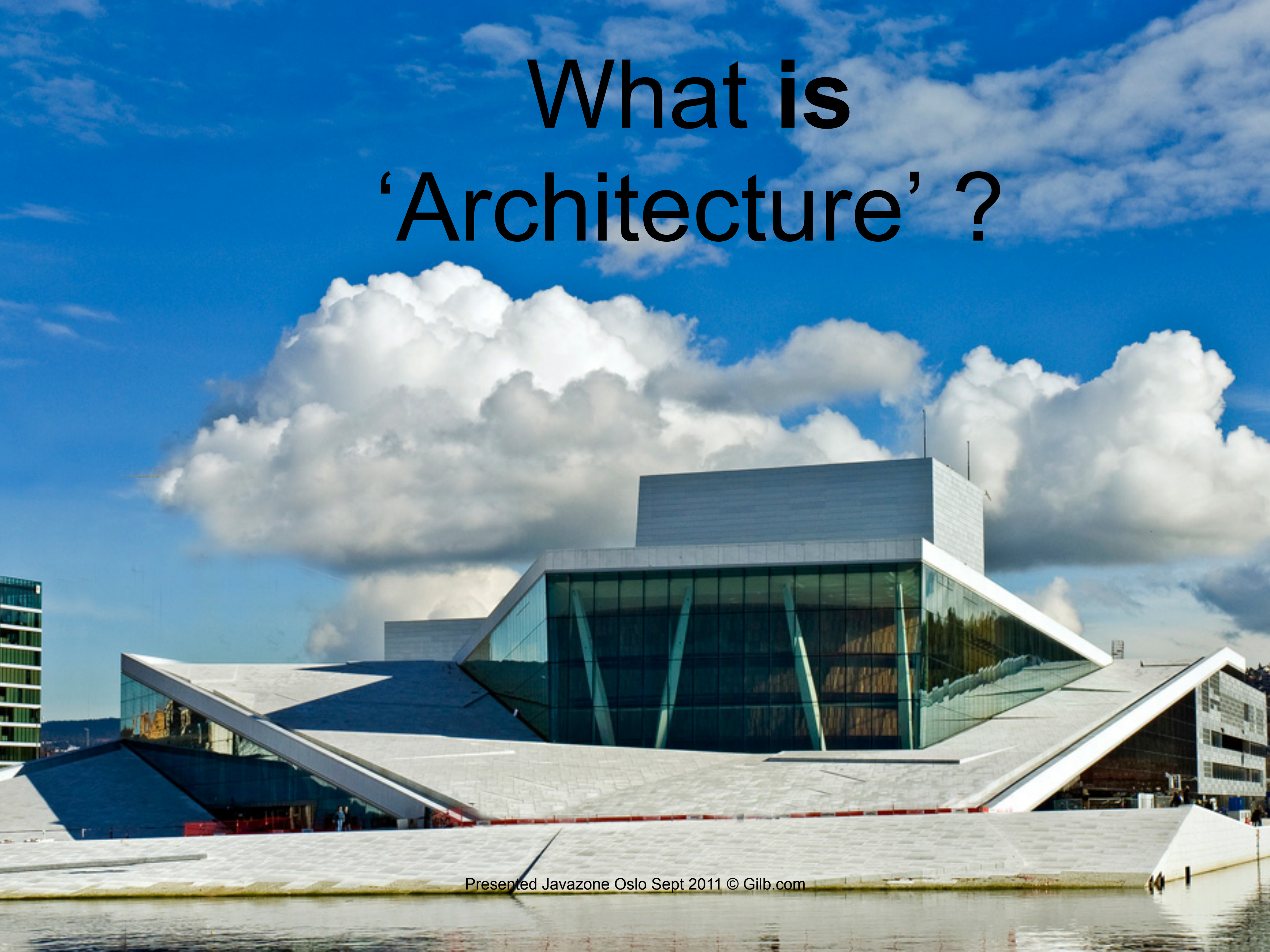
"*Real* Architecture: Engineering? or Pompous Bullshit?" by Tom Gilb and Kai Gilb

Kai & Tom Gilb

Javazone 2011 Oslo, 7-8 Sept 2011.
This Presentation 9 to 10 8th Sept

R U AN ARCHITECT ?



What **is** ‘Architecture’ ?

Architect = Master Builder

Architect is from
 ‘Archi-Tecton,’
 which means
 ‘Master Builder’.

‘Archi’ is not from
 ‘Arch’,
 but from ‘Arche’:
 primitive, original,
 primary.

Our *Personal Subjective* **Opinion** follows …

- And we are happy to discuss with you here and via tom@gilb.com, Kai@Gilb.com
- Or you can tweet your opinion at #javazone !

The architecture is there to satisfy requirements

Oslo Opera house requirements

- Qualities
- Costs
- Constraints

Oslo Opera house requirements

- Qualities
  - Impressive
  - Acoustics
  - Flexibility
  - Extendibility
  - Integratedness
  - Performance Visibility
  - National Symbol
  - Access to Fjord View
  - Comfort
- Costs
  - Building
  - Maintenance
  - Operational manpower
- Constraints
  - Legal Building
  - National Architecture
  - Archeological Site
  - Local Materials
  - Local Labour

The architecture is there to satisfy requirements

Architecture
that never refers to
necessary qualities,
performance characteristics,
costs,
and constraints
Is not really architecture
Of any kind

The architecture is there to satisfy requirements

The Architecture *process* is driven by requirements

Real (IT/Sw) Architecture

Real Architecture

- Has multidimensional *clear* design performance objectives
- Has *clear* multiple constraints
- Produces architecture ideas which enable and permit objectives to be met reasonably within constraints
- Estimates expected effects

***Pseudo* Architecture**

- Lacks dedication to clear **objectives** and **constraints**
- Does not **estimate** or articulate the expected **effects, on** objectives & constraints, of suggestions

Pseudo Architecture
Does not mention goals and constraints

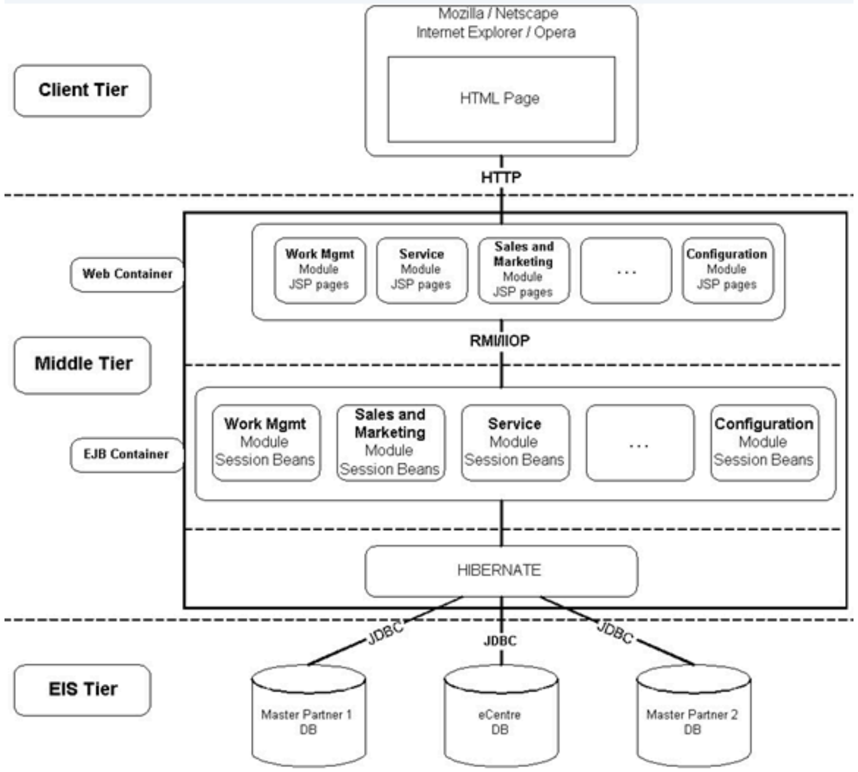
‘Bad’ ‘Arch.’ definitions

- Software architecture is a collection of software **components** unified via interfaces into decomposable system based on one or more technology platforms.
- Software Architecture shows the **structural** and **behaviour** of a system which is comprised of software **elements** and *exposing the propertie*s of those elements and relationships among them.

http://www.sei.cmu.edu/architecture/start/community.cfm

Uninformative diagrams

The following diagram shows the logical software architecture of CRM.COM Software.



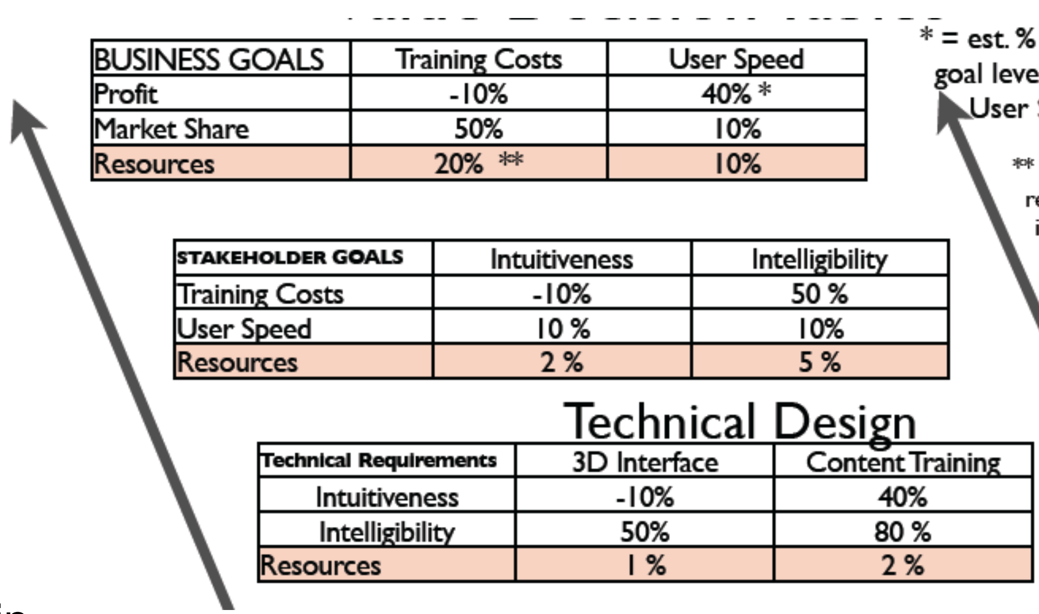
Client Tier
Mozilla / Netscape
Internet Explorer / Opera
HTML Page
HTTP
Web Container
Work Mgmt Module JSP pages
Service Module JSP pages
Sales and Marketing Module JSP pages
...
Configuration Module JSP pages
RMI/IIOP
Middle Tier
EJB Container
Work Mgmt Module Session Beans
Sales and Marketing Module Session Beans
Service Module Session Beans
...
Configuration Module Session Beans
HIBERNATE
JDBC
JDBC
JDBC
EIS Tier
Master Partner 1 DB
eCentre DB
Master Partner 2 DB

Better Architecture

Better definitions

- Software …needs to address the needs of business **stakeholders** within the organizational, technical and any other **constraints** to achieve the business, technical or any other **goals**.
  - It also needs to address software trustworthy characteristics like reliability, availability, maintainability, robustness, safety, security and survivability.
- System Architecture should contain **goals/requirements** artifacts, and structure and behavior artifacts **based on** those goals.

<u>Real</u> Architecture diagrams



* = est. %
goal leve
User
**
re
i

BUSINESS GOALS	Training Costs	User Speed
Profit	-10%	40% *
Market Share	50%	10%
Resources	20% **	10%

STAKEHOLDER GOALS	Intuitiveness	Intelligibility
Training Costs	-10%	50 %
User Speed	10 %	10%
Resources	2 %	5 %

Technical Design

Technical Requirements	3D Interface	Content Training
Intuitiveness	-10%	40%
Intelligibility	50%	80 %
Resources	1 %	2 %

A Distinction

Architecture *Process*

- A continuous, and lifecycle long, **activity of finding means for ends**

Architecture *Specification*

- A specification of
  - a set of means
  - for a set of ends

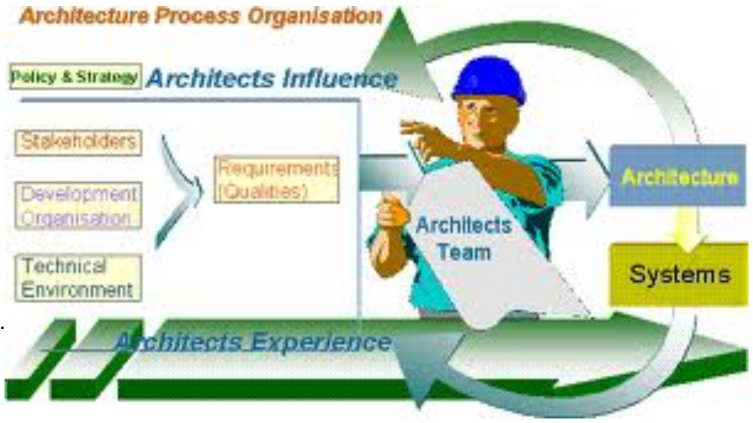
We argue that the following are **absolute essentials** for 'real' architecture

Architecture *Process* has

- Clear multiple objectives
- Clear constraints
- A process of identifying and analyzing (estimating effects of) potential means
  - For reaching objectives, within constraints

Architecture *Specification* has

- Well defined components
  - Able to deliver predictable attributes
- Credible estimates of the multiple effects of each component, and the whole



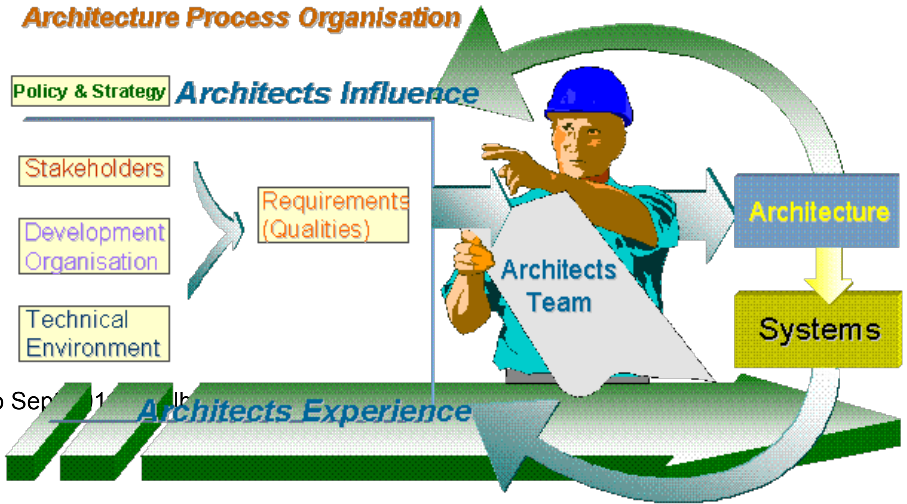
Why are these Architecture essentials, essential?

Why?

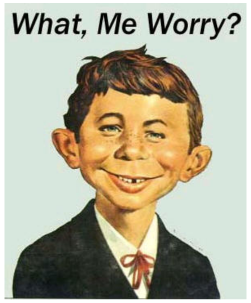
- Failure to reach even one 'critical' objective can mean total system failure
  - Example: reliability
- Failure to respect even a single constraint can mean total system failure
  - Example: cost

And if they are missing…

- You cannot expect the specified architecture will reach objectives, within constraints
- You have lost architectural control



What a Difference



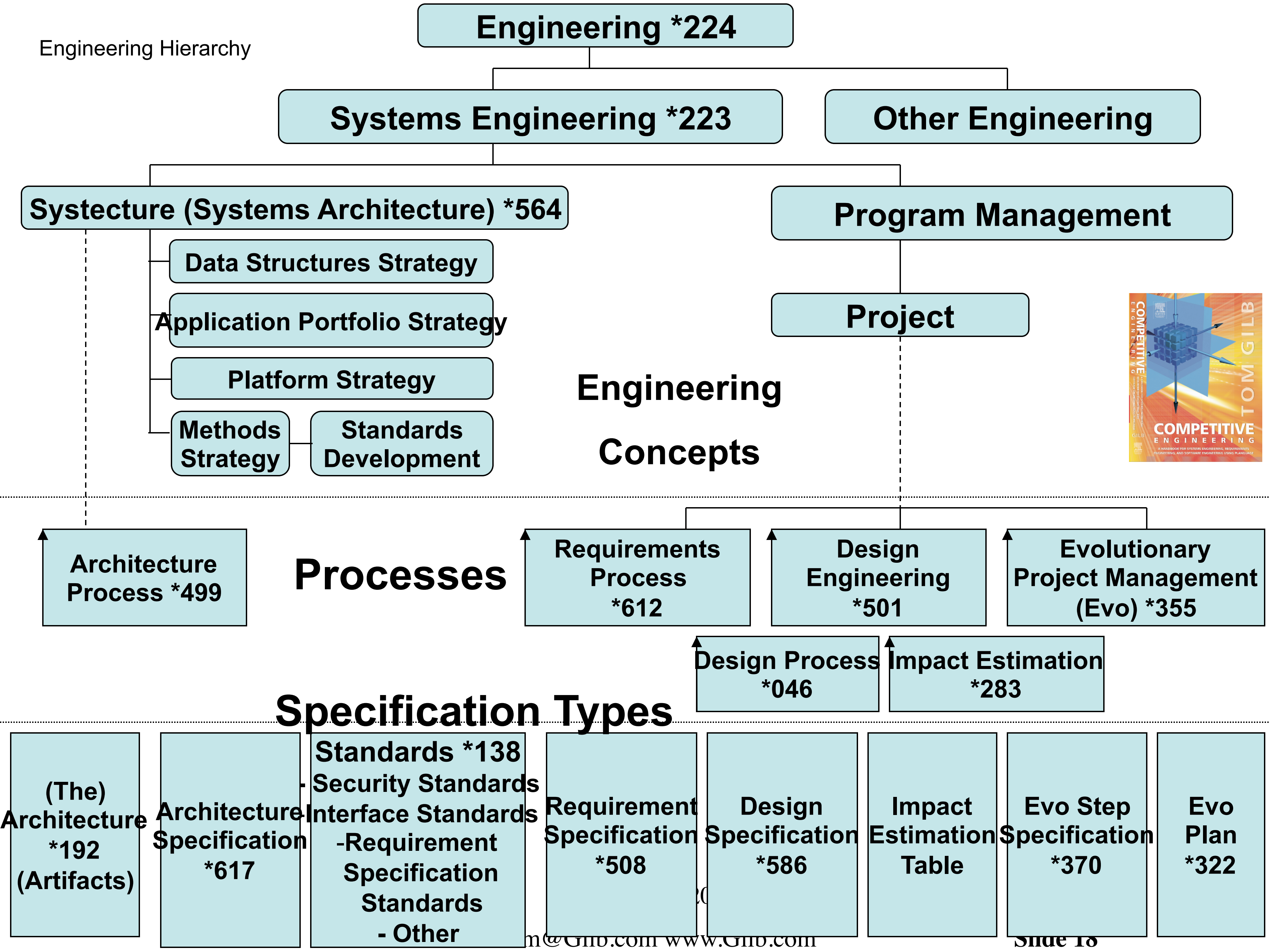
A Real Architect

- Can and does estimate resources needed for any suggested architecture
  - Capital Cost
  - Maintenance Cost
  - Skilled People hours to install and maintain
- Can and Does estimate the impact of each architecture component on the top level critical objectives
  - All '-ilities' (security etc)
  - All Performance (Capacity

