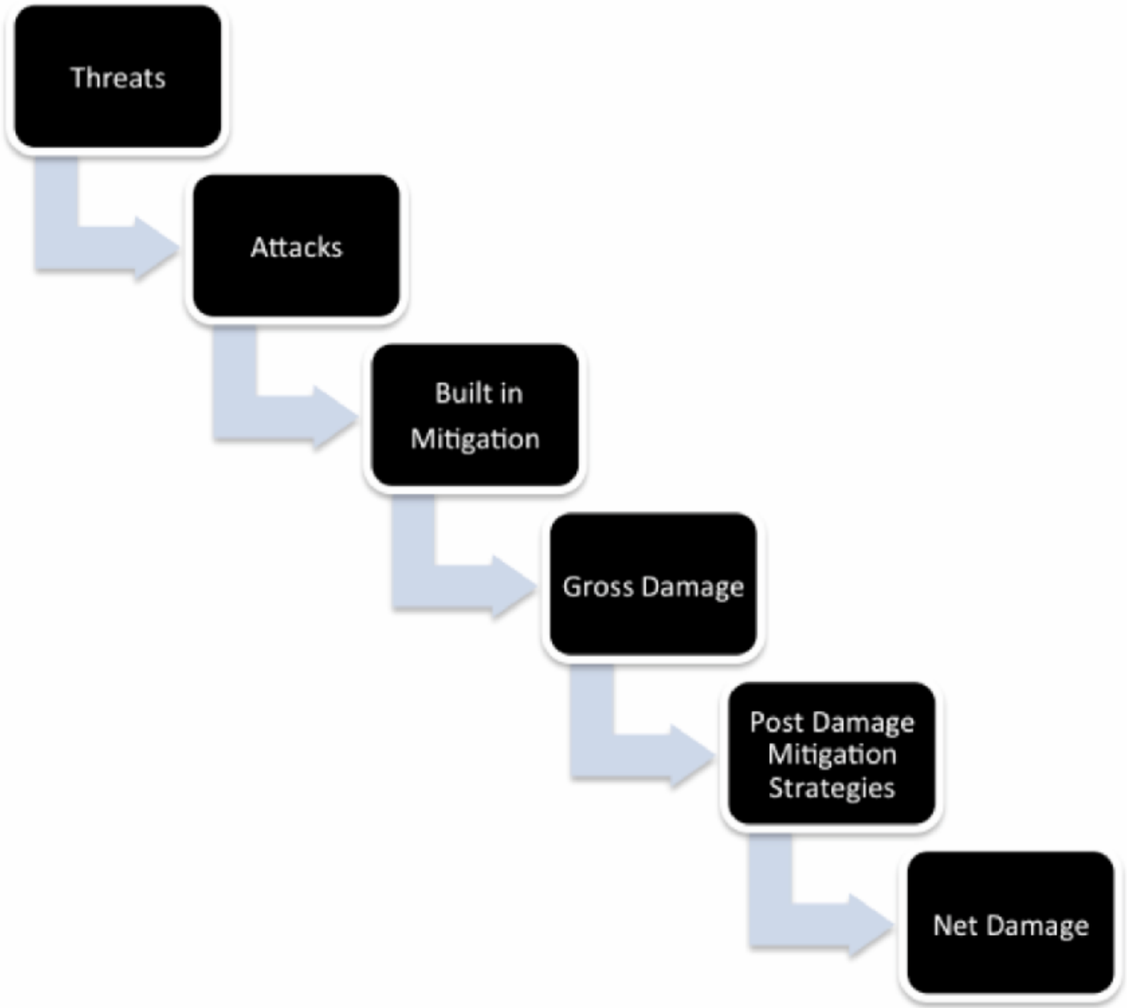


Figure 7.2 The risk environment. Threats may turn into real attacks, which get confronted with our planned mitigation. If the mitigation plan is unsuccessful, damage results, of various kinds. Notice that 'built-in to the system mitigation' is one type of mitigation. We can also mitigate risks at earlier stages, such as in planning, contracting, building trials, tests etc.

Risk Design Principles,
in a Value-Focussed, Incremental, Design Delivery, Environment.

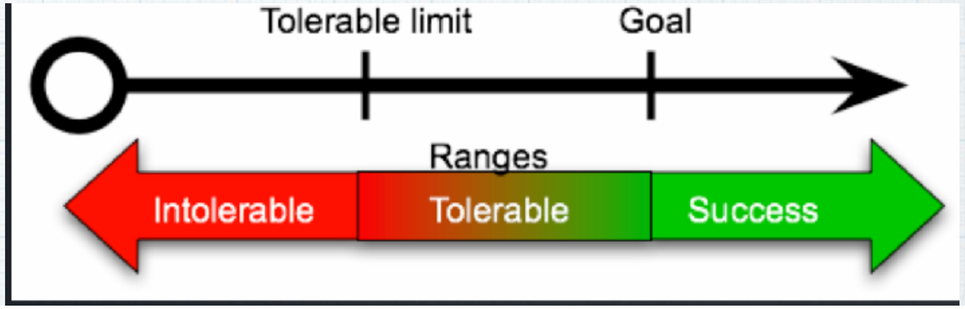
1. **RISK DIMENSIONS: Values, including qualities; and resources, are the Risk Dimensions because we can s some negative areas on their 'scales of measure'.**
2. **RISK REQUIREMENTS: The designer must be made aware of these risk areas in the Design Requirements**
3. **DESIGN FOR RISK: new method or process for design, which does not *consciously* deal with quantified va resources, is useless for risk management, and doomed to 'fail to avoid' risks.**
4. **MITIGATION DESIGNER: A competent designer can *consciously specify a design*, so as to *reduce* risks, in all Risk Dimensions.**
5. **INCOMPETENT DESIGNERS: Incompetent designers do not *consciously* design to avoid defined requirem**
6. **REQUIREMENTS SPECIFY RISKS: Value and Resource requirement specifications, can *explicitly*, and *num* inform a designer, about areas considered to be negative (example 'Intolerable' level), and not only by mea requirement level, but of course the *specific Scalar attribute conditions* the level applies to**
7. **DYNAMIC DESIGN TO COST: The designers responsibility is not complete when initial or approved design specifications are done, but they have a responsibility to follow *implementation progress*, and to *take actio* improve the design if value or cost measurements are not at targets, and especially if they are in Intolerabl territory.**
8. **ETERNAL DESIGN RESPONSIBILITY: The responsibility of the Life-Cycle Designer is not finished when the initially works well, in practice: it must continue to work well when additional designs are incremented, dur development, and later - any time in the system life-cycle.**
9. **FOOLPROOF DESIGN: The designer is responsible for 'foolproof detail' in design specification. When the spec is followed correctly, and the designers intent, for Values and Resources, is largely achieved, there ar excuses that the design implementor has 'misunderstood' the specification.**
10. **DESIGN QC EXIT: A design specification can be quality-controlled, using Specification Quality Control, ar suitable Rules (design quality standards); and not released (Exited) unless the defect level is low enough (0 defects per 300 words)**



Part 8.