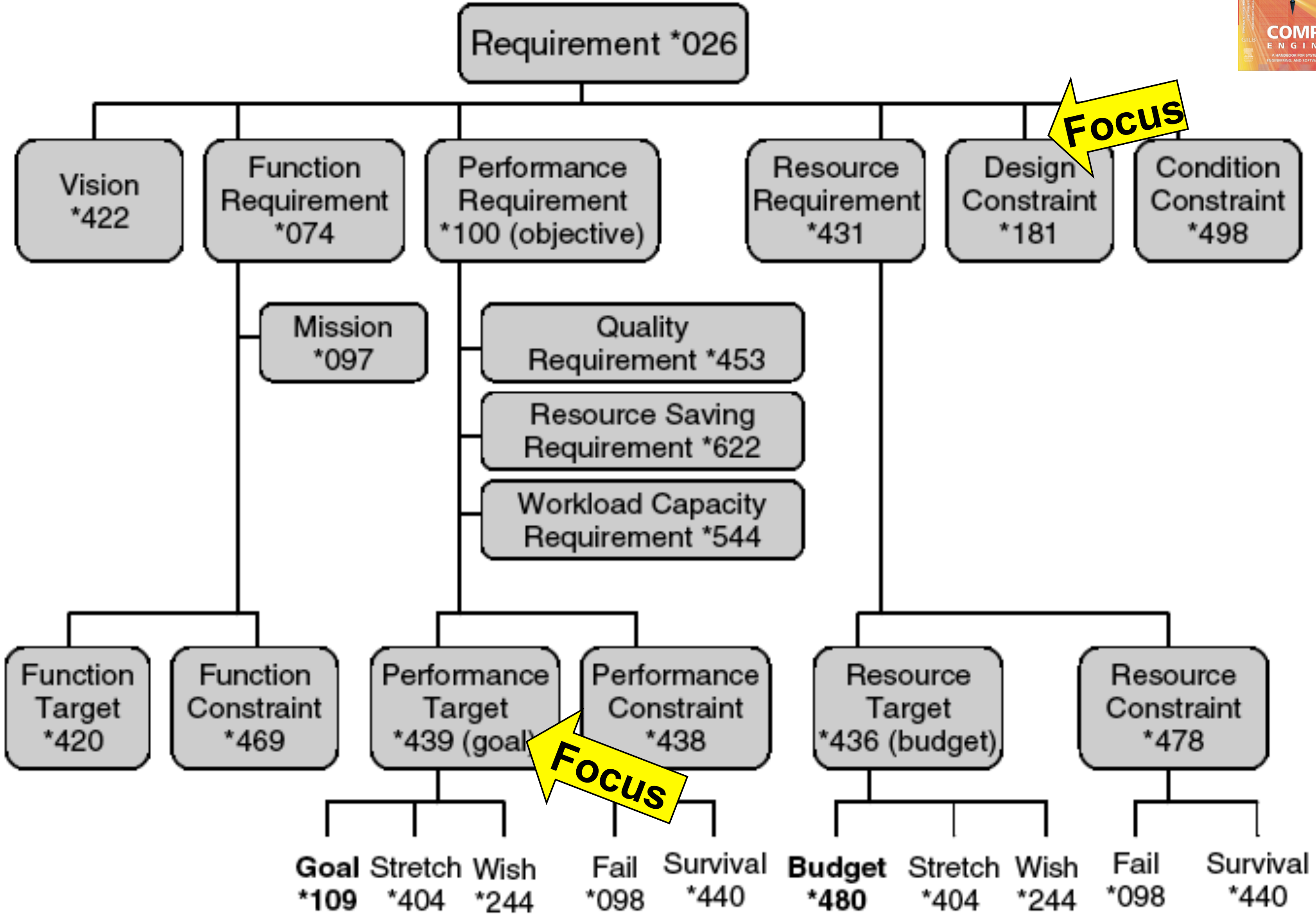
A False Architect

- Does not even try to **estimate any costs**
- of any architectures
  - Does **not know how** to do so if asked
  - If they try to estimate they are at least 10x wrong
- Does not **even try to estimate the numeric impact** on even the most critical architectural objectives
- Does not even **realize** they need **quantified performance and quality objectives** to drive and justify architecture
- They have no specific verifiable idea of the impact their ideas have on numeric quality and performance levels.
- It is all 'smoke and mirrors'
- They take **no responsibilit**y for the performance and quality attributes or costs of their suggested architecture: no skin in the game.

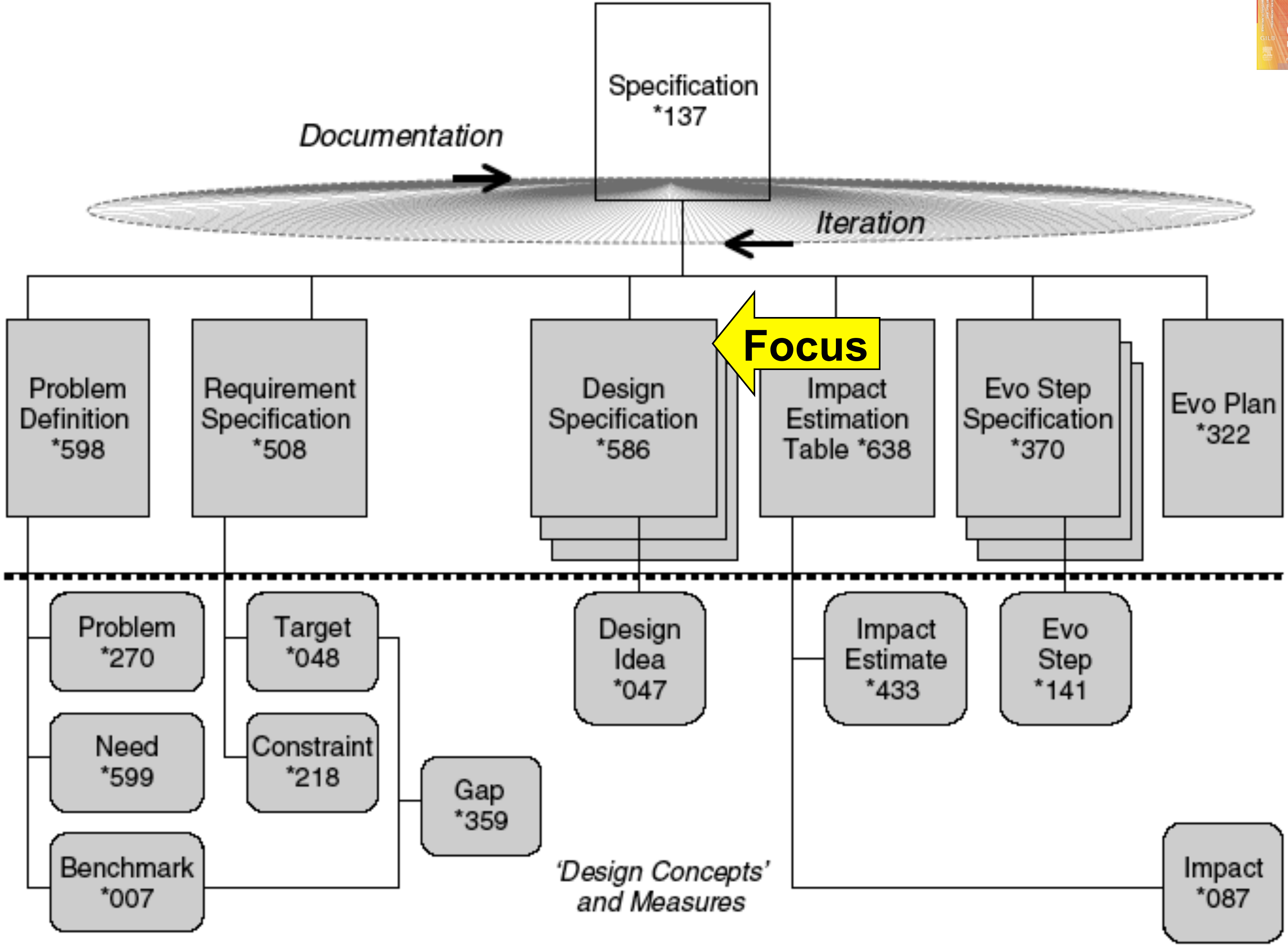
Engineering Hierarchy



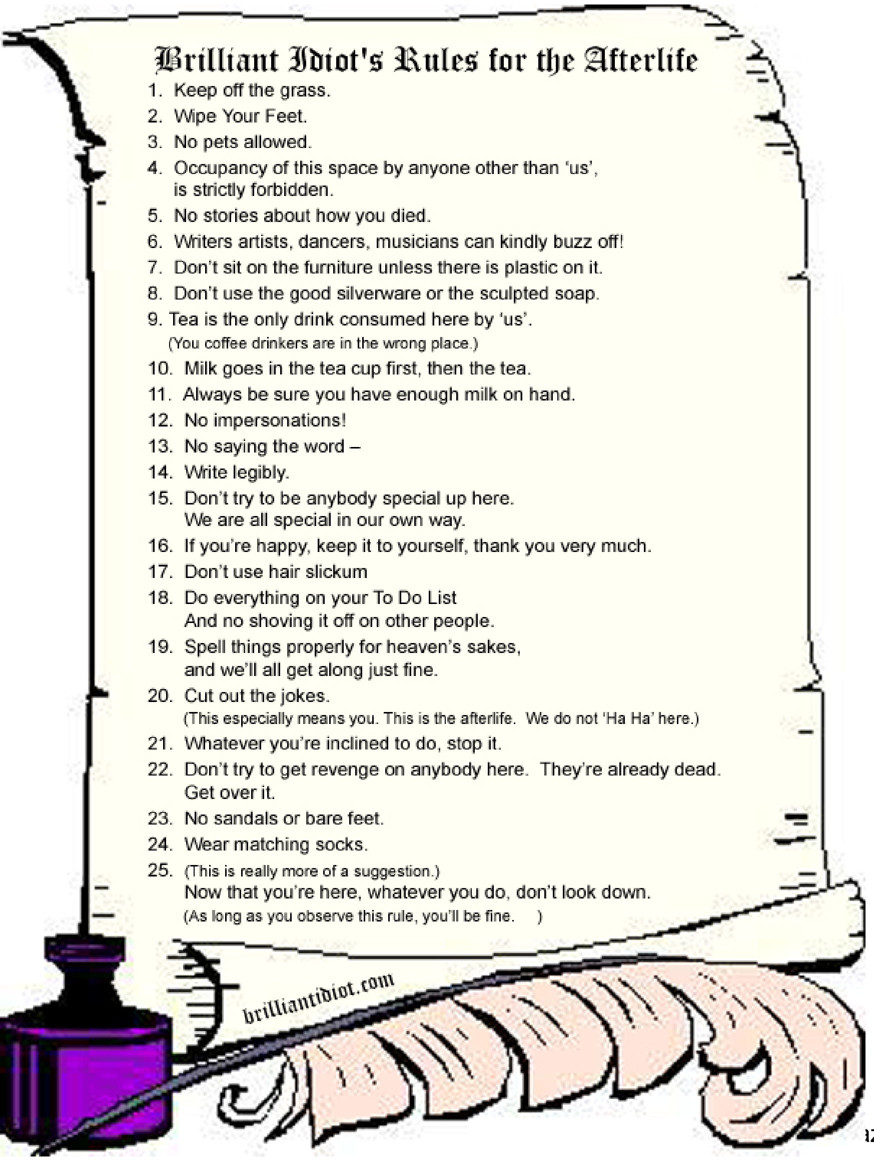
Requirement Concepts for Architects



Specification Types for Architects



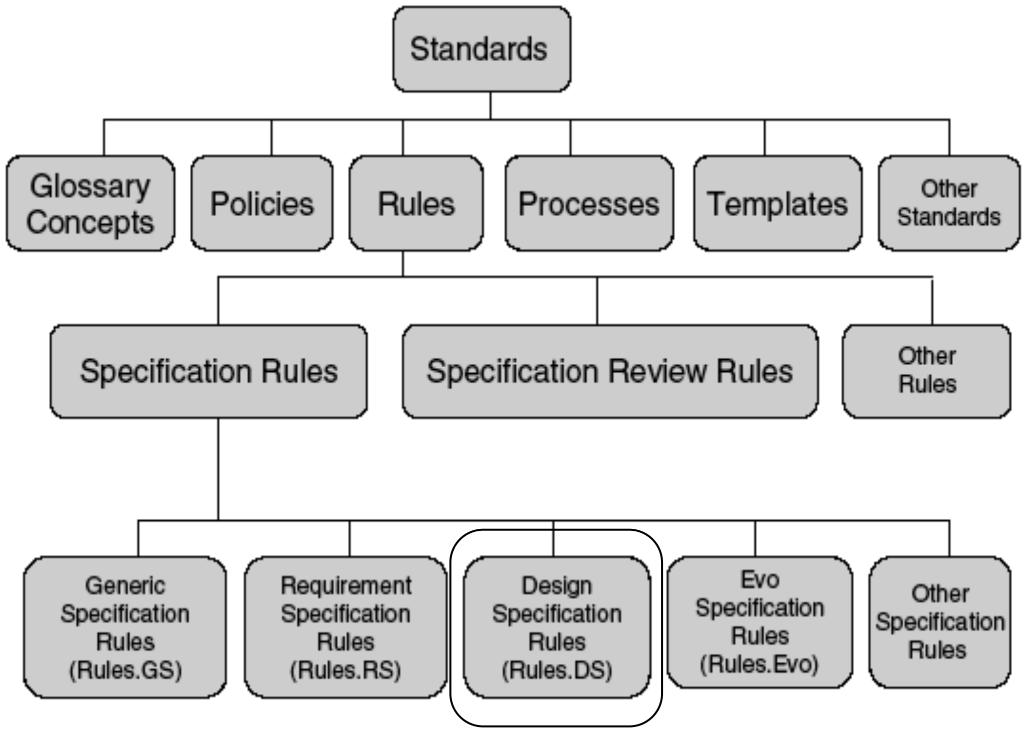
Specification Rule Types: useful for Architecture Processes and Specification 3



Brilliant Idiot's Rules for the Afterlife

1. Keep off the grass.
2. Wipe Your Feet.
3. No pets allowed.
4. Occupancy of this space by anyone other than 'us', is strictly forbidden.
5. No stories about how you died.
6. Writers artists, dancers, musicians can kindly buzz off!
7. Don't sit on the furniture unless there is plastic on it.
8. Don't use the good silverware or the sculpted soap.
9. Tea is the only drink consumed here by 'us'. (You coffee drinkers are in the wrong place.)
10. Milk goes in the tea cup first, then the tea.
11. Always be sure you have enough milk on hand.
12. No impersonations!
13. No saying the word –
14. Write legibly.
15. Don't try to be anybody special up here. We are all special in our own way.
16. If you're happy, keep it to yourself, thank you very much.
17. Don't use hair slickum
18. Do everything on your To Do List And no shoving it off on other people.
19. Spell things properly for heaven's sakes, and we'll all get along just fine.
20. Cut out the jokes. (This especially means you. This is the afterlife. We do not 'Ha Ha' here.)
21. Whatever you're inclined to do, stop it.
22. Don't try to get revenge on anybody here. They're already dead. Get over it.
23. No sandals or bare feet.
24. Wear matching socks.
25. (This is really more of a suggestion.) Now that you're here, whatever you do, don't look down. (As long as you observe this rule, you'll be fine.)

brilliantidiot.com


Standards
Glossary Concepts
Policies
Rules
Processes
Templates
Other Standards
Specification Rules
Specification Review Rules
Other Rules
Generic Specification Rules (Rules.GS)
Requirement Specification Rules (Rules.RS)
Design Specification Rules (Rules.DS)
Evo Specification Rules (Rules.Evo)
Other Specification Rules

See next slide
For detailed example

Architecture Specification Rules from CE Book Ch. 7

7.4 Rules: Design Specification

(edited down for simplicity)

R1: Design Separation: Only design ideas that are intentionally 'constraints' (Type: Design Constraint) are specified in the requirements. Any other design ideas are specified separately (Type: Design Idea).

R2: Detail: A design specification should be specified in enough detail so that we know precisely what is expected, and do not, and cannot,

inadvertently assume or include design elements, which are not actually intended.

R3: Explode: Any design idea (Type: Complex Design Idea), whose impact on attributes can be better controlled by detailing it, should be broken down into a list of the tag names of its elementary and/or complex sub-design ideas.

R4: Dependencies: Any known dependencies for successful implementation of a design idea need to be specified explicitly.

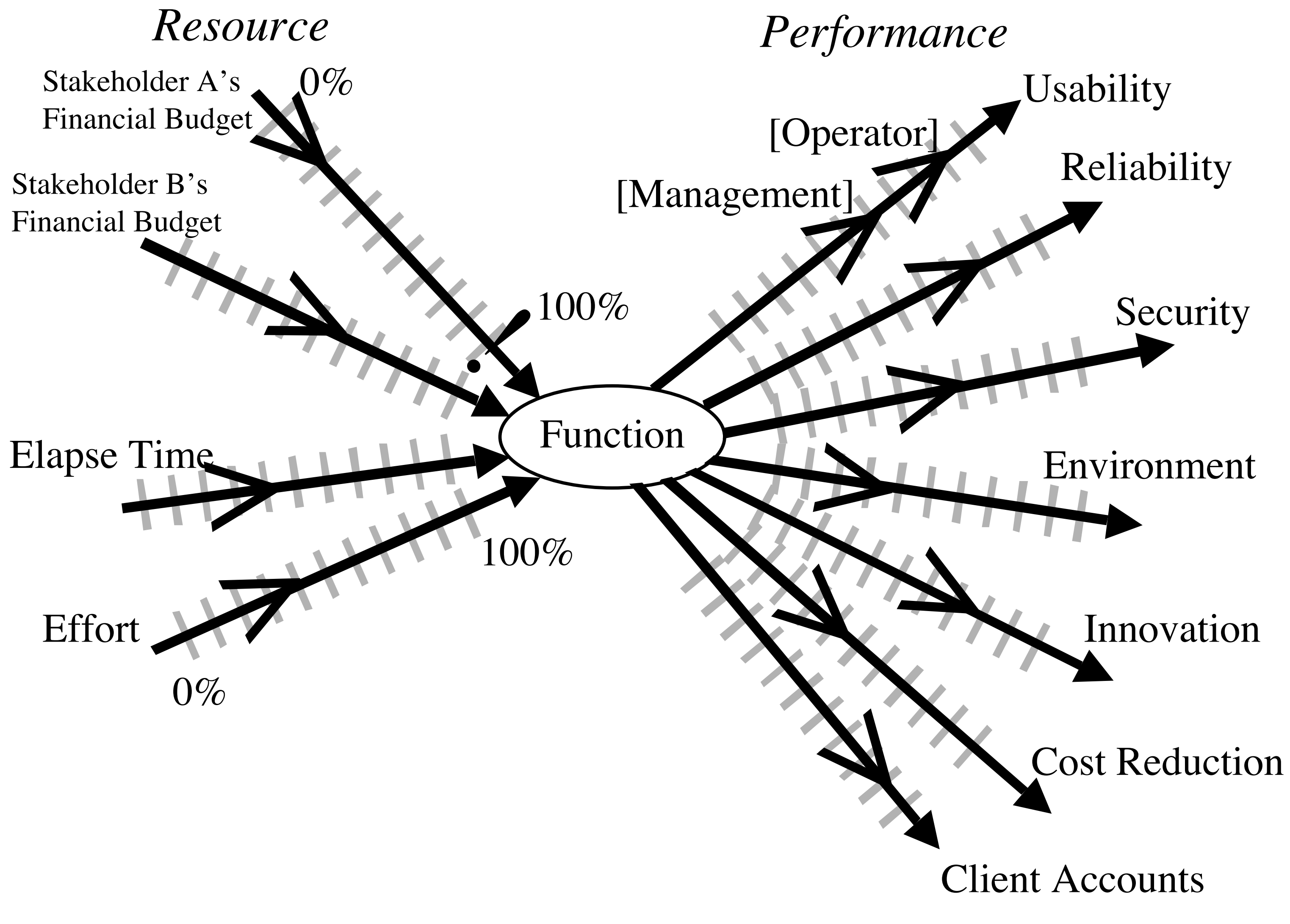
R5: Impacts: For each design idea, specify at least one main performance attribute impacted by it. Use an impact arrow '->' or the Impacts parameter.

R6: Side Effects: Document in the design specification any side effects of the design idea (on defined requirements or other specified potential design ideas) that you expect or fear. Do this using explicit parameters, such as Risks, Impacts [Side Effect] and Assumptions.

R7: Background Information: Capture the background information for any estimated or actual impact of a design idea on a performance/ cost attribute. The <u>evidence</u> supporting the impact, the level of, the level of <u>credibility</u> of any information and the <u>source</u>(s) for all this information should be given as far as possible.

R8: IE table: The set of design ideas specified to meet a set of requirements should be validated at an early stage by using an Impact Estimation (IE) table.

Multiple Required Performance and Cost Attributes are the basis for architecture selection and evaluation



Planguage Glossary

(full glossary 650+ concepts download at www.gilb.com)
http://www.gilb.com/tiki-download_file.php?fileId=387

- **Architecture (collective noun):**
  - Concept *192. May 9 2005

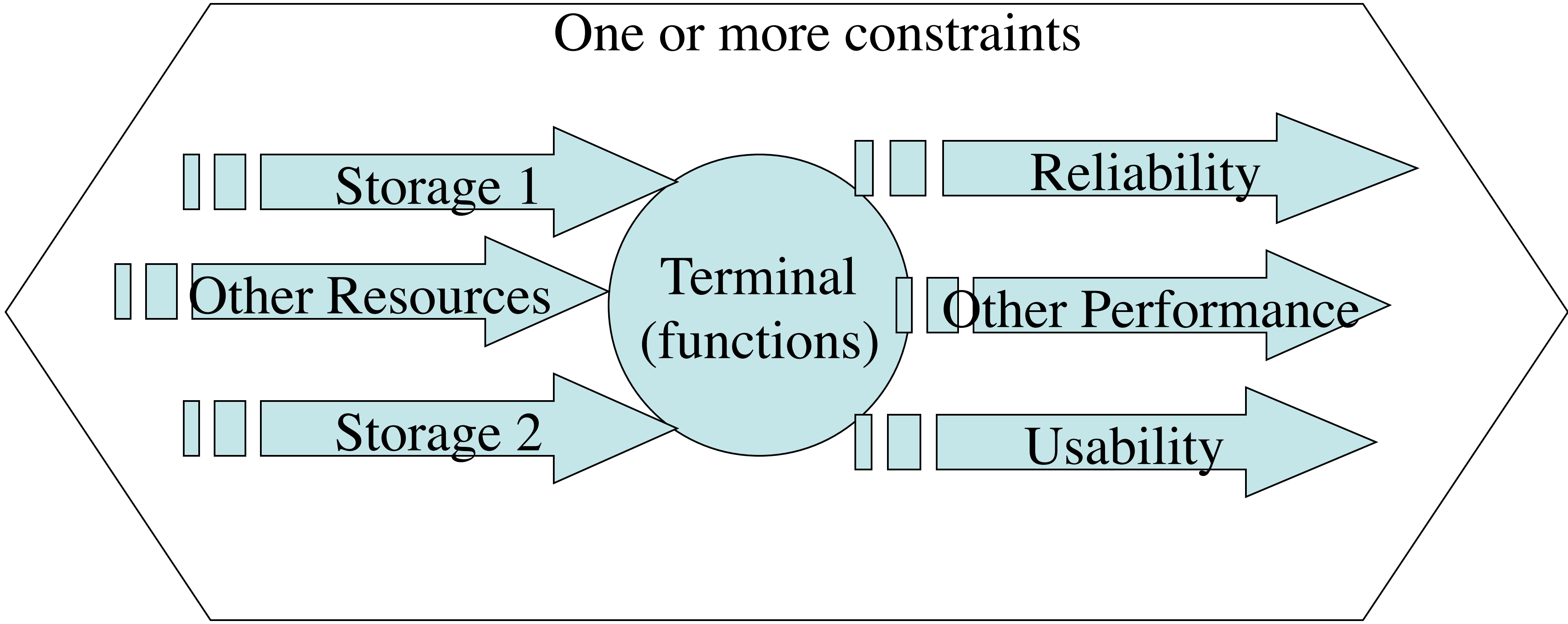
- The 'architecture' is
  - the set of entities that in fact exist
  - and impact a set of system attributes
  - directly, or indirectly, by
    - constraining,
    - or influencing,
      - related engineering decisions.

Architecture *Requirements*

- *Requirements are*
  - *a set of architecture process inputs which include:*
    - *function (what the system must do)*
    - *performance goals (how well it must perform its functions)*
    - constraints
      - (resource constraints, performance constraints, design constraints, other restrictions).

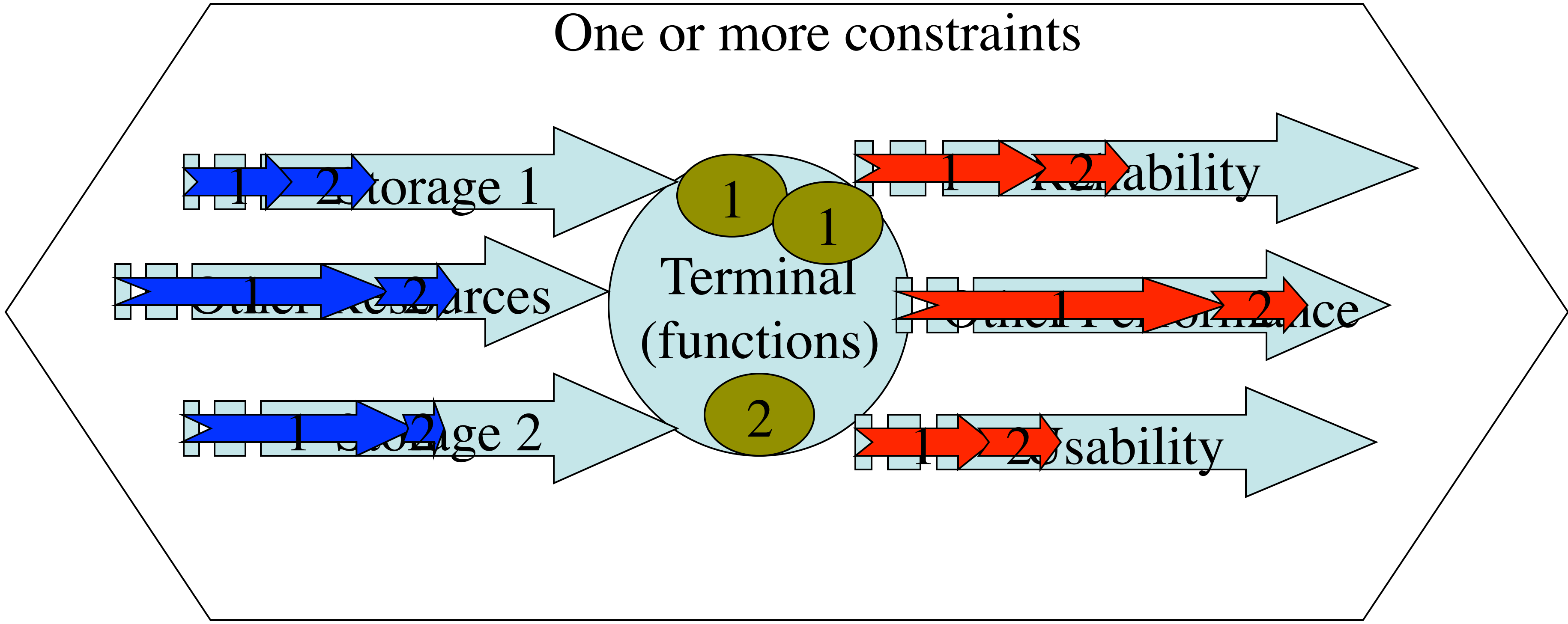
Evo and Requirements, Conceptually

Requirements are the framework for Evo development



Basic requirements model:
We need to meet performance and function requirements,
Within available/planned resources and within constraints.

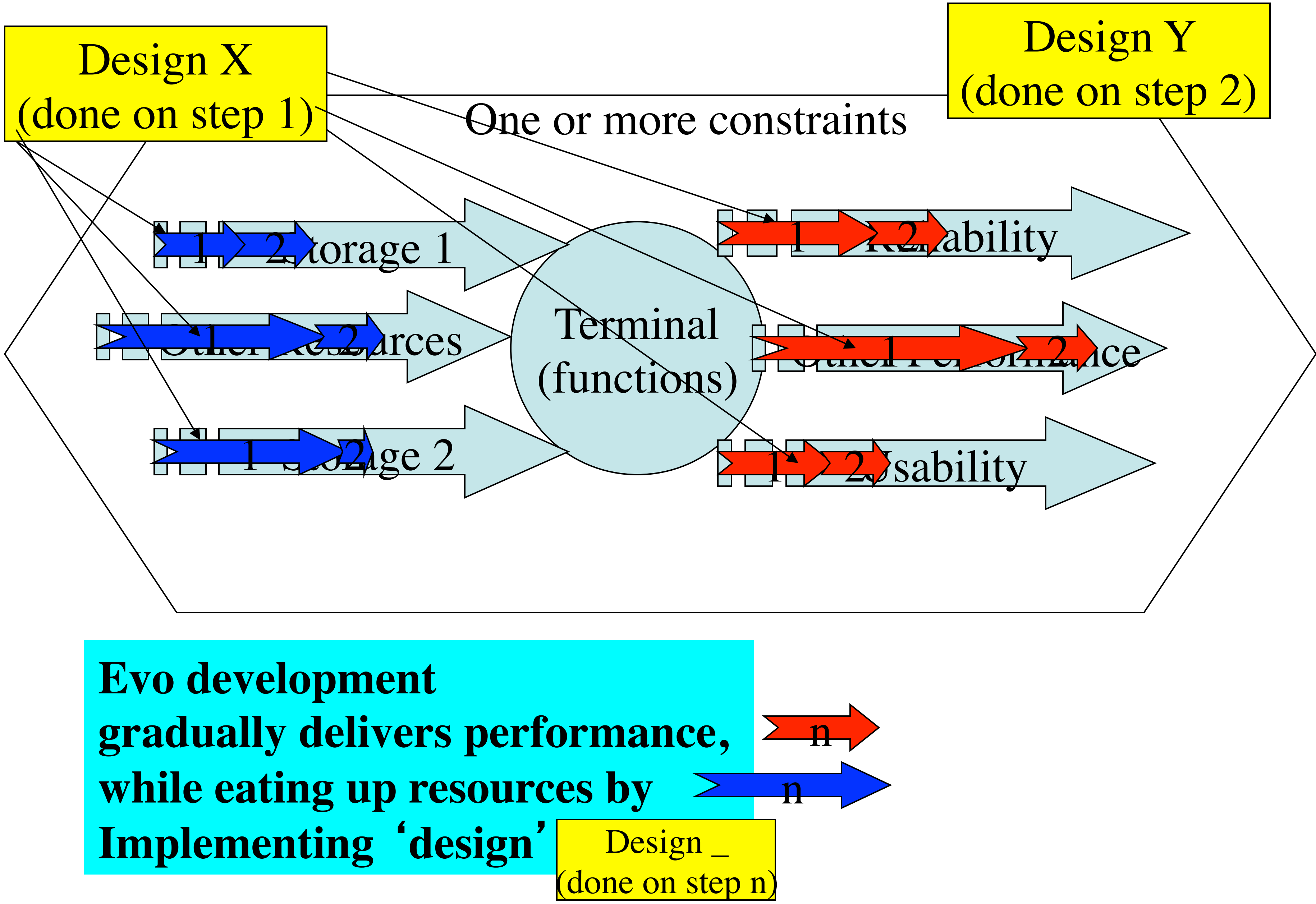
Evo and Requirements, Conceptually
Evo steps deliver partial requirements



Evo development n
gradually delivers function and performance, n
while eating up resources n

Evo and Requirements, Conceptually

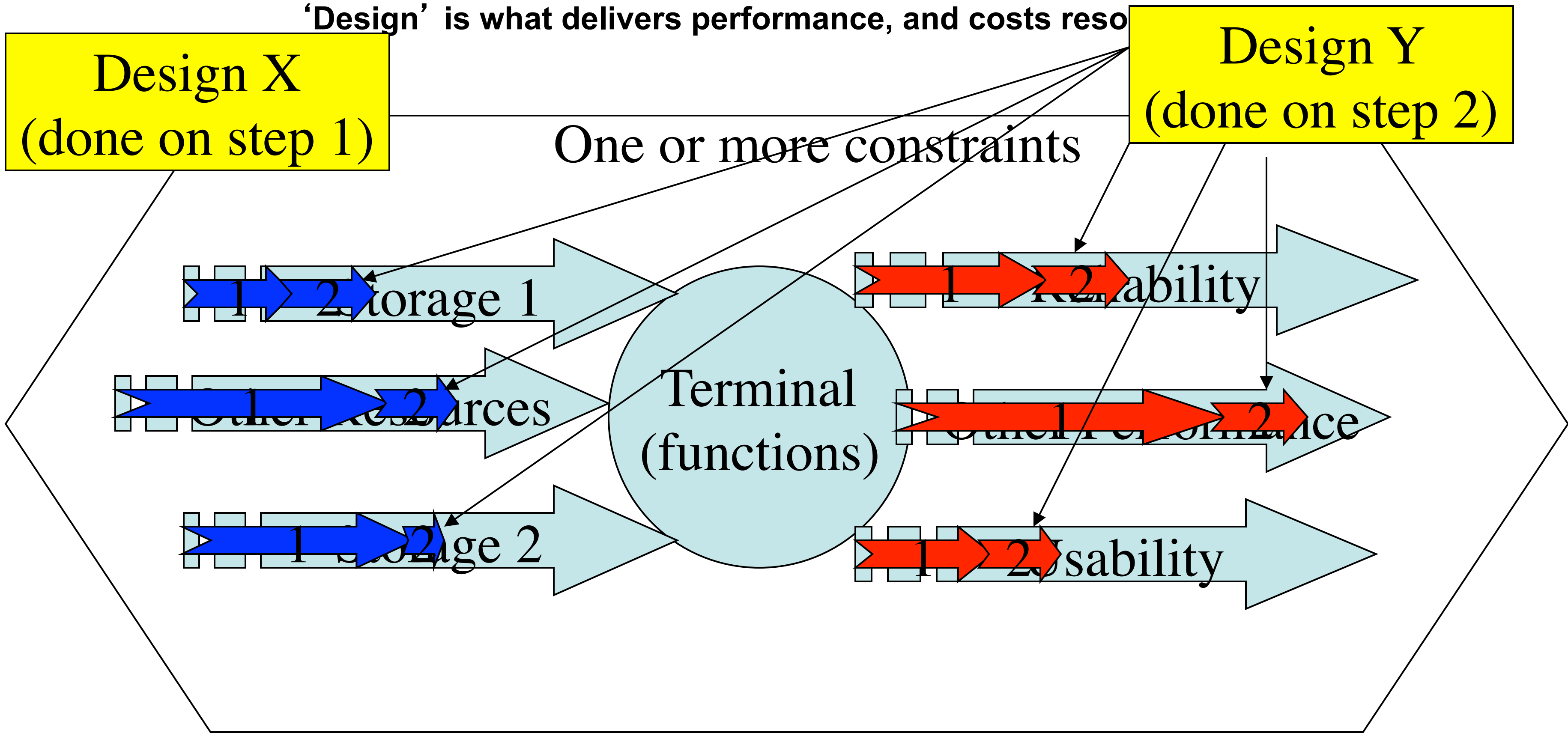
'Design' is what delivers performance, and costs resource



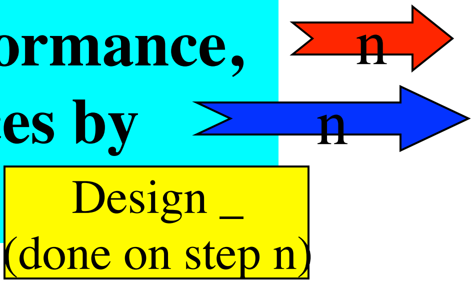
Evo development gradually delivers performance, while eating up resources by Implementing 'design'