Design Prioritisation.

There are some general principles of value prioritization that can sum up the discipline of Value-Driven Design



1.VALUE BASED: Priority is *based* on Stakeholder Values.

2. VARIED PRIORITY: Stakeholder values *vary* from time to time, and thus so does priority.

3. NUMERIC PRIORITY SIGNALS: A Value priority can be stated *numerically*, as a constraint or target requirement-level.

4. CONDITIONS PRIORITY: A requirement level should include *scale-parameter conditions*: when, where, who, why attributes: this implies that a priority statement can be very specific

5. GENERAL PRIORITY: We can state *quite general* priority policies, which also apply to the quite-specific levels and conditions for a requirement:

for example *'Prioritize Maximum Critical-Value-Set Impact, in relation to Minimum Critical Resources Set, with regard to Uncertainty, Credibility, and Conditions'.*

6. VALUE FOR MONEY: In general, some variation of 'value for money' will be the dominant priority

7. EXCEPTIONAL PRIORITY: In special circumstances, a single delivery step, might prioritize a narrow value such as,

***'Do not use any money for your next design. We have used up the budget!'* Notice this has dramatic 'design' consequences. You will have to prioritize designs that cost nothing**

.8. SURVIVAL THEN SUCCESS:The defined constraint-and-target requirement-levels imply strong priority ideas. In simple terms survival first (above Tolerable level, for all critical values), then 'Success' (reach all Goal levels before using resources to go beyond any Goal level).

9. RESOURCE PRIORITY: The resource budgets also imply strong prioritization: 'do not design or implement, so that budgets are violated, or threatened, too early'.

As a simple example, the designer cannot use most of the Capital budget on a single design component, among many other design components to come. The designer has to both minimize unnecessary use of resources, and a *set* of designers (usability designer, security designer, style designer, performance designer, for example) have to co-operate, in using a limited budgeted set of resources (mainly time and money). That means that all designers need to always have some resources estimates, of their suggested design specifications, as opposed to 'no idea'.

10. COMPLEX LOGICAL PRIORITY: In summary you can see that determining priority is potentially non-simplistic. It can be both objective, subjective, numeric, and computed for us, for a given policy.

Priority Signals

Current Status	Improvements		Survey Engine .NET		
Units	Units	%	Past	Tolerable	Goal
			Backwards.Compatibility (%)		
83,0	48,0	80,0	0	85	95
0,0	67,0	100,0	7	0	0
			Generate.WI.Time (small/medium/large seconds)		
4,0	59,0	100,0	3	8	4
10,0	397,0	100,0	07	100	10
94,0	2290,0	103,9	384	500	180
			Testability (%)		
10,0	10,0	13,3		100	100
			Usability.Speed (seconds/user rating 1-10)		
774,0	507,0	51,7	81	600	300
5,0	3,0	60,0		5	7
			Runtime.ResourceUsage.Memory		
0,0	0,0	0,0		?	?
			Runtime.ResourceUsage.CPU		
3,0	35,0	97,2	8	3	2
			Runtime.ResourceUsage.MemoryLeak		
0,0	800,0	100,0	00	0	0
			Runtime.Concurrency (number of users)		
1350,0	1100,0	146,7	50	500	1000
			Development resources		
64,0			0		84

Figure 6.7 . Incremental Value Tracking at Confirmit.

Value Planning (VP) book, Chapter 6 Prioritization.
https://www.dropbox.com/sh/34llx1a7ckyagxl/AAAOpDzSxN5WmoP9lOKROMpca?dl=0

Part 9.

Dynamic Design to Requirements (which includes some priority stuff)

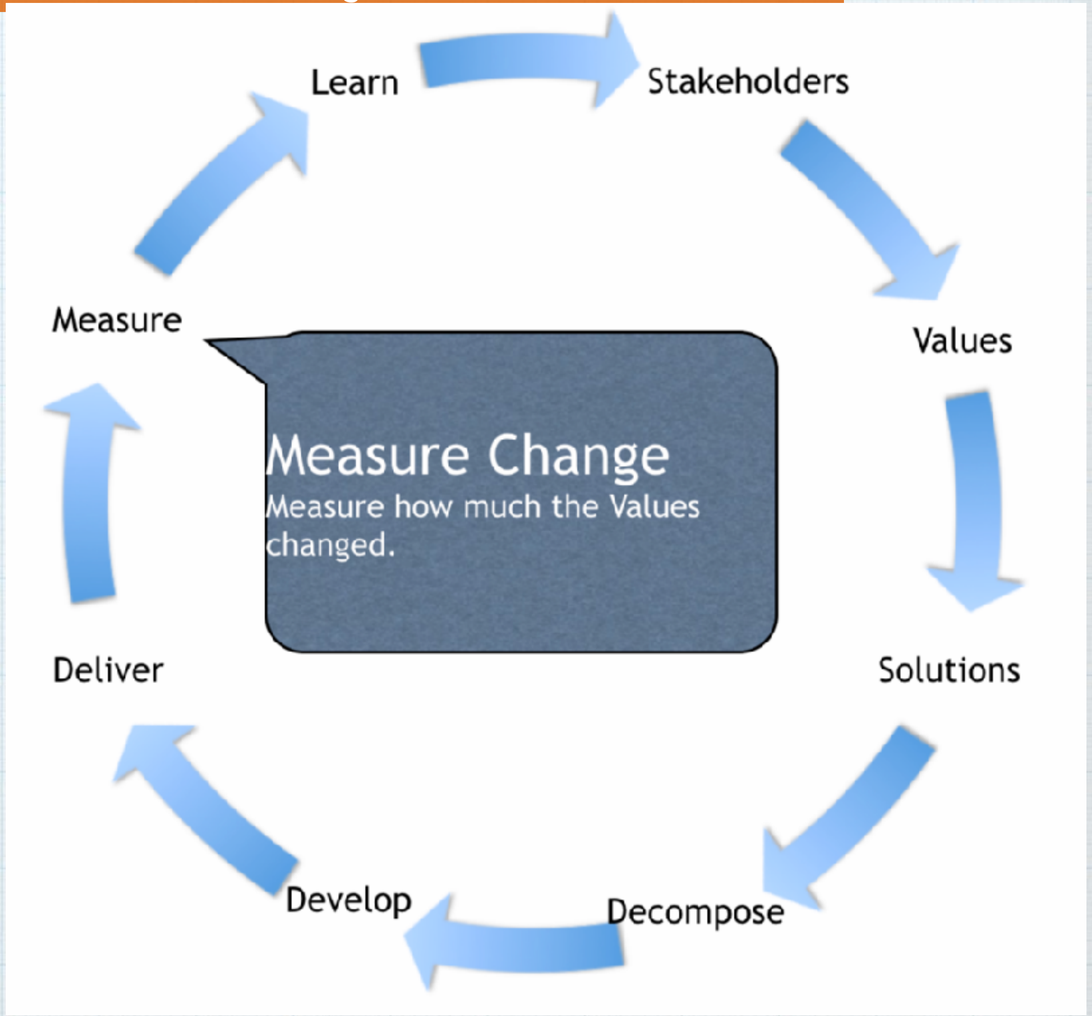
Here are some basic definitions:

Design To Cost (D->C): a well-established engineering concept. You can find designs to meet a given cost requirement.

Design To Value (D->V): the same concept as Design to Cost, except the design process is directed towards meeting a *Value* (including any *quality*) Requirement Level of Performance.

Design To Requirements (D->R): the combination, perhaps simultaneously in a single delivery cycle, of attempting to design to any set of both 'Value Requirement Levels', and 'Cost Requirement Levels'.

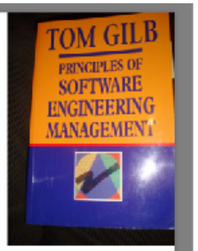
Dynamic Design to Requirements (DD->R): A cyclical Design process, to meet any set of Value and/or Cost requirements, by using measurement, after incremental design-implementation, comparing with requirements, predicting future cost and value levels, and re-designing if necessary, to better reach the requirements.



Principles of 'Dynamic Design to Requirements'

1. It is easier to *measure* accurately than to *predict* accurately.
2. It is faster to *adjust* a bad design, than to figure out the *right* design to begin with.
3. Even small systems can be complex, so frequent incremental measurement of progress is smart.
4. Everything can interact with everything else, now and in future, so incremental measurement of critical factors is the only safe way to keep score.
5. It is better to 'design to a cost' you require, than to 'estimate a design cost' you can't afford.
6. It is easier to 'design to a stakeholder *value* level' than to just *hope* your design will meet that level.
7. Estimation of design impacts, helps sort out 'probably good design' early.
8. Only *measurement* of design value-delivery and design-costs will confirm you got it right.
9. Designs interact with all other system designs, past and future, compounded by changing stakeholder needs - and consequent re-design.
10. Small incremental steps of design, greatly simplify understanding of complex design.

Robert E. Quinnan (-2015): IBM FSD Cleanroom *Dynamic Design to Cost*



Quinnan describes the process control loop used by IBM FSD to ensure that cost targets are met.

'Cost management. . . yields valid cost plans linked to technical performance. Our practice carries cost management farther by introducing design-to-cost guidance. Design, development, and managerial practices are applied in an integrated way to ensure that software technical management is consistent with cost management. The method [illustrated in this book by Figure 7.10] consists of developing a design, estimating its cost, and ensuring that the design is cost-effective.' (p. 473)

He goes on to describe a design iteration process trying to meet cost targets by either redesign or by sacrificing 'planned capability.' When a satisfactory design at cost target is achieved for a single increment, the 'development of each increment can proceed concurrently with the program design of the others.'

'Design is an iterative process in which each design level is a refinement of the previous level.' (p. 474)

It is clear from this that they avoid the big bang cost estimation approach. Not only do they iterate in seeking the appropriate balance between cost and design for a single increment, but they iterate through a series of increments, thus reducing the complexity of the task, and increasing the probability of learning from experience, won as each increment develops, and as the true cost of the increment becomes a fact.

'When the development and test of an increment are complete, an estimate to complete the remaining increments is computed.' (p. 474)

Source: Robert E. Quinnan, 'Software Engineering Management Practices', IBM Systems Journal, Vol. 19, No. 4, 1980, pp. 466–77

This text is cut from Gilb: The Principles of Software Engineering Management, 1988

Copyright gilb.com 2019.

‘Cleanroom’
An advanced software development process
(‘perfect’ complex project management)

Dr. Harlan Mills, IBM Federal Systems Division

A real ‘agile’ software engineer pioneer

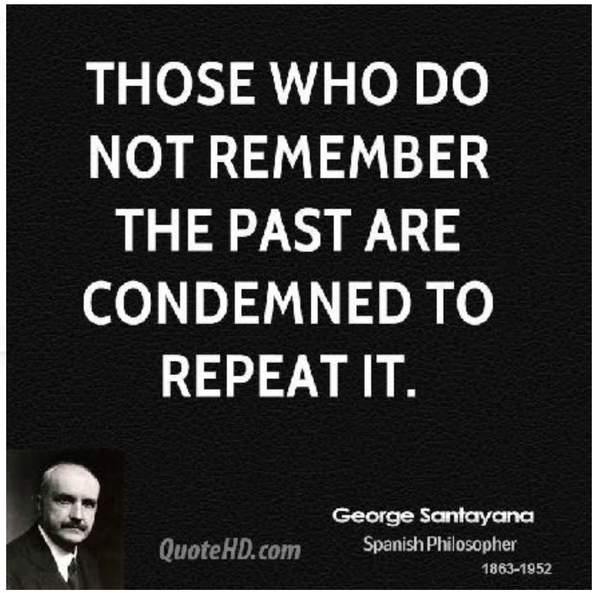
https://trace.tennessee.edu/utk_harlan/?utm_source=trace.tennessee.edu/utk_harlan/5&utm_medium=PDF&utm_campaign=PDFCoverPages