Evo and Requirements, Conceptually

'Design' is what delivers performance, and costs reso

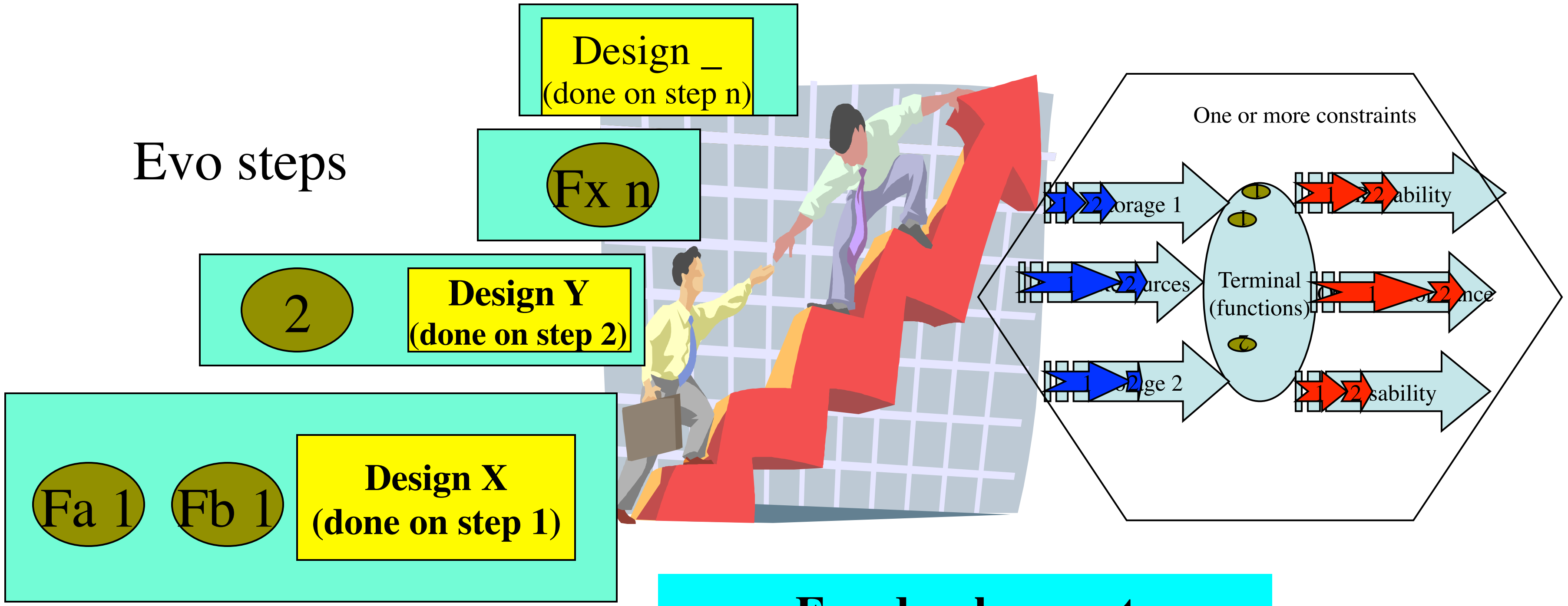


Evo development gradually delivers performance, while eating up resources by Implementing 'design'

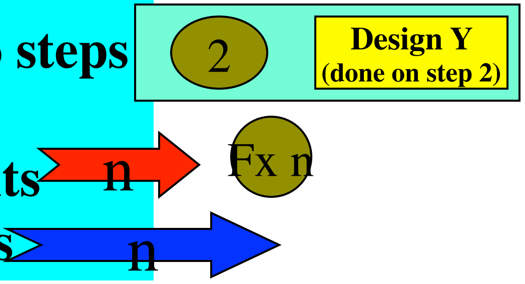


Evo and Requirements, Conceptually

'Design' is what 'delivers performance', and 'costs resource'
Function is selected or built to deliver more function
Evo steps are packages of either function and/or design



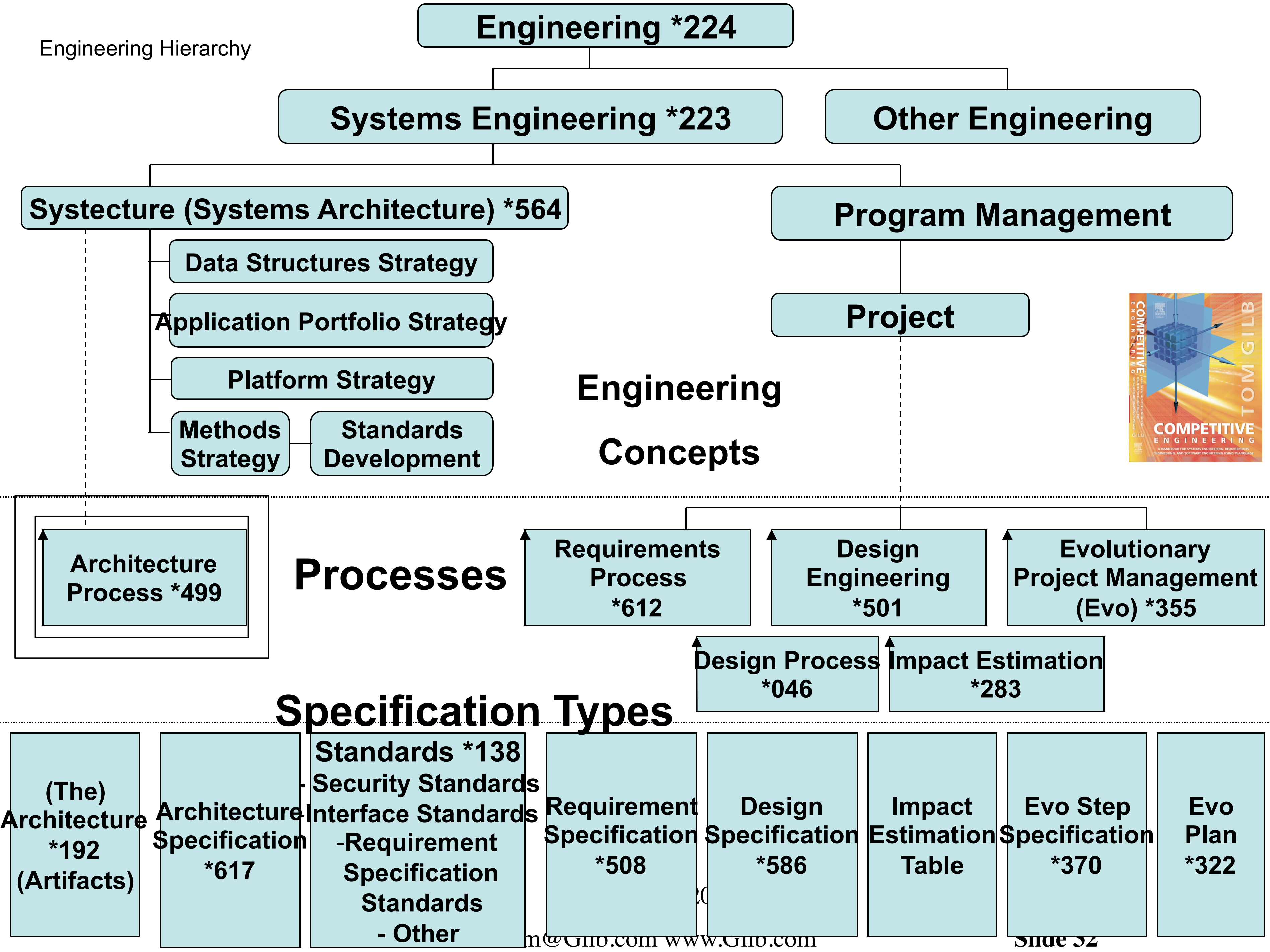
Evo development
Plans and executes Evo steps
which
Deliver requirements
And cost resources



The **Architecture** is (collective noun)

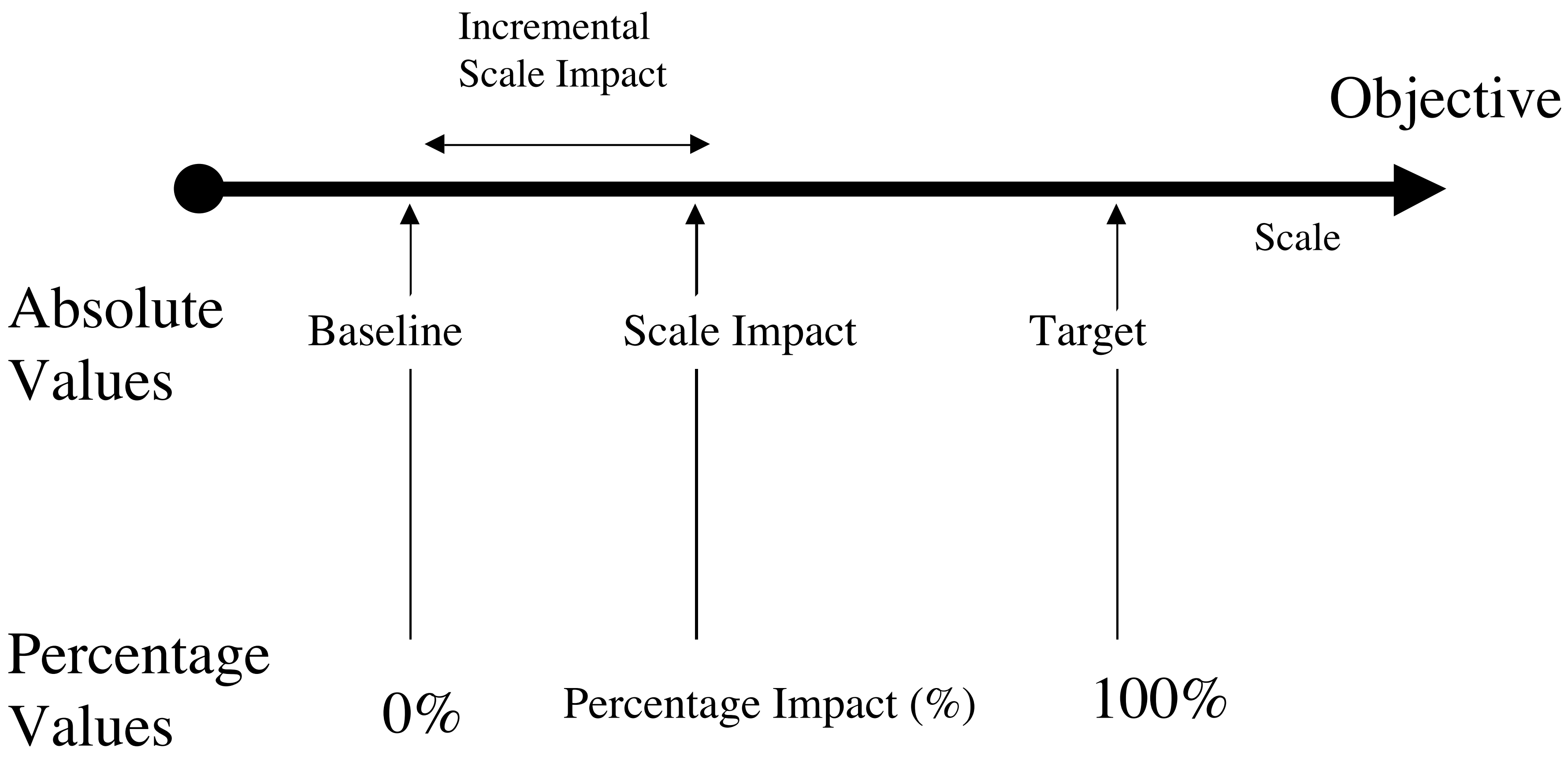
- the set of entities,
- that in fact exist
- and impact,
- a set of system attributes
- directly, or indirectly,
- by
  - constraining,
  - or influencing,
    - related engineering decisions.

Engineering Hierarchy



Engineering *224
Systems Engineering *223
Other Engineering
Systecture (Systems Architecture) *564
Program Management
Data Structures Strategy
Application Portfolio Strategy
Platform Strategy
Methods Strategy
Standards Development
Project
Engineering Concepts
Architecture Process *499
Processes
Requirements Process *612
Design Engineering *501
Evolutionary Project Management (Evo) *355
Design Process *046
Impact Estimation *283
Specification Types
(The) Architecture *192 (Artifacts)
Architecture Specification *617
Standards *138 - Security Standards - Interface Standards -Requirement Specification Standards - Other
Requirement Specification *508
Design Specification *586
Impact Estimation Table
Evo Step Specification *370
Evo Plan *322

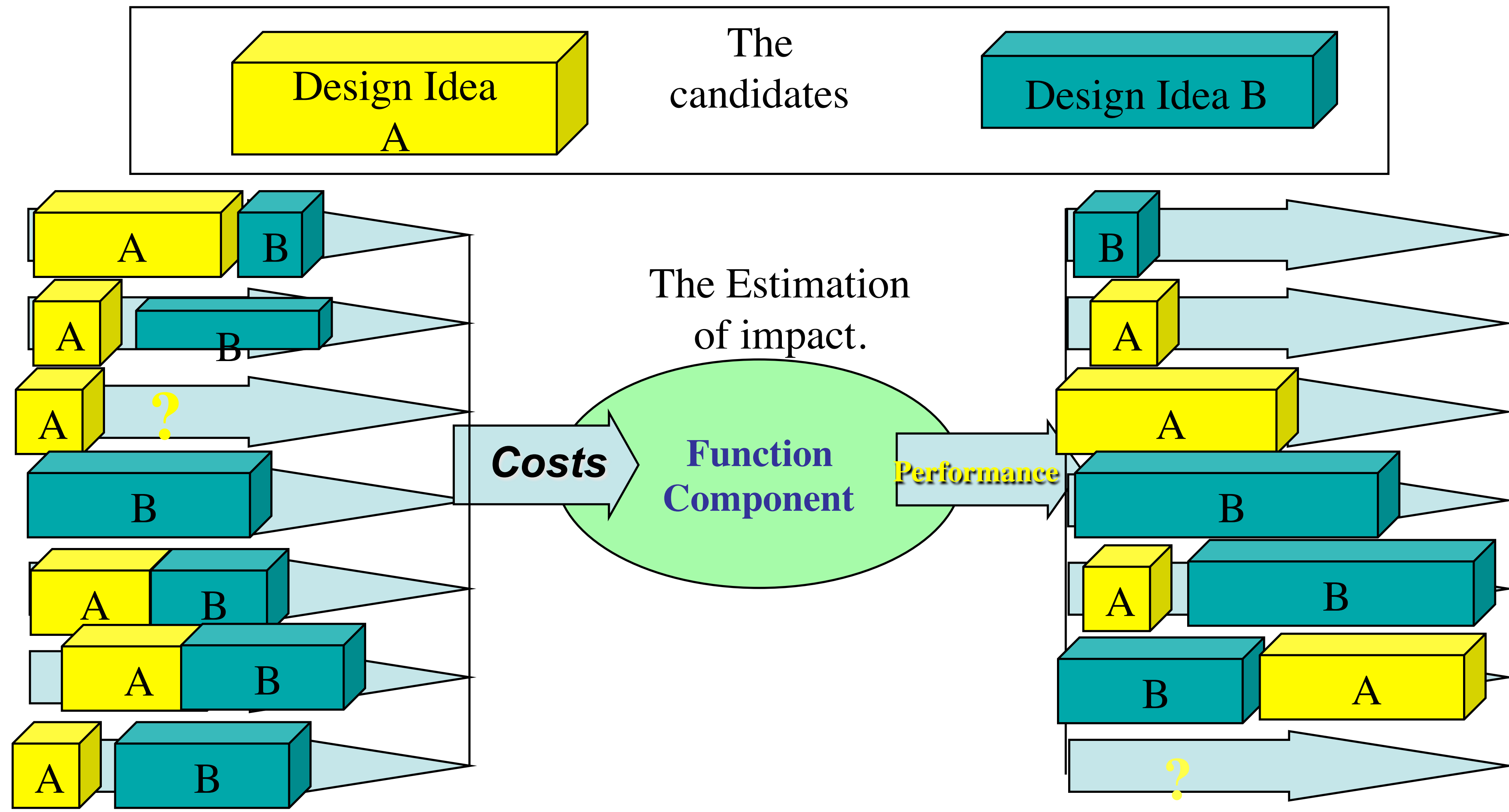
Impact Estimation Basic Concepts



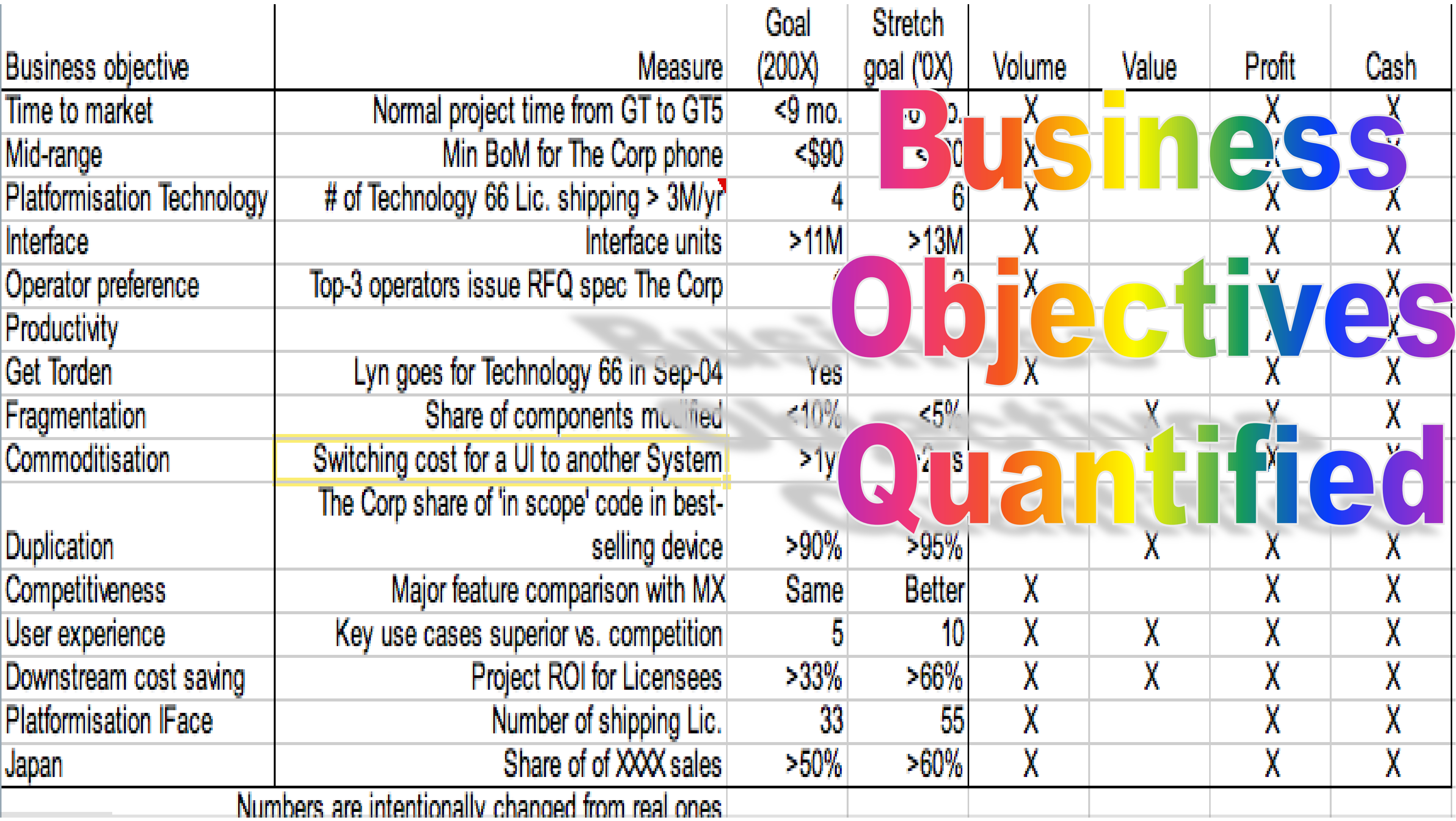
Source: Lindsey Brodie, Editor of Competitive Engineering May 2000

Impact Estimation:

How much do designs impact all critical cost and quality attributes?



•Figure 1: Real (NON-CONFIDENTIAL version) example of an initial draft of setting the objectives that engineering processes must meet.

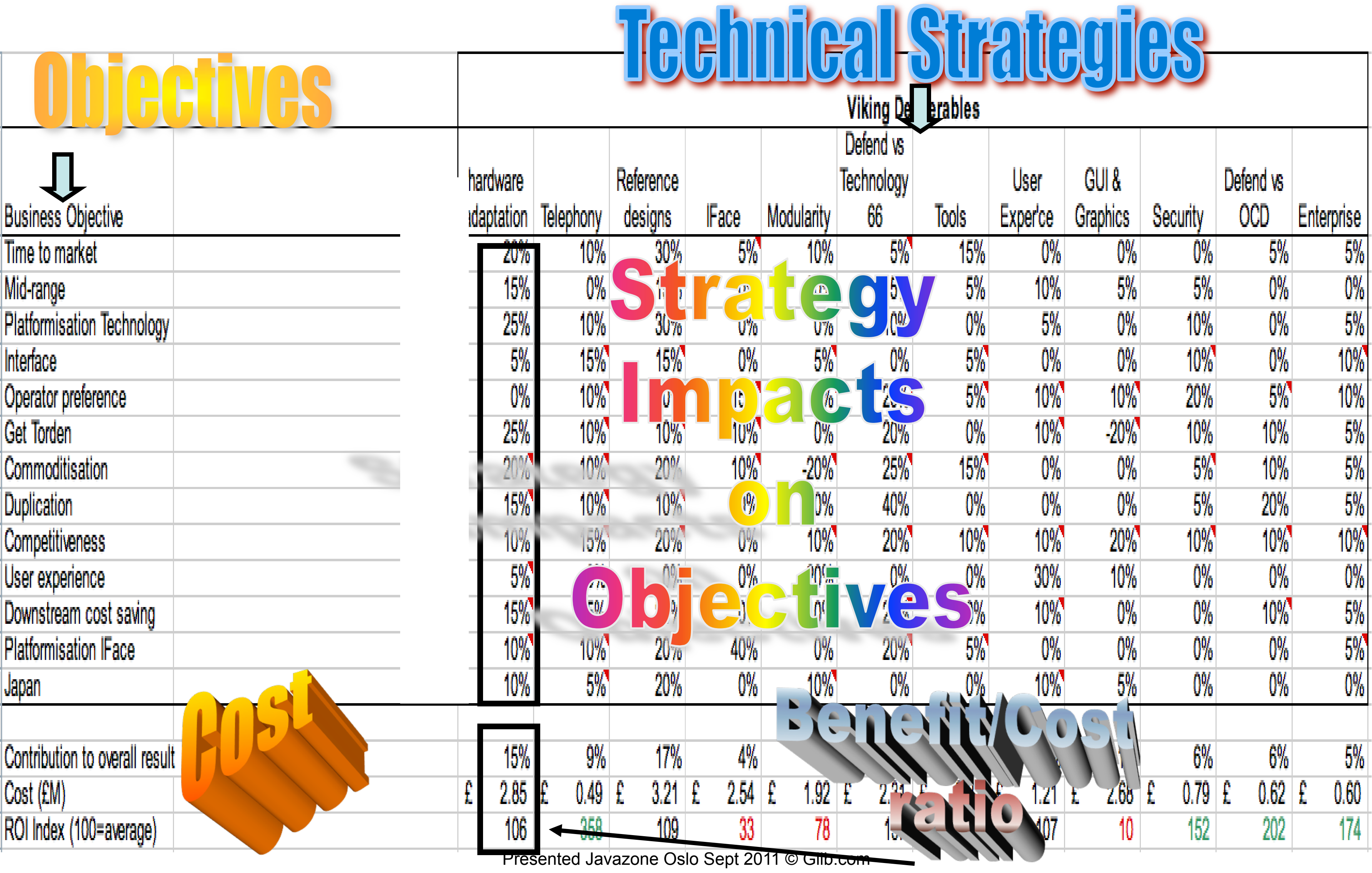


Business Objectives Quantified

Business objective	Measure	Goal (200X)	Stretch goal ('0X)	Volume	Value	Profit	Cash
Time to market	Normal project time from GT to GT5	<9 mo.	[illegible]	X		X	X
Mid-range	Min BoM for The Corp phone	<$90	[illegible]	X		X	X
Platformisation Technology	# of Technology 66 Lic. shipping > 3M/yr	4	6	X		X	X
Interface	Interface units	>11M	>13M	X		X	X
Operator preference	Top-3 operators issue RFQ spec The Corp	[illegible]	[illegible]	X		X	X
Productivity						X	X
Get Torden	Lyn goes for Technology 66 in Sep-04	Yes		X		X	X
Fragmentation	Share of components modified	<10%	<5%		X	X	X
Commoditisation	Switching cost for a UI to another System	>1y	[illegible]			X	X
Duplication	The Corp share of 'in scope' code in best-selling device	>90%	>95%		X	X	X
Competitiveness	Major feature comparison with MX	Same	Better	X		X	X
User experience	Key use cases superior vs. competition	5	10	X	X	X	X
Downstream cost saving	Project ROI for Licensees	>33%	>66%	X	X	X	X
Platformisation IFace	Number of shipping Lic.	33	55	X		X	X
Japan	Share of of XXX sales	>50%	>60%	X		X	X

Numbers are intentionally changed from real ones

Strategy Impact Estimation



Technical Strategies
Objectives
Strategy Impacts on Objectives
Cost
Benefit/Cost ratio

Viking De[illegible]erables

Business Objective	hardware adaptation	Telephony	Reference designs	IFace	Modularity	Defend vs Technology 66	Tools	User Exper'ce	GUI & Graphics	Security	Defend vs OCD	Enterprise
Time to market	20%	10%	30%	5%	10%	5%	15%	0%	0%	0%	5%	5%
Mid-range	15%	0%	[illegible]	[illegible]	[illegible]	[illegible]	5%	10%	5%	5%	0%	0%
Platformisation Technology	25%	10%	30%	0%	0%	[illegible]	0%	5%	0%	10%	0%	5%
Interface	5%	15%	15%	0%	5%	0%	5%	0%	0%	10%	0%	10%
Operator preference	0%	10%	[illegible]	[illegible]	[illegible]	[illegible]	5%	10%	10%	20%	5%	10%
Get Torden	25%	10%	10%	10%	0%	20%	0%	10%	-20%	10%	10%	5%
Commoditisation	20%	10%	20%	10%	-20%	25%	15%	0%	0%	5%	10%	5%
Duplication	15%	10%	10%	[illegible]	[illegible]	40%	0%	0%	0%	5%	20%	5%
Competitiveness	10%	15%	20%	0%	10%	20%	10%	10%	20%	10%	10%	10%
User experience	5%	[illegible]	0%	0%	[illegible]	0%	0%	30%	10%	0%	0%	0%
Downstream cost saving	15%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10%	0%	0%	10%	5%
Platformisation IFace	10%	10%	20%	40%	0%	20%	5%	0%	0%	0%	0%	5%
Japan	10%	5%	20%	0%	10%	0%	0%	10%	5%	0%	0%	0%
Contribution to overall result	15%	9%	17%	4%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	6%	6%	5%
Cost (£M)	£ 2.85	£ 0.49	£ 3.21	£ 2.54	£ 1.92	£ [illegible]	[illegible]	£ 1.21	£ 2.68	£ 0.79	£ 0.62	£ 0.60
ROI Index (100=average)	106	358	109	33	78	[illegible]	[illegible]	107	10	152	202	174

Ask for free digital copy!
(tom@gilb.com)

TOM GILB
COMPETITIVE ENGINEERING
A HANDBOOK FOR SYSTEMS ENGINEERING, REQUIREMENTS ENGINEERING, AND SOFTWARE ENGINEERING USING PLANGUAGE
ELSEVIER
GILB

Questions and Discussion

- On Real Architecture

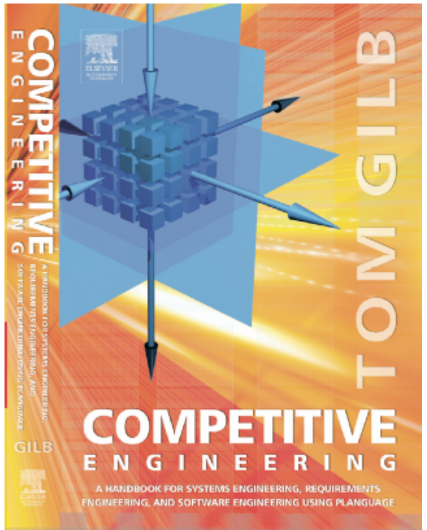
Advanced Reserve Slides

- Which we do not plan to present at Javazone
- But are in reserve
- They can give you more detail
- And might be used to answer questions in more detail

Software and Systems Engineering

- Our opinion about Software Architecture applies fully to the higher level of the system of which our 'code' is a component

- i.e. it is a *systems* engineering perspective


COMPETITIVE ENGINEERING
TOM GILB
A HANDBOOK FOR SYSTEMS ENGINEERING, REQUIREMENTS ENGINEERING, AND SOFTWARE ENGINEERING USING PLANGUAGE

Rationale: (for the Architecture definition)

- *Rationale: this definition has the following intents by the author (TG):*
- *to bring in the concept that architecture is related to multiple requirements,*
- *and must be judged in terms of*
  - *its satisfaction,*
  - *and optimization degree,*
  - *for multiple performance goals,*
  - *within multiple constraints.*
    - *This seems missing in other definitions [Maier02, Art of Architecting]*
- *to avoid the notion that architecture is done by one instance,*
  - *it can exist and have evolved, even in a 'new' system.*
- *to avoid the notion that architecture*
  - *is formally specified (this can be stated as an adjective, 'architecture specification', see below)*
- *to differentiate architecture from other design*
  - *by invoking the notion that it has the power to constrain the decisions of other engineering levels*

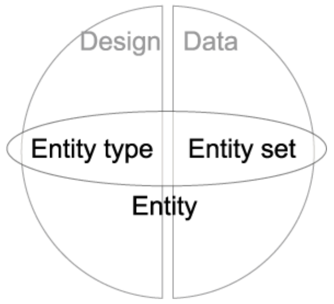
Rejected Architecture Notions

- *In particular I reject some notions common in other definitions of architecture:*
- ***structure** (MIL STD 498, Maier02 p285) : this term is commonly used to define architecture.*
  - *Even in Civil Architecture it is at best one category of the architecture.*
  - *In systems engineering it is practically, but not totally, irrelevant.*
  - *It hides the more central notion of a 'design artifact',*
    - *which is something that determines system properties or enables them*
    - *. (this point is also made by IEEE Architecture Working Group [Maier02, p285-6])*
- ***component, interfaces & connections**: same principle as for 'structure',*
  - *these describe specific but narrow classes of design artifacts.*
  - *This in practice leads to the exclusion of the more general concept of 'anything which satisfies the requirements'.*
  - *It certainly does not include concepts like training, operator selection, motivation, human communication, contracts, policies and other 'non-hardware',*
    - *which can be every bit as dramatic in influencing the architecture's impact on the system requirements.*

Interpretations of terms used in the definition of ‘The Architecture’:

“the set of entities,
that in fact exist
and impact,
a set of system attributes
directly, or indirectly,
by
constraining,
or influencing,
related engineering
decisions.”

What do we mean by the ***"Set"*** *(of entities):*

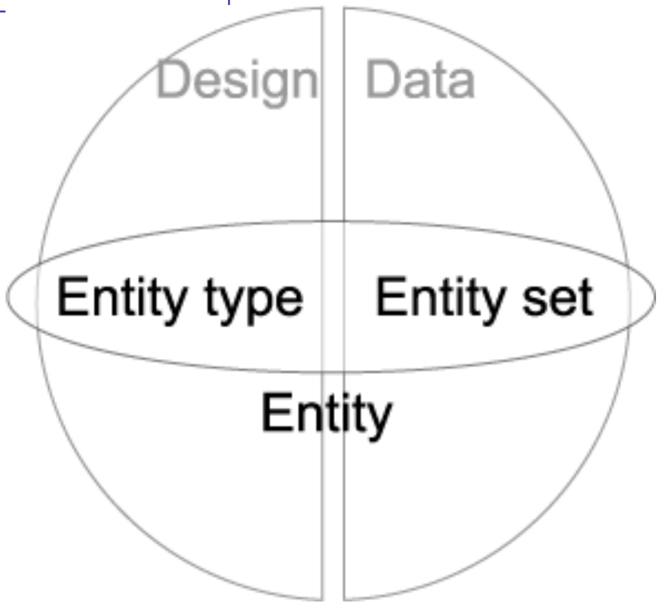


- *the notion of a set of* *<u>entities,</u>*
- *the notion of the architecture as a 'set' of arbitrarily different devices*
  - *for impacting*
  - *or controlling*
  - *the attributes of a system.*

- the **<u>set</u>** of entities,
- that in fact exist
- and impact,
- a set of system attributes
- directly, or indirectly,
- by
  - constraining,
  - or influencing,
    - related engineering decisions.

*Why do we use the term "**Entities**":*

- *this is intended to be **extremely broad** in scope*
  - *covering **everything imaginable** and discernable*
  - *which is **intended to satisfy** requirements,*
  - *and which is **intended to constrain other design**, operational environment, or life cycle activity.*
- *In particular it goes **way beyond the traditional notion** of structure, and organization.*
- *It for example includes notions of agreements, contracts, social mores, and motivation -*
  - *which never seem to get mentioned in the conventional definitions.*
- *It is also intended to cover all discernible mechanisms which are operating at this level,*
  - *no matter who selected them, when they were selected, or if the formal 'architects' are aware of them.*
- *Entities are **not** necessarily design **specifications** (*586).*
- *They are the existing design concepts (*047) themselves, no matter how they are represented, or determined.*



" in fact exist":

- ***the design artifacts may '<u>exist' because of</u>***
  - ***Conscious selection (design), tradition, accident or unintentionally, - even foolishly,***
  - ***by anybody or anything –***
    - ***including cultures, legal systems, political systems, and nature – even the formal 'architect'.***
  - ***But the point is that they are in fact in existence***
    - ***in either a real system or a model of such a system.***
  - ***The selection is not necessarily a conscious act for formal engineering***
  - ***but the design artifact is observably in place and in force – irrespective of its history.***

Implication

- An architect,
- Doing an architecture process
- May add conscious and intentional architecture entities
- To an *existing* architecture
- Containing earlier, less conscious or unconscious architecture entities

Design Process

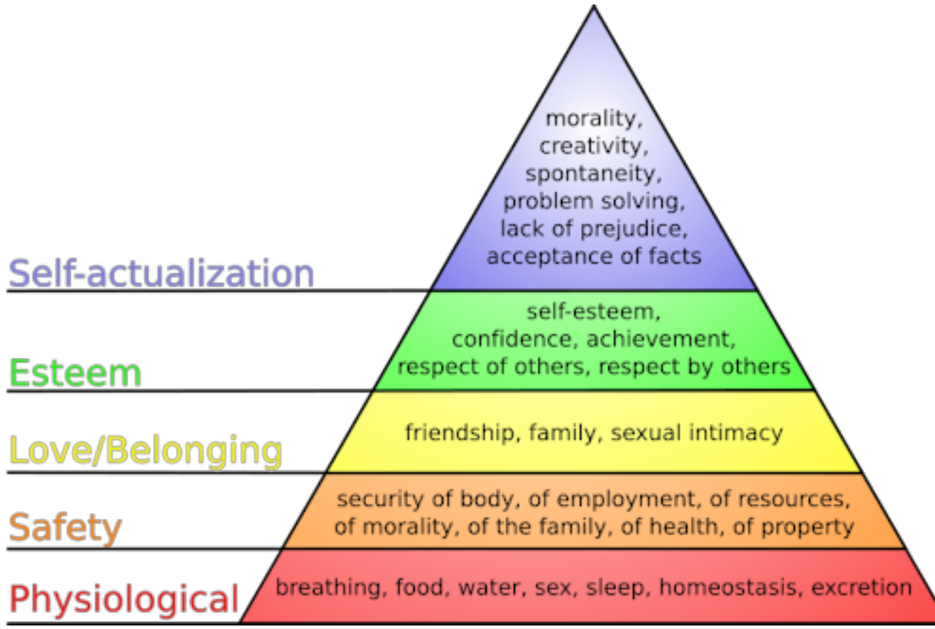
Concept *046 July 18, 2003

- The design process
  - is the act of searching for,
  - specifying,
  - evaluating and
  - selecting design ideas,
  - in an attempt to **satisfy** specified stakeholder requirements.

- Design is finding a set of solutions (design ideas) for a set of defined requirements.