500 Collected works of Harlan Mills

Reprinted from

IBM Systems Journal

Volume Nineteen | Number Four | 1980



The management of software engineering Part I: Principles of software engineering

by H. D. Mills

http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1004&context=utk_harlan

"The first guarantee of quality":

Designing Qualities into a system
Is much *simpler*
than trying to get qualities, by *testing* them in**

Harlan Mills

- **"The first guarantee of quality in design is in well-informed, well-educated, and well-motivated designers.**
- **Quality must be built into designs, and cannot be inspected in or tested in.**
- **Nevertheless, any prudent development process verifies quality through inspection and testing.**
- **Inspection by peers in design, by users or surrogates, by other financial specialists concerned with cost, reliability, or maintainability not only increases confidence in the design at hand, but also provides designers with valuable lessons and insights to be applied to future designs.**
- **The very fact that designs face inspections motivates even the most conscientious designers to greater care, deeper simplicities, and more precision in their work."**

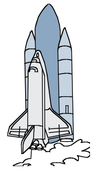
http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1004&context=utk_harlan

** testing qualities into a system, is impossibly **complex**, and takes infinite time.

In the ‘Cleanroom Method’, developed by IBM’s Harlan Mills (IBM SJ No. 4/1980) they reported:

- **“Software Engineering began to emerge in FSD” (IBM Federal Systems Division, from 1996 a part of Lockheed Martin Marietta) “some ten years ago [Ed. about 1970] in a continuing evolution that is still underway:**
- **Ten years ago general management expected the worst from software projects – cost overruns, late deliveries, unreliable and incomplete software**
- **Today [Ed. 1980!], management has learned to expect on-time, within budget, deliveries of high-quality software. A Navy helicopter ship system, called LAMPS, provides a recent example. LAMPS software was a four-year project of over 200 person-years of effort, developing over three million, and integrating over seven million words of program and data for eight different processors distributed between a helicopter and a ship in 45 incremental deliveries [Ed. Note 2%!]s. Every one of those deliveries was on time and under budget**

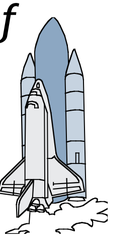
- **A more extended example can be found in the NASA space program,**



- **- Where in the past ten years, FSD has managed some 7,000 person-years of software development, developing and integrating over a hundred million bytes of program and data for ground and space processors in over a dozen projects.**
- **- There were few late or overrun deliveries in that decade, and none at all in the past four years.”**

In the Cleanroom Method, developed by IBM's Harlan Mills (1980) they reported:

- *"Software Engineering began to emerge in FSD" (IBM Federal Systems Division, from 1[redacted] in a c[redacted]*
- *T[redacted] cost overruns, late deliveries, unreliable and incomplete software*
- *Today [Ed. 1980!], management has learned to expect on-time, within budget, deliveries of high-quality software. A Navy helicopter ship system, called LAM[redacted] provides a recent example. LAMPS software was a four-year project of over 200 person-years of effort, developing over three million, and integrating over seven million words of program and data for eight different processors distributed betwe[redacted] 2%!]s. Every one of[redacted]*
- *A more[redacted]*
- *- Wher[redacted]rs of softwa[redacted]ion bytes of progra[redacted]ects.*
- *- There[redacted]t all in the past fo[redacted]*



in 45 incremental deliveries

Agile!

were few late or overrun deliveries in that decade, and none at all in the past four years

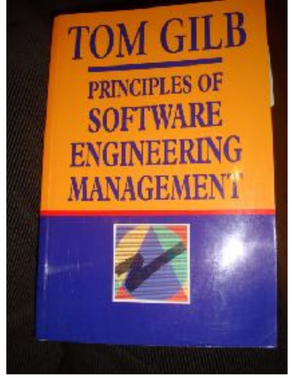
wow! *normal* agile fails 19% (Scrum) to 40%: Jeff Sutherland

Mills on Design-to-Cost
(call it 'agile', incremental, design)

- "To meet cost/schedule commitments based on imperfect estimation techniques, a software engineering manager must adopt a **manage-and-design-to-cost/schedule process**.
- That process requires a **continuous and relentless rectification** of design objectives with the cost/schedule needed to achieve those objectives."
- in IBM sj 4 80 p.420

See Quinnan's flow chart "Design to Cost" below For process detail

Robert E. Quinnan (Cleanroom Architect): IBM FSD Cleanroom *Dynamic Design to Cost*



Quinnan describes the process control loop used by IBM FSD to ensure that cost targets are met.

'Cost management. . . yields valid cost plans linked to technical performance. Our practice carries cost management farther by introducing design-to-cost guidance. Design, development, and managerial practices are applied in an integrated way to ensure that software technical management is consistent with cost management. The method [illustrated in this book by Figure 7.10] consists of developing a design, estimating its cost, and ensuring that the design is cost-effective.' (p. 473)

He goes on to describe a design iteration process trying to meet cost targets by either redesign or by sacrificing 'planned capability.' When a satisfactory design at cost target is achieved for a single increment, the 'development of each increment can proceed concurrently with the program design of the others.'

'Design is an iterative process in which each design level is a refinement of the previous level.' (p. 474)

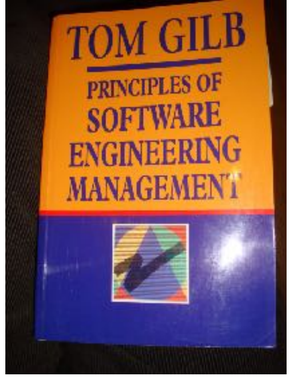
It is clear from this that they avoid the big bang cost estimation approach. Not only do they iterate in seeking the appropriate balance between cost and design for a single increment, but they iterate through a series of increments, thus reducing the complexity of the task, and increasing the probability of learning from experience, won as each increment develops, and as the true cost of the increment becomes a fact.