“Satisfy”: design process tries to

- ***satisfy*** is intended in the broadest sense.
- It means there is a *discernible relation* between some **design artifacts**, and some **requirements** –
- and that the purpose, intent, or at least actual effect of the design artifacts is
  - to some degree
  - to impact some performance levels, in the direction of goals,
  - and/or to avoid violating or threatening some constraints.
- There is **no notion of full satisfaction** or optimization implied or intended here.
- The degree of satisfaction actually delivered will be **limited** by priorities, resources and technology.
  - And the satisfaction will vary in time, as requirements change, and the system environment changes



Self-actualization
morality, creativity, spontaneity, problem solving, lack of prejudice, acceptance of facts
Esteem
self-esteem, confidence, achievement, respect of others, respect by others
Love/Belonging
friendship, family, sexual intimacy
Safety
security of body, of employment, of resources, of morality, of the family, of health, of property
Physiological
breathing, food, water, sex, sleep, homeostasis, excretion

“Architecture Engineering”

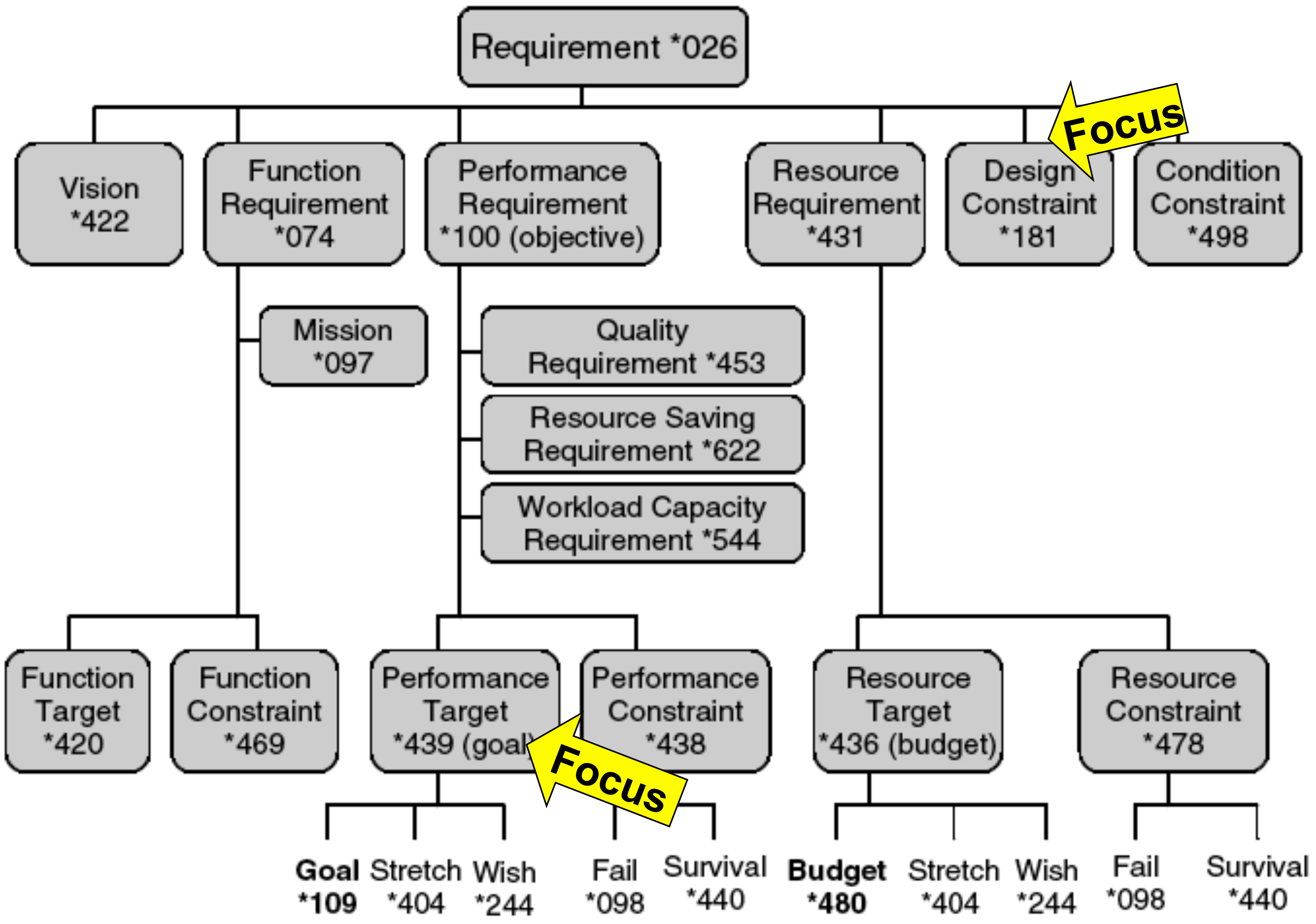
A high level design process

- The <u>architecture engineering</u> process
  - puts in place the systems architecture,
  - which is a controlling mechanism for the **design engineering** of any project.

- Architecture engineering
  - defines the strategic framework (the systems architecture),
    - which design engineering has to work within.
  - It lays down the standards, which control such matters as the tradeoff processes amongst requirements.
  - It helps synchronize design engineering disciplines across different systems.

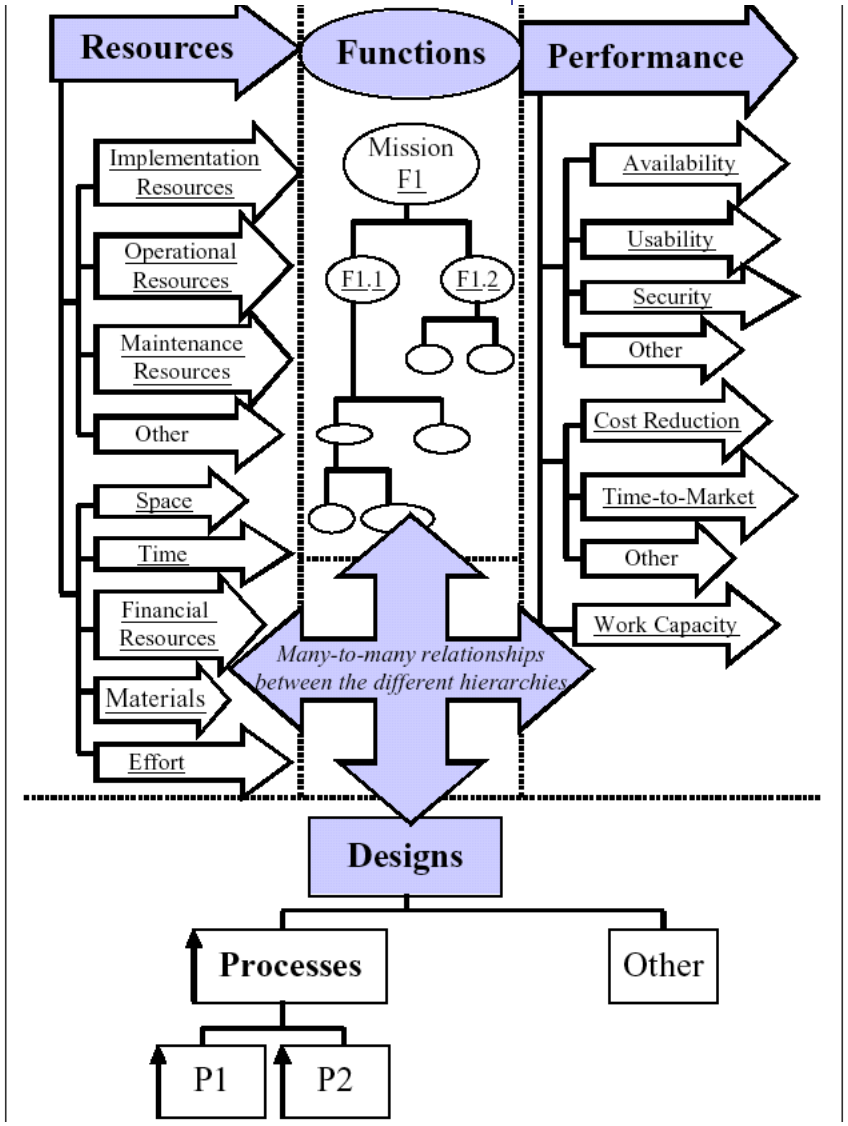
- The architecture engineering process (*499) is a *subset* of the Systems Engineering process (*233).

Requirement Concepts <- CE, page 401, Figure G20, *026



System:

- the "system" is
  - any arbitrarily delineated system
  - or sub-system
  - that anyone chooses to
    - study
    - or deal with
    - that has requirements attached to it
      - formally and informally.



"Stakeholder"

- ***Stakeholders** include*
  - ***any person,***
  - ***organizational grouping***
  - ***or other entity,***
  - ***internal or external to a given development project,***
  - ***of any kind***
  - ***which observably has requirements (performance goals, function or constraints) regarding a system,***
    - ***whether these requirements are known, accepted, formalized, specified or not yet does not disqualify a stakeholder from potentially influencing architecture to satisfy its requirements.***
    - ***This is a much needed generalization of the concept of 'client'. ('Architect satisfies client needs')***

Performance:

- *the attributes of a system*
  - *which describe 'how well' its function is carried out.*
  - *One first level decomposition is into*
    - *work capacity,*
    - *quality and*
    - *savings.*

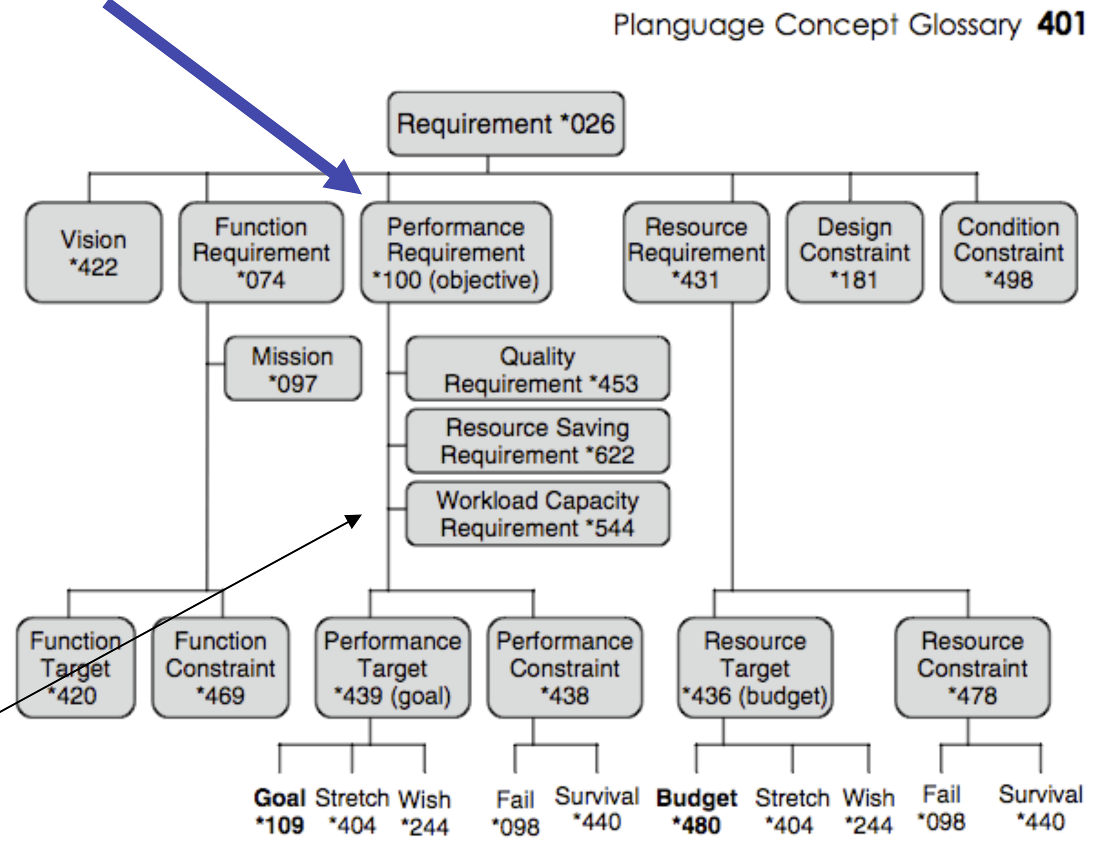
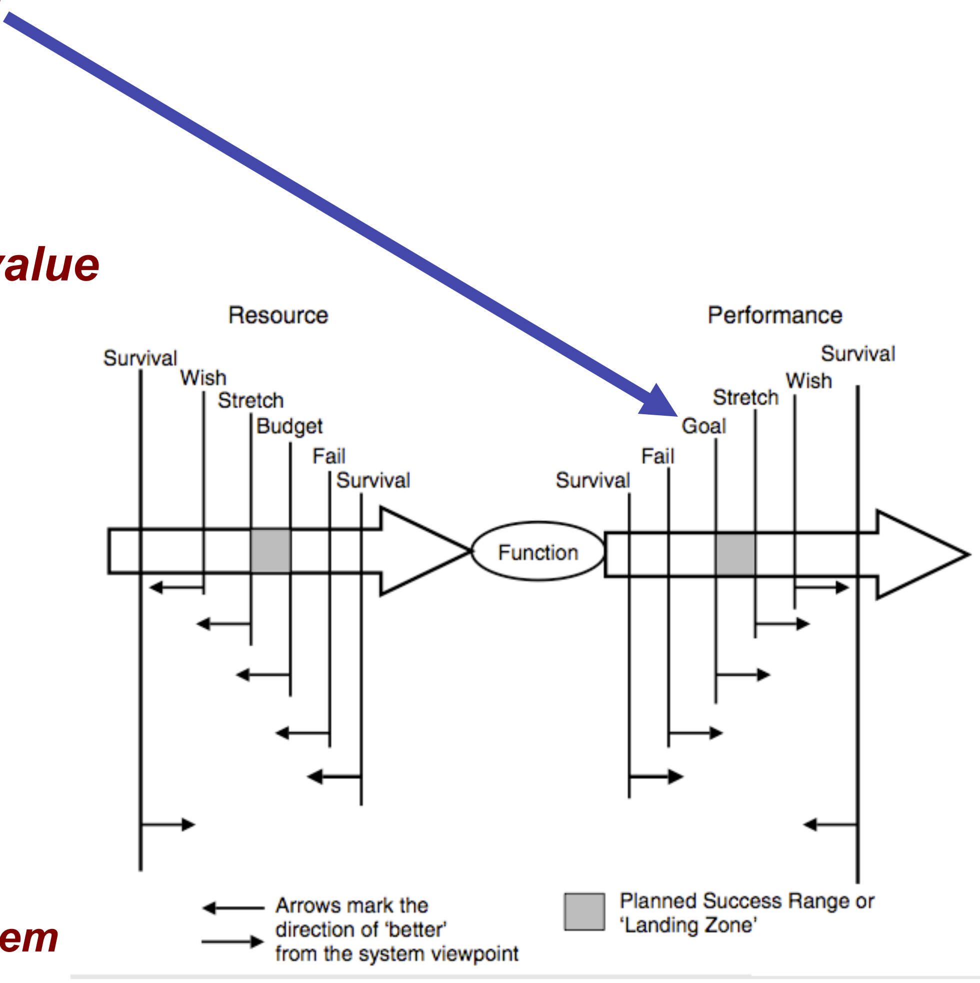

Planguage Concept Glossary 401
Requirement *026
Vision *422
Function Requirement *074
Performance Requirement *100 (objective)
Resource Requirement *431
Design Constraint *181
Condition Constraint *498
Mission *097
Quality Requirement *453
Resource Saving Requirement *622
Workload Capacity Requirement *544
Function Target *420
Function Constraint *469
Performance Target *439 (goal)
Performance Constraint *438
Resource Target *436 (budget)
Resource Constraint *478
Goal *109
Stretch *404
Wish *244
Fail *098
Survival *440
Budget *480
Stretch *404
Wish *244
Fail *098
Survival *440

Figure G20
Requirement Concepts.

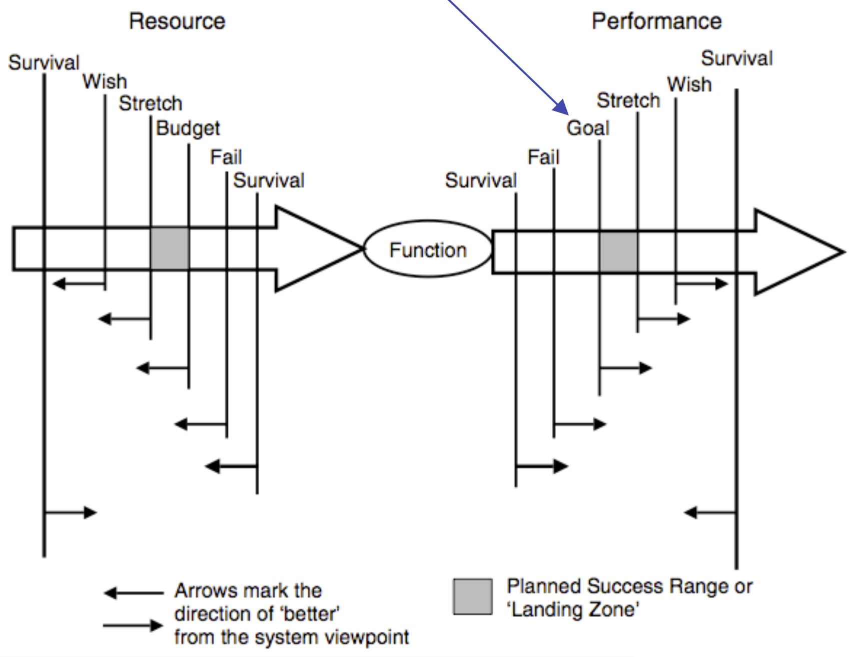
Goals:


Resource
Performance
Survival
Wish
Stretch
Budget
Fail
Survival
Survival
Fail
Goal
Stretch
Wish
Survival
Function
Arrows mark the direction of 'better' from the system viewpoint
Planned Success Range or 'Landing Zone'

- *goals are*
  - *levels of performance*
  - *which some set of stakeholders value and sponsor.*
- *They are*
  - *specifiable levels*
  - *on defined scales of measure.*
- *They are*
  - *the architectural basis*
  - *for judging the need for design artifacts*
    - *to control and <u>enable</u>*
    - *the detailed engineering of a system*
    - *to deliver to those levels*
      - *when and as needed.*

Conditions for A Goal Level
When is a goal level really valid? <-CE 366, *109

1. Technically possible - within state of art
2. Economically Possible - resources exist
3. Costs consistent with other Requirements
4. Effective, and effect necessary to satisfy stakeholder needs
5. Profitable: value over cost
6. Prioritized: by any rules of priority
    1. Effectiveness
    2. Profitability
    3. Politics
7. All [Conditions] in the Goal statement are 'true'