'When the development and test of an increment are complete, an estimate to complete the remaining increments is computed.' (p. 474)

Source: Robert E. Quinnan, 'Software Engineering Management Practices', IBM Systems Journal, Vol. 19, No. 4, 1980, pp. 466~77
This text is cut from Gilb: The Principles of Software Engineering Management, 1988

Copyright Tom@Gilb.com

Initial design and cost estimates are incrementally reviewed and improved



Quinnan describes the process control loop used by IBM FSD to ensure that cost targets are met.

'Cost management. . . yields valid cost plans linked to technical performance. Our practice carries cost management farther by introducing design-to-cost guidance. Design, development, and managerial practices are applied in an integrated way to ensure that software technical management is consistent with cost management. The method [illustrated in this book by Figure 7.10] consists of developing a design, estimating its cost, and ensuring that the design is cost-effective.' (p. 473)

He goes on to describe a design iteration process trying to meet cost targets by either redesign or by sacrificing 'planned capability.' When a satisfactory design at cost target is achieved for a single increment, the 'development of each increment can proceed concurrently with the program design of the others.'

'Design is an iterative proc

of developing a design, estimating its cost, and ensuring that the design is cost-effective

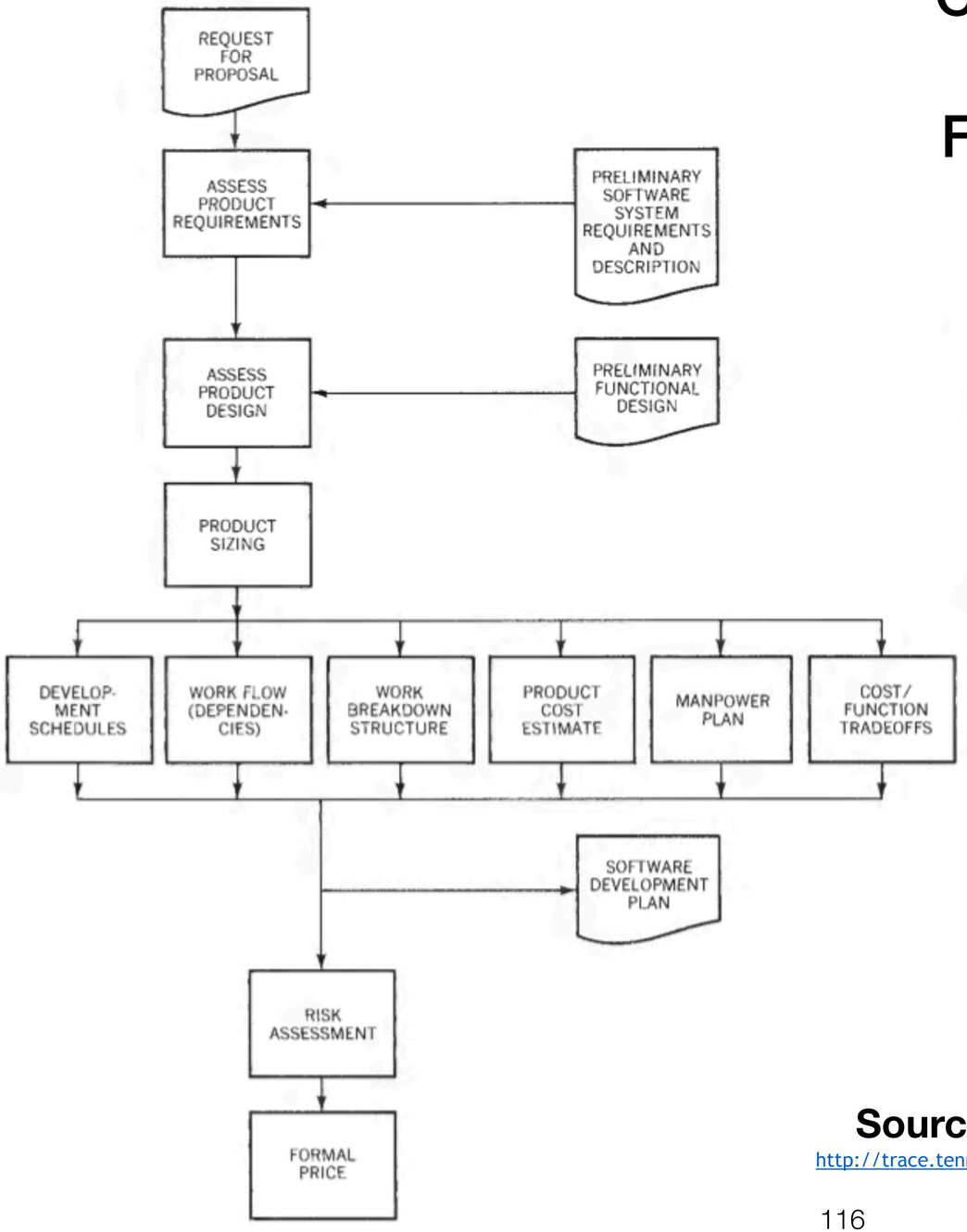
It is clear from this **seeking the appropriate balance between cost and** **us reducing the complexity of the task, an** **evelops, and as the true cost of the increment bec**

'When the development ar **ments is computed.' (p. 474)**

Source: Robert E. Quinnan, **30, pp. 466~77**

This text is cut from Gilb: The Principles of Software Engineering Management, 1988

Figure 2 Planning and estimating



Cleanroom Planning and Estimating. For making a fixed price bid

"developing a design, estimating its cost"

<— making sure it is cost effective (static)