Resource
Survival
Wish
Stretch
Budget
Fail
Survival
Function
Performance
Survival
Fail
Goal
Stretch
Wish
Survival
Arrows mark the direction of 'better' from the system viewpoint
Planned Success Range or 'Landing Zone'

(Quality) Requirements Specification Template with <hints>

HOW WE SPECIFY SCALAR ATTRIBUTE PRIORITY

<name tag of the objective>

Ambition: **<give overall real ambition level in 5-20 words>**

Version: **<dd-mm-yy each requirements spec has a version, at least a date>**

Owner: **<the person or instance allowed to make official changes to this requirement>**

Type: **<quality|objective|constraint>**

Stakeholder: **{ , , }** **"who can influence your profit, success or failure?"**

Scale: **<a defined units of measure, with [parameters] if you like>**

Meter **[<for what test level?>]**

====Benchmarks ============ the Past

Past [ ] **<estimate of past> <--<source>**

Record **[<where>, <when >, <estimate of record level>] <-- <source of record data>**

Trend **[<future date>, <where?>] <prediction of level> <-- <source of prediction>**

===== Targets ============ the future needs

Wish **[ ] <-- <source of wish>**

Goal **[…] <target level> <-- Source**

Value **[Goal] <refer to what this impacts or how much it creates of value>**

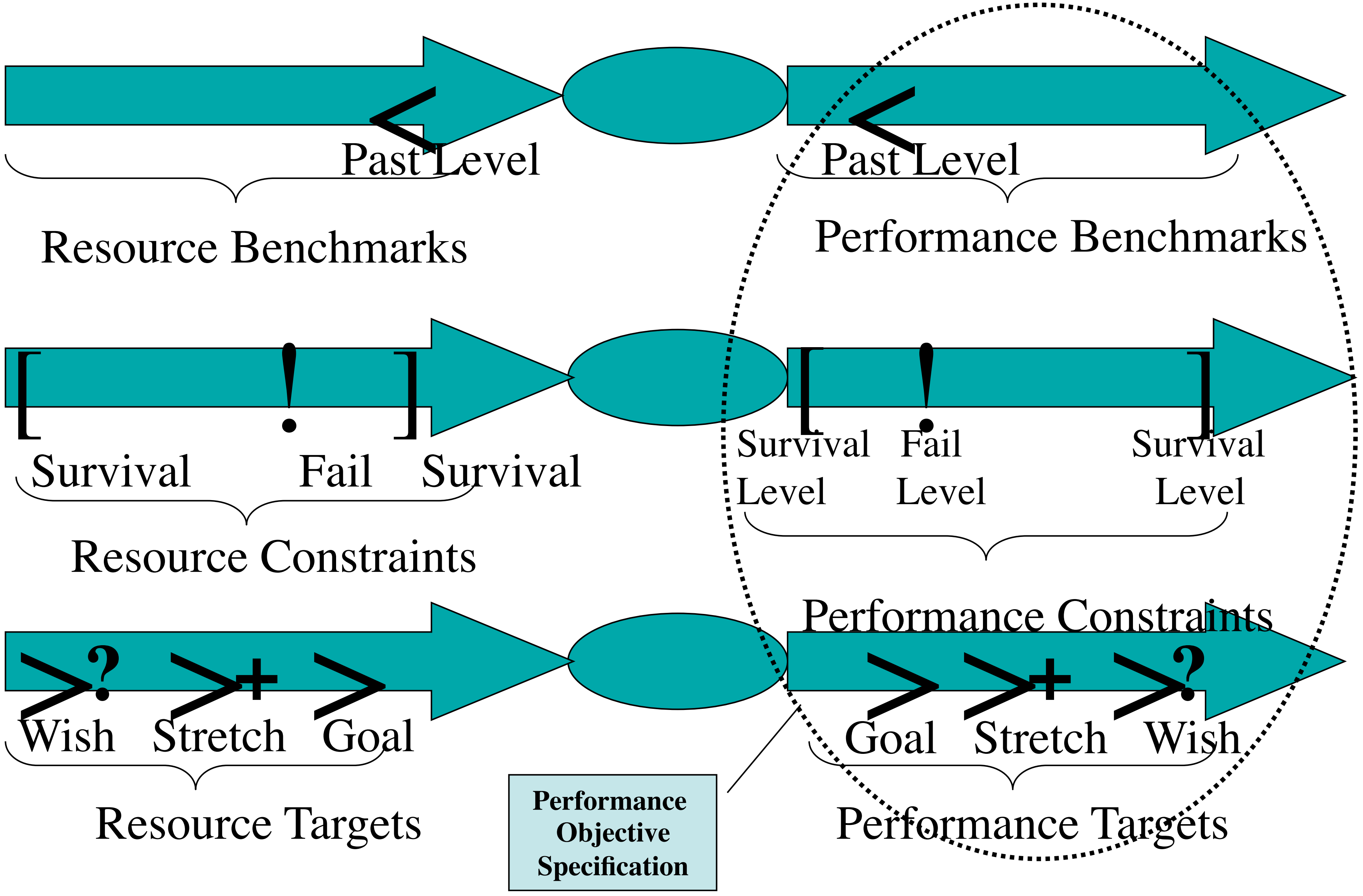
Stretch **[ ] <motivating ambition level> <-- <source of level>**

========== Constraints =======================

Fail **[ ] <-- <source> 'Failure Point'**

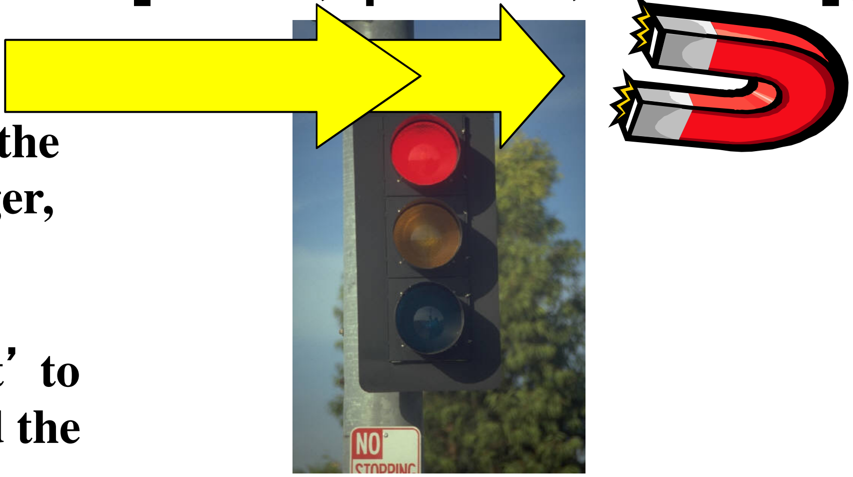
Survival **[ ] <- <source of limit> 'Survival Point'**

Scale Parameter Concepts



Goal (parameter): --->--------> Concept *109. April 7 2002

- A Goal parameter states a future, 'sufficient', performance or budget level requirement, on a defined Scale, under specified conditions [time, place, event], for an attribute.



A Goal acts as a magnet on the designer and project manager,

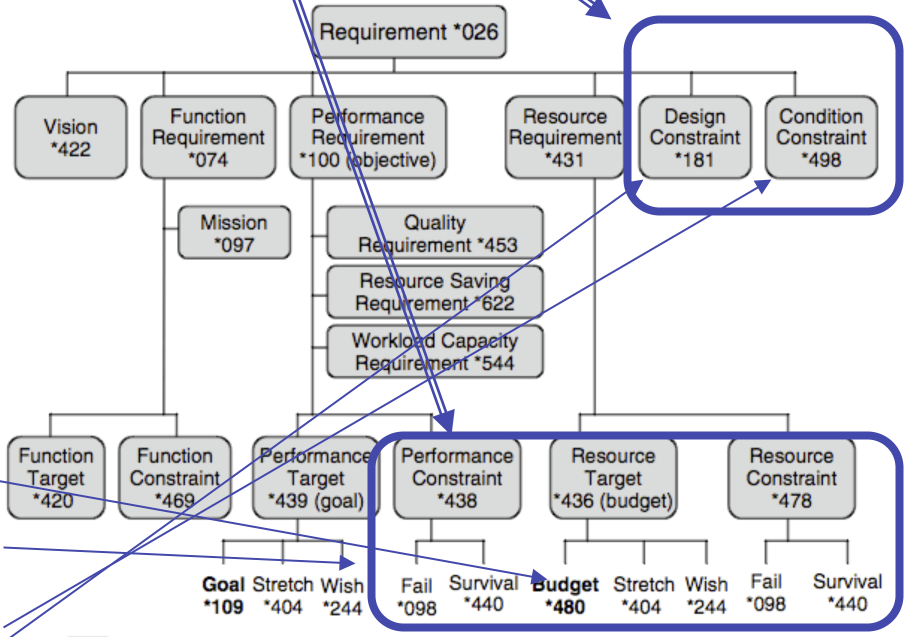
until it is reached.

Then it acts like a 'red light' to stop using resources beyond the Goal level

Constraints:

- *constraints are*
  - *any class of requirement*
  - *which <u>intentionally restricts</u> the freedom*
  - *of an architect or designer of any kind*
  - *to select design artifacts*
    - *either at the architectural level*
    - *or the engineering,*
    - *operational*
    - *Or other life cycle levels*
      - *(such as disposal, or maintenance).*
- *Constraints are of several types,*
  - *and few are absolute*
  - *all can be judged for their relative priority and traded off.*
- *The major types of constraints are*
  - *<u>resource budgets</u> (including budgeted levels and worst case levels)*
  - *<u>performance constraints</u> (worst acceptable levels of any performance attribute)*
  - *<u>restrictions</u> (things the system must not do)*
  - *<u>demands</u> (things the system must do)*
  - *<u>design constraints</u> (any restrictions regarding design which are inputs to a given level of architecture).*

Planguage Concept Glossary **401**



Requirement *026
Vision *422
Function Requirement *074
Performance Requirement *100 (objective)
Resource Requirement *431
Design Constraint *181
Condition Constraint *498
Mission *097
Quality Requirement *453
Resource Saving Requirement *622
Workload Capacity Requirement *544
Function Target *420
Function Constraint *469
Performance Target *439 (goal)
Performance Constraint *438
Resource Target *436 (budget)
Resource Constraint *478
Goal *109
Stretch *404
Wish *244
Fail *098
Survival *440
Budget *480
Stretch *404
Wish *244
Fail *098
Survival *440

Figure G20
Requirement Concepts.

"Constrain"

- *means that the requirements,*
  - *if known or perceived in any way,*
  - *limit the ability of the architect to choose design artifacts,*
  - *and impose upon the architect*
    - *the necessity of designing artifacts*
    - *which limit the ability of other design engineers*
    - *to avoid satisfying requirements.*

"Influence"

- *means that the requirements are somehow taken into consideration,*
- *even if they are prioritized so low that their real influence is at one given moment zero.*
- *They may have the potential to be reconsidered*
  - *later and*
  - *under different circumstances.*
- *They are possibly latent later in the system life cycle.*

"Related (Engineering Decisions)"

- *these include*
  - *all other architecture and requirements decisions*
  - *decisions by any engineering specialty*
    - *or other decision-making entity*
    - *that is controllable by the architectural level of decision-making*
      - *to any degree*
      - *by any means.*
  - *Decisions made after initial system delivery*
    - *by any other entities*
    - *which can influence the attributes of the system*
    - *or some offspring of it.*
    - *These specifically include*
      - *customers,*
      - *markets,*
      - *trade associations,*
      - *license holders,*
      - *military alliances,*
      - *trade blocs*
      - *and the like.*

Engineering Decisions:

- *are decisions*
  - *by any engineering process,*
  - *scientific or art,*
  - *about any notion of design artifact*
  - *intended to influence the outcome*
  - *according to their level of requirements.*

Interesting specializations

- *Perceivable Architecture: the architecture which*
  - *is somehow directly or indirectly perceivable in a real system,*
  - *as determining the range of performance and cost attributes possible.*
  - *This applies regardless of who, if anyone, consciously specified the architecture design artifacts.*
- *Inherited Architecture: architecture which was not consciously selected at a particular level of architecture activity, but was either:*
  - *incidentally inherited from older systems,*
  - *accidentally inherited from specified design artifacts, specified by architects, managers or engineers.*
- *Specified Architecture: the formally defined architecture specifications at a given level and lifecycle point,*
  - *including stakeholder requirements interpretation,*
  - *architecture specification,*
  - *engineering specification done by this architecture level,*
  - *certification criteria,*
  - *cost estimates,*
  - *models,*
  - *prototypes,*
  - *and any other artifact produced as a necessary consequence of fulfilling the architecting responsibility.*

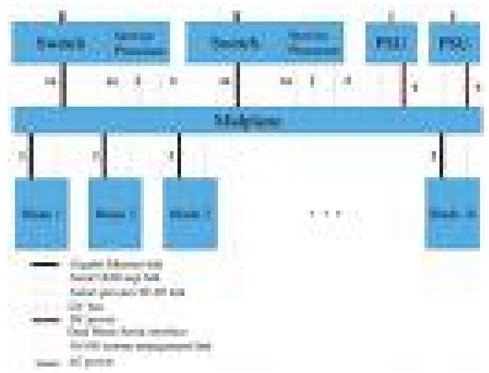
Federal Aviation Definition [Architecture]

- ***Architecture: A high level design that provides decisions about:***
  - ***purpose (What problem(s) that the product(s) will solve)***
  - ***function description(s) (Why has it been decomposed into these components?)***
  - ***relationships between components (How do components relate in space and time?)***
  - ***dynamic interplay description (How is control passed between and among components?)***
  - ***flows (How does data or in-process product flow in space and time?)***
  - ***resources (What resources are consumed where, in the process or system?)***
    - *Source: Standard: FAA-iCMM Appraisal Method Version 1.0 A-19, INCOSE Conference CD, June 1999, Brighton UK [FAA98]*
- *This definition differs from Planguage in that we are primarily concerned with design aspects, and this contains three requirement notions.*

IEEE definition of Architecture

- *Architecture*
  - *The organizational structure of a system or component.*
  - *Source: [IEEE 90] in [SEI-95-MM-003]*

•



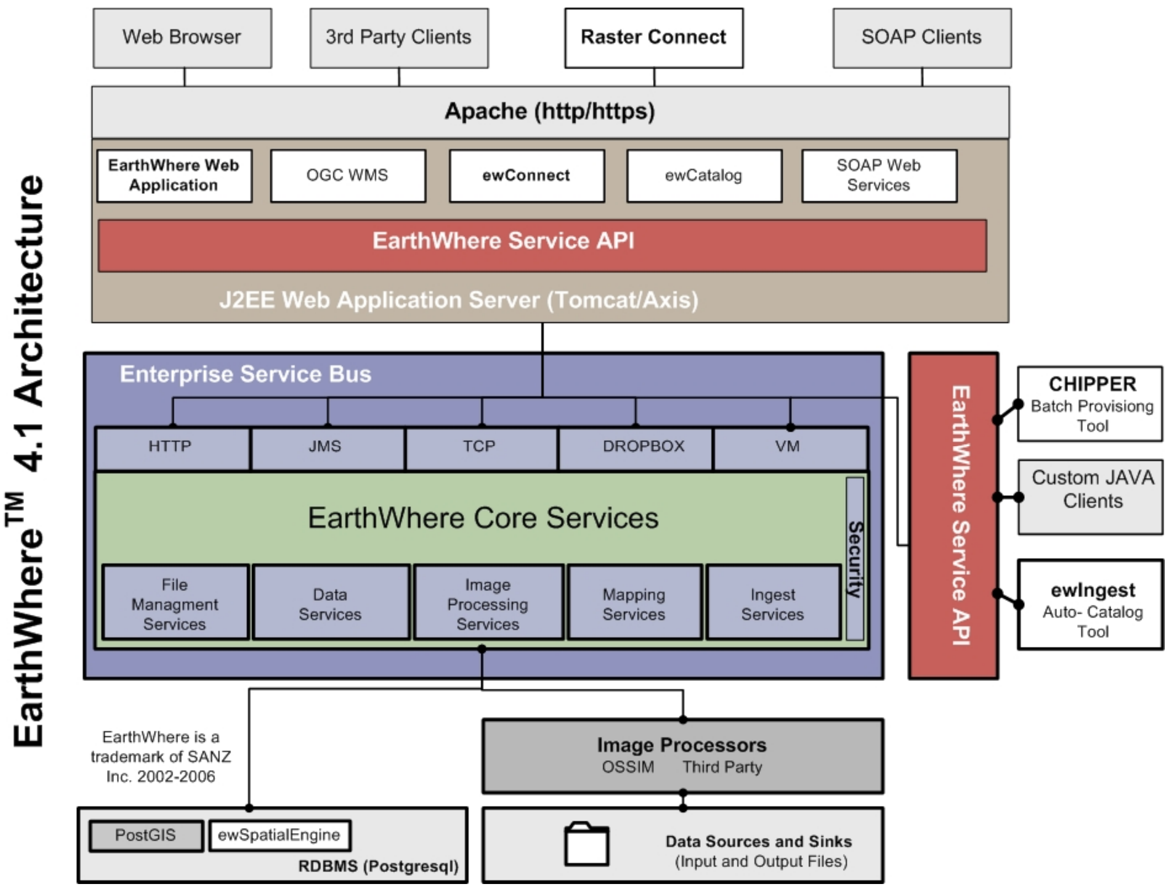
Architectural Description

Concept *618

Architectural description is

– "a collection of products to *document* an architecture."

- This concept is generic and can apply to any specific architecture type.



Architecture Specification

– **Architecture Specification**

Concept *617 June 17, 2003

- An architecture *specification* is the
  - *written* definition
  - of an architectural component.

Defining a Design/Solution/Architecture/Strategy
(Planguage, CE Design Template)
1. enough detail to estimate, 2. some impact assertion, 3. Assumptions, Risks, Issues

Orbit Application Base: (formal Cross reference Tag)

Type: Primary Architecture Option

============ Basic Information ==========

Version: Nov. 30 20xx 16:49, updated 2.Dec by telephone and in meeting. 14:34

Status: Draft

Owner: Brent Barclays

Expert: Raj Shell, London

Authority: for differentiating business environment characteristics, Raj Shell, Brent Barclays(for overview)

Source: <Source references for the information in this specification. Could include people>. Various, can be done later BB

Gist: risk and P/L aggregation service, which also provides work flow/adjustment and outbound and inbound feed support. Currently used by Rates ExtraBusiness, Front Office and Middle Office, USA & UK.

Description: <Describe the design idea in sufficient detail to support the estimated impacts and costs given below>.

> **D1**: ETL Layer. Rules based highly configurable implementation of the ETL Pattern, which allows the data to be onboarded more quickly. Load and persist new data very quickly. With minimal development required. -> <u>Business-Capability-Time-To-Market, Business Scalability</u>
>
> **D2**: high performance risk and P/L aggregation processing (Cube Building). -> <u>Timeliness, P/L Explanation, Risk & P/L Understanding, Decision Support, Business Scalability, Responsiveness.</u>
>
> **D3**: Orbit supports BOTH Risk and P/L -> <u>P/L Explanation, Risk & P/L Consistency, Risk & P/L Understanding, Decision Support.</u>
>
> **D4**: a flexible configurable workflow tool, which can be used to easily define new workflow processes -> <u>Books/Records Consistency, Business Process Effectiveness, Business Capability Time to Market.</u>
>
> **D5:** a report definition language, which provides 90+% of the business logic contained with Orbit, allows a quick turnaround of new and enhanced reports with minimal regression testing and release procedure impact. -> <u>P/L Explanation, Risk & P/L Understanding, Business Capability Time to Market, Business Scalability.</u>
>
> **D6:** Orbit GUI. Utilizes an Outlook Explorer metaphor for ease of use, and the Dxx Express Grid Control, to provide high performance Cube Interrogation Capability. -> <u>Responsiveness, People Interchangeability, Decision Support, Risk & P/L Understanding.</u>
>
> **D7:** downstream feeds. A configurable event-driven data export service, which is used to generate feeds. -> Business Process Effectiveness

==================== Priority and Risk Management ====================

Assumptions: <Any assumptions that have been made>.

> A1: **FCCP is assumed to be a part of Orbit.** FCxx does not currently exist and is Dec 20xx 6 months into Requirements Spec. <- Picked up by TsG from dec 2 discussions AH MA JH EC.
>
> > Consequence: FCxx must be a part of the impact estimation and costs rating.
>
> A2: **Costs**, the development costs will not be different. All will base on a budget of say $nn mm and 3 years. The o+
>
> costs may differ slightly, like $n mm for hardware. MA AH 3 dec
>
> A3:Boss X will continue to own Orbit. TSG DEC 2
>
> A4: the schedule, 3 years, will constrained to a scope we can in fact deliver, OR we will be given additional budget. If not “I would have a problem” <- BB
>
> A5: the cost of expanding Orbit will not be prohibitive. <- BB 2 dec
>
> A6: we have made the assumption that we can integrate Oribit with PX+ in a sensible way, even in the short term <- BB

Dependencies: <State any dependencies for this design idea>.

> D1: FCxx replaces Px+ in time. ? tsg 2.12

Risks: <Name or refer to tags of any factors, which could threaten your estimated impacts>.

> R1. FCxx is delayed. Mitigation: continue to use Pxx <- tsg 2.12
>
> R2: the technical **integration** of Px+ is not as easy as thought & we must redevelop Oribit
>
> R3: the and or scalability and cost of **coherence** will not allow us to meet the delivery.
>
> R4: **scalability** of Orbit team and infrastructure, first year especially <- BB. People, environments, etc.
>
> R5: re Cross Desk reporting Requirement, major impact on technical design. **Solution not currently known**. Risk no solution allowing us to report all P/L

Issues: <Unresolved concerns or problems in the specification or the system>.

> I1: Do we need to put the fact that we own Orbit into the objectives (Ownership). MA said, other agreed this is a huge differentiator. Dec 2.
>
> I2: what are the time scales and scope now? Unclear now BB
>
> I3: what will the success factors be? We don’t know what we are actually being asked to do. BB 2 dec 20xx

Design Spec Enlarged 1 of 2

Spec Headers	**Detailed Description and -> Impacted Objectives**
Orbit Application Base: (formal Cross reference Tag) **Type**: Primary Architecture Option ==== Basic Information ========== **Version**: Nov. 30 20xx 16:49, updated 2.Dec by telephone and in meeting. 14:34 **Status**: Draft (PUBLIC EXAMPLE EDIT) **Owner**: Brent Barclays **Expert**: Raj Shell, London **Authority**: for differentiating business environment characteristics, Raj Shell, Brent Barclays(for overview) **Source**: <Source references for the information in this specification. Could include people>. Various, can be done later BB **Gist**: risk and P/L aggregation service, which also provides work flow/ adjustment and outbound and inbound feed support. Currently used by Rates Extra Business, Front Office and Middle Office, USA & UK.	**Description**: <Describe the design idea in sufficient detail to support the estimated impacts and costs given below>. **D1**: ETL Layer. Rules based highly configurable implementation of the ETL Pattern, which allows the data to be onboarded more quickly. Load and persist new data very quickly. With minimal development required. -> Business-Capability-Time-To-Market, Business Scalability **D2**: high performance risk and P/L aggregation processing (Cube Building). -> Timeliness, P/L Explanation, Risk & P/L Understanding, Decision Support, Business Scalability, Responsiveness. **D3**: Orbit supports BOTH Risk and P/L -> P/L Explanation, Risk & P/L Consistency, Risk & P/L Understanding, Decision Support. **D4**: a flexible configurable workflow tool, which can be used to easily define new workflow processes -> Books/Records Consistency, Business Process Effectiveness, Business Capability Time to Market. **D5:** a report definition language, which provides 90+% of the business logic contained with Orbit, allows a quick turnaround of new and enhanced reports with minimal regression testing and release procedure impact. -> P/L Explanation, Risk & P/L Understanding, Business Capability Time to Market, Business Scalability. **D6:** Orbit GUI. Utilizes an Outlook Explorer metaphor for ease of use, and the Dxx Express Grid Control, to provide high performance Cube Interrogation Capability. -> Responsiveness, People Interchangeability, Decision Support, Risk & P/L Understanding. **D7:** downstream feeds. A configurable event-driven data export service, which is used to generate feeds -> Business Process Effectiveness, Business Capability Time to Market.

Design Spec Enlarged 2 of 2

==== Priority & Risk Management ========

Assumptions*: <Any assumptions that have been made>.*

A1: **FCCP is assumed to be a part of Orbit.** FCxx does not currently exist and is Dec 20xx 6 months into Requirements Spec. <- Picked up by TsG from dec 2 discussions AH MA JH EC.

Consequence: FCxx must be a part of the impact estimation and costs rating.

A2: **Costs**, the development costs will not be different. All will base on a budget of say $ nn mm and 3 years. The ops costs may differ slightly, like $n mm for hardware. MA AH 3 dec

A3:Boss X will continue to own Orbit. TSG DEC 2

A4: the schedule, 3 years, will constrained to a scope we can in fact deliver, OR we will be given additional budget. If not "I would have a problem" <- BB

A5: the cost of expanding Orbit will not be prohibitive. <- BB 2 dec

A6: we have made the assumption that we can integrate Oribit with PX+ in a sensible way, even in the short term <- BB

Dependencies: *<State any dependencies for this design idea>.*

Risks*: <Name or refer to tags of any factors, which could threaten your estimated impacts>.*

R1. FCxx is delayed. Mitigation: continue to use Pxx<- tsg 2.12

R2: the technical **integration** of Px+ is not as easy as thought & we must redevelop Oribit

R3: the and or scalability and cost of **coherence** will not allow us to meet the delivery.

R4: **scalability** of Orbit team and infrastructure, first year especially <- BB. People, environments, etc.

R5: re Cross Desk reporting Requirement, major impact on technical design. **Solution not currently known**. Risk no solution allowing us to report all P/L

Issues*: <Unresolved concerns or problems in the specification or the system>.*

I1: Do we need to put the fact that we own Orbit into the objectives (Ownership). MA said, other agreed this is a huge differentiator. Dec 2.

I2: what are the time scales and scope now? Unclear now BB

I3: what will the success factors be? We don't know what we are actually being asked to do. BB 2 dec 20xx

I4: for the business other than flow options, there is still a lack of clarity as to what the requirements are and how they might differ from Extra and Flow Options. BB

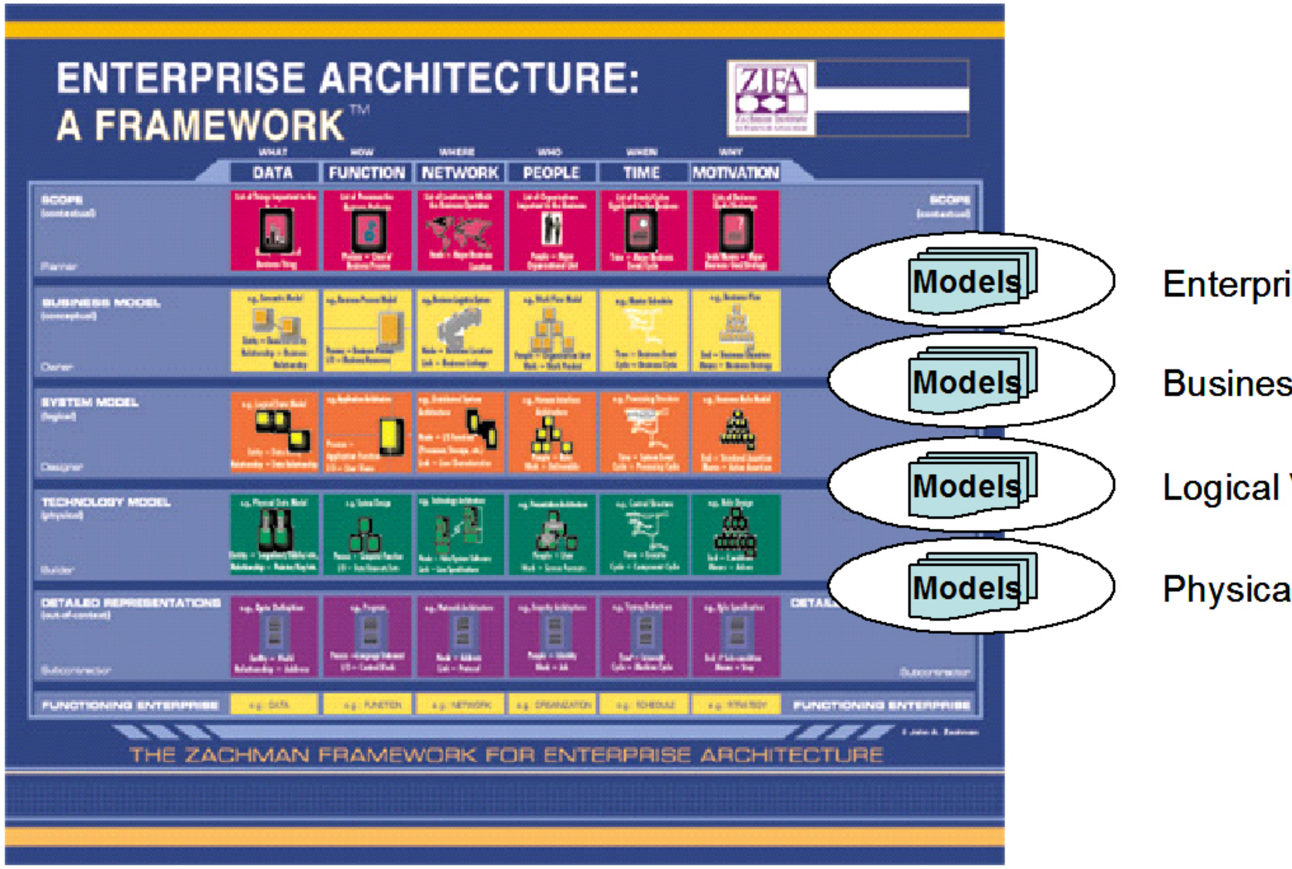
I5: the degree to which this option will be seen to be

Systems Architect

- **Systems Architect**

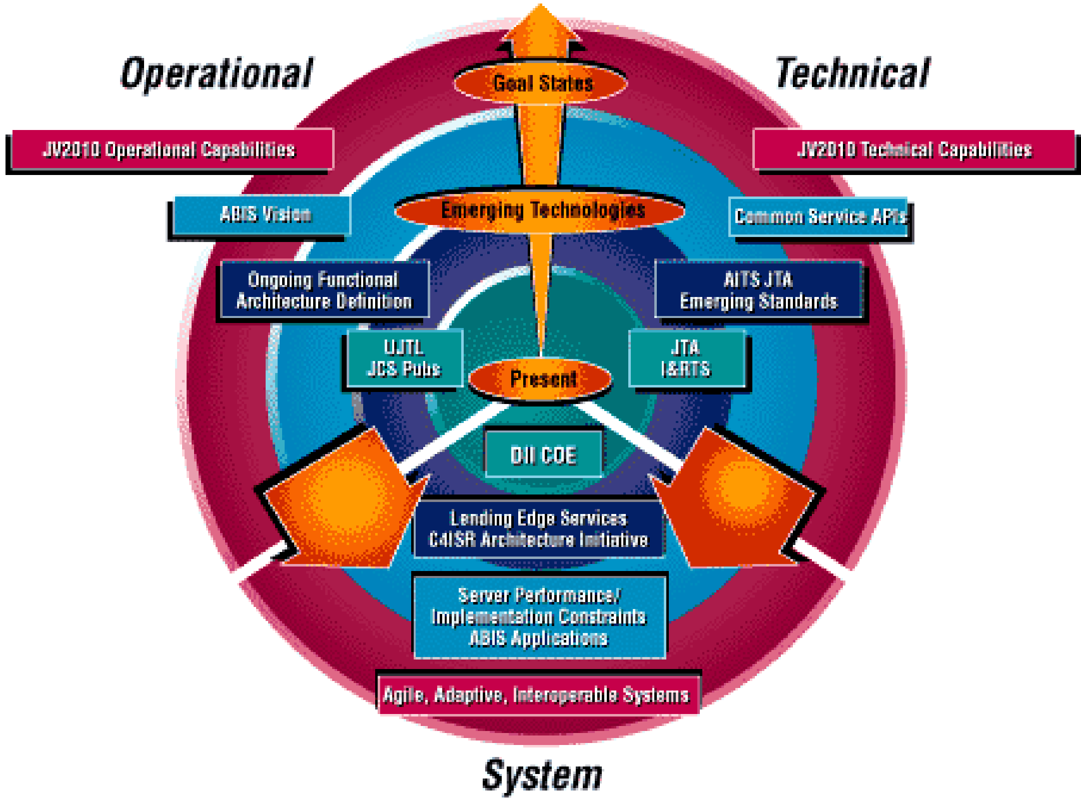
  Concept *193 May 6, 2003

- A systems architect
  - is a person or group,
  - who carries out the work tasks
  - of systems architecture (a process).



Systems Architecture

- **Systems Architecture**
  - **Concept *564 May 28, 2003**
- Systems Architecture is
  - the set of artifacts
  - produced by Architecture Engineering.
- A systems architecture is
  - a strategic framework
  - and consists of
    - models,
    - standards and
    - design constraints
      - specifying mandatory and recommended best practice for implementing and maintaining systems.

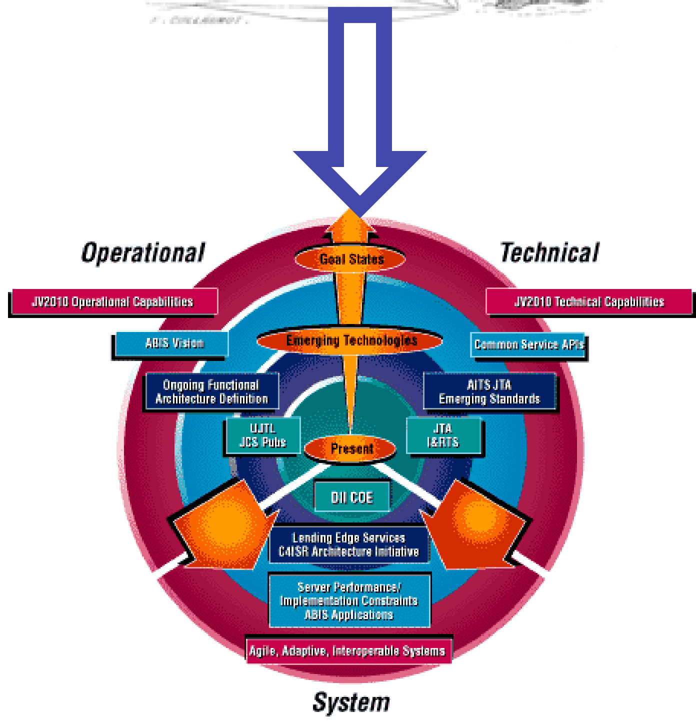

Operational
Technical
Goal States
JV2010 Operational Capabilities
JV2010 Technical Capabilities
ABIS Vision
Emerging Technologies
Common Service APIs
Ongoing Functional Architecture Definition
AITS JTA Emerging Standards
UJTL JCS Pubs
Present
JTA I&RTS
DII COE
Leading Edge Services C4ISR Architecture Initiative
Server Performance/ Implementation Constraints ABIS Applications
Agile, Adaptive, Interoperable Systems
System

Systecture

– **Systecture** © **Gilb**

Concept *564 May 27, 2003

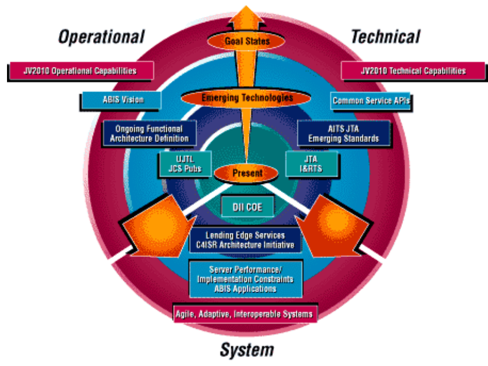
- See Systems Architecture *564.
- Systecture is
  - a conjunction of the term
  - *'system architecture'*.


Operational
Technical
Goal States
JV2010 Operational Capabilities
JV2010 Technical Capabilities
ABIS Vision
Emerging Technologies
Common Service APIs
Ongoing Functional Architecture Definition
AITS JTA Emerging Standards
UJTL JCS Pubs
Present
JTA I&RTS
DII COE
Lending Edge Services C4ISR Architecture Initiative
Server Performance/ Implementation Constraints ABIS Applications
Agile, Adaptive, Interoperable Systems
System

Systect

- **Systect: Concept *565. July 19, 2002**

- A systect is
  - a person who does Systecture
  - (systems architecture) – a systems architect.
  - It is a conjunction (systems architect).


Operational
Technical
Goal States
JV2010 Operational Capabilities
JV2010 Technical Capabilities
ABIS Vision
Emerging Technologies
Common Service APIs
Ongoing Functional Architecture Definition
AITS JTA Emerging Standards
UJTL JCS Pubs
Present
JTA I&RTS
DII COE
Lending Edge Services C4ISR Architecture Initiative
Server Performance/ Implementation Constraints ABIS Applications
Agile, Adaptive, Interoperable Systems
System