Source: Quinnan, IBM SJ, page 472

http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1004&context=utk_harlan

Cleanroom Cost Management Process

Figure 1 Cost management process

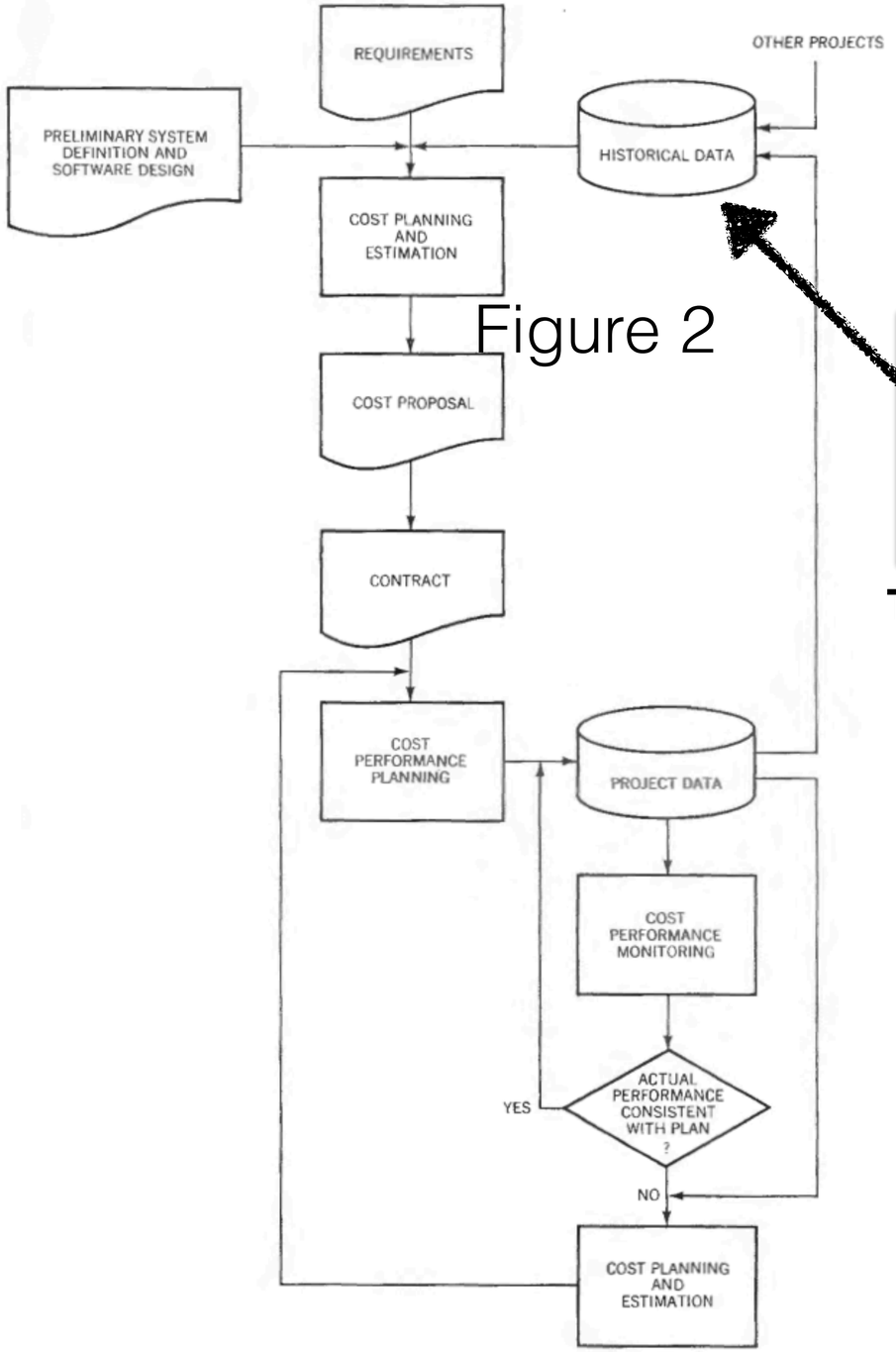
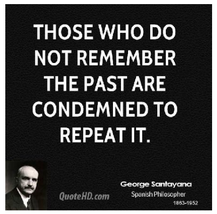


Figure 2

IBM FSD had a very advanced detailed collection of historical data from previous projects. Published in IBM SJ, Walston and Felix About 20 pages of data per project were collected



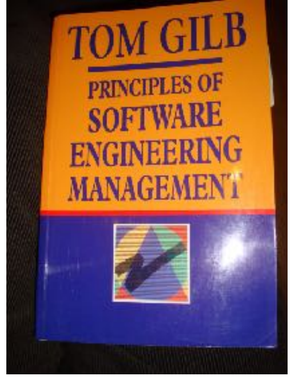
Think: Fighting Covid-19 Virus by data collection

"ensuring that the design is cost effective"

Source: Quinnan, IBM SJ, page 471

http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1004&context=utk_harlan

Design better, ‘as needed’.
See your need for tradeoffs, ‘as needed’



Quinnan describes the process control loop used by IBM FSD to ensure that cost targets are met.

'Cost management. . . yields valid cost plans linked to technical performance. Our practice carries cost management farther by introducing design-to-cost guidance. Design, development, and managerial practices are applied in an integrated way to ensure that software technical management is consistent with cost management. The method [illustrated in this book by Figure 7.10] consists of developing a design, estimating its cost, and ensuring that the design is cost-effective.' (p. 473)

He goes on to describe a design iteration process trying to meet cost targets by either redesign or by sacrificing 'planned capability.' When a satisfactory design at cost target is achieved for a single increment, the 'development of each increment can proceed concurrently with the program design of the others.'

'Design is an iterative proc

It is clear from this **seeking the appropriate balance between cost and** **us reducing the complexity of the task, an** **evelops, and as the true cost of the increment bec**

'When the development ar **ments is computed.' (p. 474)**

Source: Robert E. Quinnan, **30, pp. 466~77**

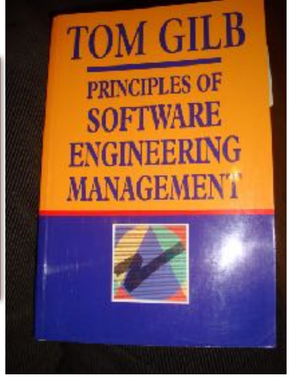
This text is cut from Gilb: T

iteration process trying to meet cost targets by either *redesign* or by *sacrificing* 'planned capability’

See diagram Figure 3 below

Copyright Tom@Gilb.com

Do design based on incremental feedback and facts, *'one small step for mankind'*



Quinnan describes the process control loop used by IBM FSD to ensure that cost targets are met.

'Cost management. . . yields valid cost plans linked to technical performance. Our practice carries cost management farther by introducing design-to-cost guidance. Design, development, and managerial practices are applied in an integrated way to ensure that software technical management is consistent with cost management. The method [illustrated in this book by Figure 7.10] consists of developing a design, estimating its cost, and ensuring that the design is cost-effective.' (p. 473)

He goes on to describe a design iteration process trying to meet cost targets by either redesign or by sacrificing 'planned capability.' When a satisfactory design at cost target is achieved for a single increment, the 'development of each increment can proceed concurrently with the program design of the others.'

'Design is an iterative process in which each design level is a refinement of the previous level.' (p. 474)

ing

the a
incr
won

'Wh
com
Sour
466~
This

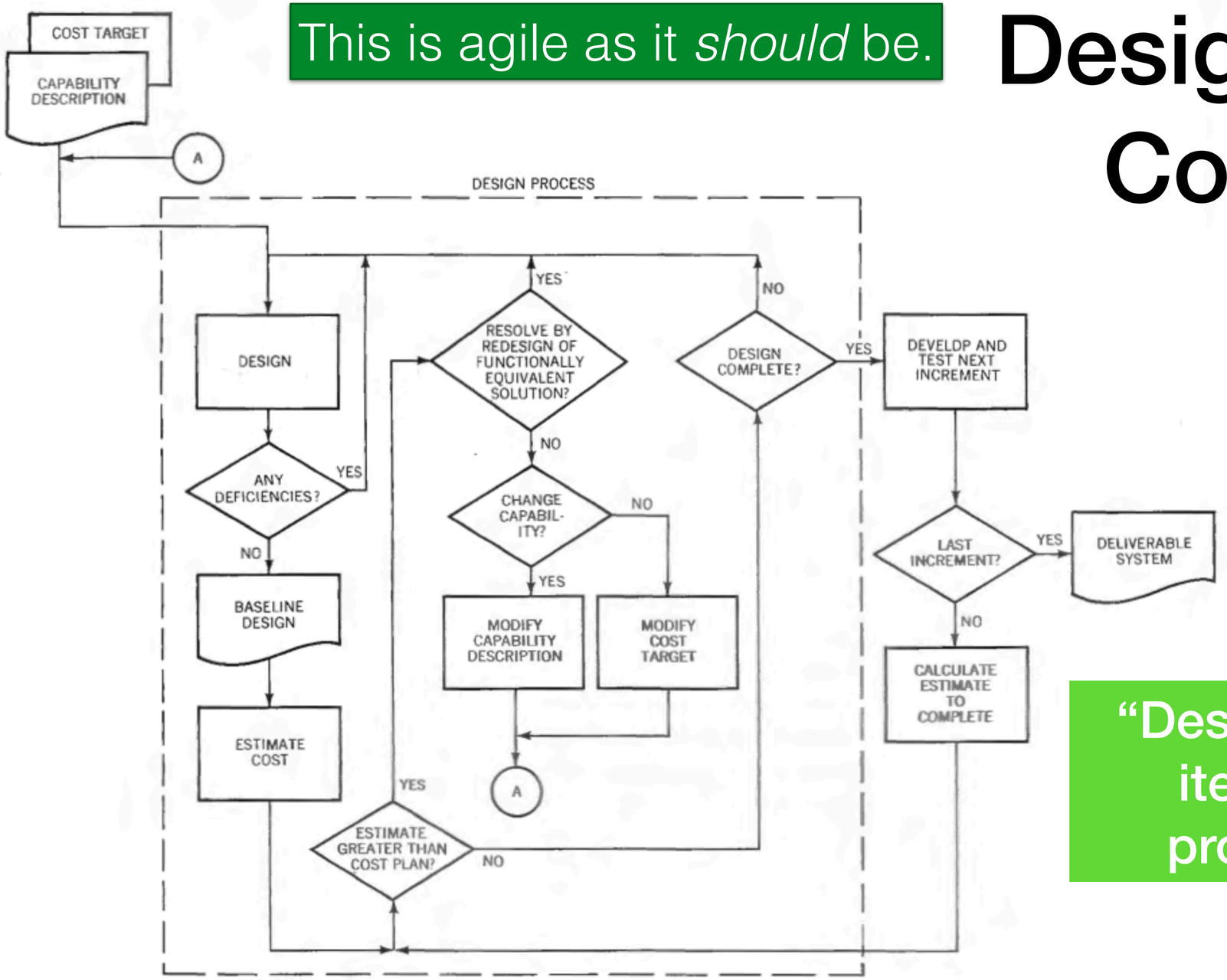
s is
o.

Design is an iterative process

Design to Cost

This is agile as it *should* be.

Figure 3 Design-to-cost



"Design is an iterative process"

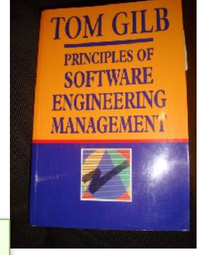
Value Agile
https://www.dropbox.com/sh/o2g7ib3z2g2uzfw/AAAypXlN0yA2WS4obwlDzZR3a?dl=0

Source: Quinnan, IBM SJ, page 473

http://trace.tennessee.edu/cgi/viewcontent.cgi?article=1004&context=utk_harlan

And PoSEM page 105, Figure 7.10

‘Small increments’ reduce complexity



but they iterate through a series of increments,
thus *reducing the complexity* of the *task*,
and *increasing the probability of learning from experience*

It is clear from this that they avoid the big bang cost estimation approach. Not only do they iterate in seeking the appropriate balance between cost and design for a single increment, but they iterate through a series of increments, thus reducing the complexity of the task, and increasing the probability of learning from experience, won as each increment develops, and as the true cost of the increment becomes a fact.

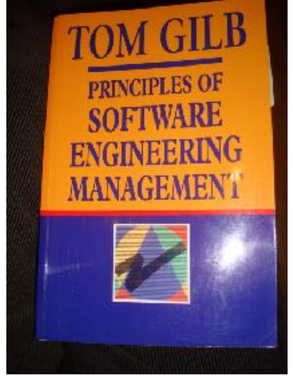
'When the development and test of an increment are complete, an estimate to complete the remaining increments is computed.' (p. 474)

Source: Robert E. Quinnan, 'Software Engineering Management Practices', IBM Systems Journal, Vol. 19, No. 4, 1980, pp. 466~77

This text is cut from Gilb: The Principles of Software Engineering Management, 1988

Copyright Tom@Gilb.com

It is less complex to estimate future costs: *incrementally*, based on *experience*



"an estimate to complete the remaining increments is computed."

(See Figure 3 above for flowchart)

Qu

'Co
int
tha
co

ement farther by
ed way to ensure
by Figure 7.10]

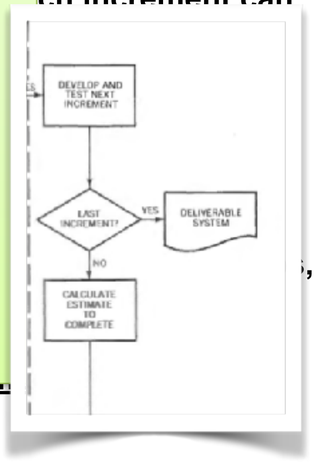
ca
pr

crificing 'planned
ch increment can

'De

ap
re
an

'When the development and test of an increment are complete, an estimate to complete the remaining increments is computed.' (p. 474)



Source: Robert E. Quinnan, 'Software Engineering Management Practices', IBM Systems Journal, Vol. 19, No. 4, 1980, pp. 466~77
This text is cut from Gilb: The Principles of Software Engineering Management, 1988

Copyright Tom@Gilb.com

Part 10.
Organising the value design process

Here are some suggestions for specialist roles. These are mainly responsibilities, rather than full time jobs. But they do require training and knowledge.

Value Analyst: analyzes stakeholder needs, and priorities, and selects critical, or possibly critical, needs and specified them as requirements, at least at the 'Wish' level (potential Goal requirement).

Specification Owner: a person (or group) which has undertaken responsibility, by name, for the update and maintenance of a specification object, such as a requirement, a design, or a table.

Implementation Responsible: a person (or group) which has taken named responsibility, as specified in the specification object, for actual practical implementation of a design object. This can be for a requirement level (reach the requirement Goal), or for a design (deliver the design and try to get the maximum value from it).

Value Designer: a generic (all possible design areas) designer (or team) who undertakes to identify possible design components to reach a Value Requirement level, on time. To research them as to all side-effects and costs, documenting such facts in the design object and corresponding Value Tables. The Value Designer might hand over exploration of a design idea to a Specialist.

Value Engineering Specialist: a designer with a narrow speciality (usability, security, performance, organizational improvement, AI) who is updated on the state of the art, and has a good international network of people and sources to find good specialist designs.

Value Architect: A person or team, who sits at the Apex of the system, and synchronizes all ongoing efforts in order to get maximum necessary value for available resources. Manages the top critical values, and the top level design architecture.

Stakeholders: the dozens or more sources of requirements which the designers need to know about, and make sure their design fits them.

Value Process Manager: a person or team responsible for getting a best possible value stream flowing from the other people involved. Sort of like old project manager, except they are focussed on the Values/Costs numbers, not building stuff. They allocate resources (money, time), and assign people to specialist tasks.

Value Quality Control: these people carry out Specification Quality Control of specifications, to make sure the Defects Per Page is economically low enough before Exit to any other process. They are also responsible for measuring value levels and costs after incremental implementation. They will check that designs are in fact implemented as specified by suppliers for Exiting to integration delivery.

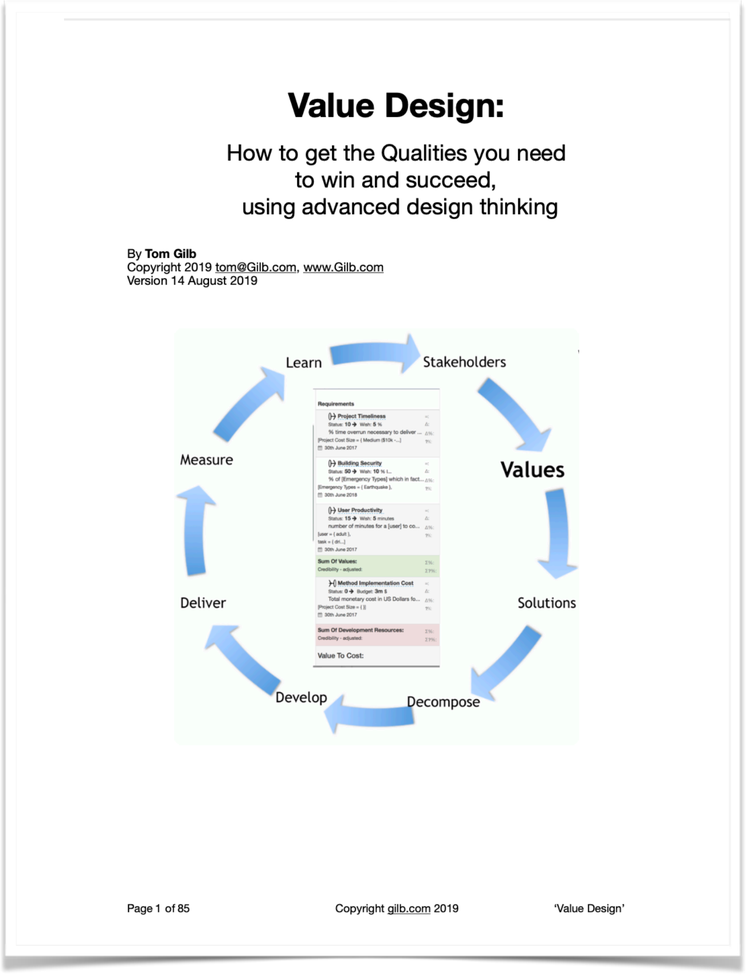
Value Policy: this is the written policy that gives clear guidance to the Value process, from organizational management. Perhaps Chief Technical Officer level.

Summary: 'Value Design'

Values are the main reason for 'projects' and 'management processes' and 'investments'.

'Value Design' offers a very clear set of tools for delivering value, quickly and as needed.

- **You must quantify all critical values, for your critical stakeholders.**
- **You need to be quantitative about resource budgets, and other constraints.**
- **You need to design consciously to reach value levels.**
- **Designs need to be decomposed into small implementable value-giving design components.**
- **You need to measure value increments, and costs.**
- **You will need to adjust initial designs so that they work better, in practice.**



Last Slide

* tom@Gilb.com
* www.Gilb.com
* @ImTomGilb
* www.linkedin.com/in/tomgilb
* +47 920 66 705
* Honorary Fellow of BCS
* Slides = http://concepts.gilb.com/dl972
* Video= https://www.youtube.com/playlist?list=PLKBhokJ0qd3_wlvr0j85YhmNfNj8ZJ8M-

Pictures from Spring edition of the Masterclass

7 / 41

GO BACK TO LAST SLIDE