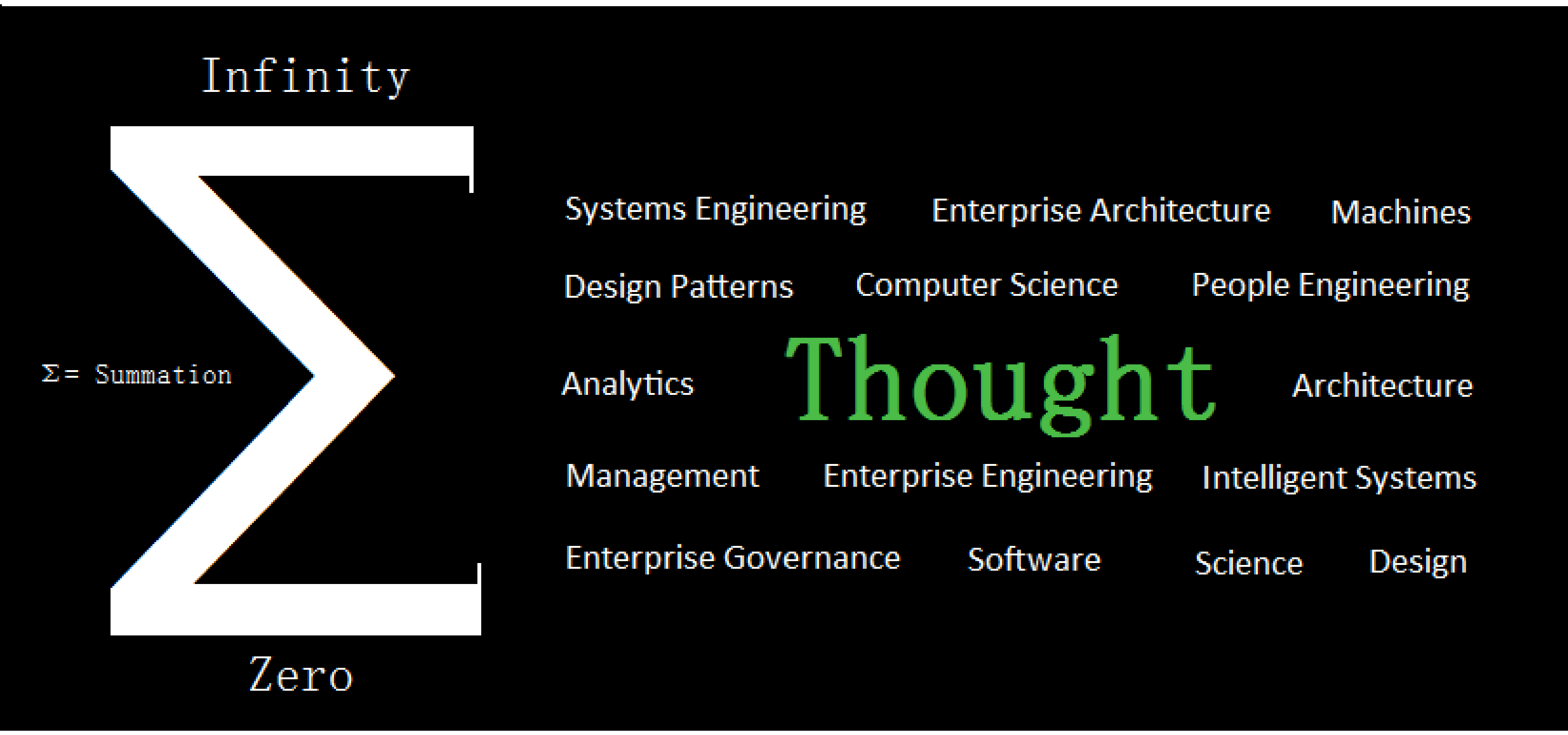
T&C VECTORS

A SMARTER WORLD : The IQ Shift

Breaking the IQ [Intelligence Quotient] Barrier



The Programmed world -THE MATRIX ,

Have you ever stood and stared at it ,Marvelled at it's beauty ,it's Genius ?

A Definitive Guide to Building a Smarter World

[Thought Action Existence]

For those who make their space a serious Discipline !

CHAPTERS

T&C Vectors general introduction

The aim of this handbook is to introduce to the world at large, new systems engineering perspectives. Though the book is intended for a technical audience, a general audience may be in a position to appreciate T&C Vectors point of views and directions of guidance. T&C Vectors encourages readers to appraise this book as a technical exercise, involving activities of de-learning and re-learning. Eventually the book intends to simulate a healthy workout for the mind. The language that's been used is mildly technical to allow an easy comprehension for an audience consisting of students, professionals and accomplished experts.

Note: T&C Vectors is a firm specialized in enterprise engineering. Enterprise engineering is a discipline of systems engineering in that it applies the knowledge of systems engineering to the design, architecture and an efficient working of the enterprise. In summary, the core purpose of the exercise is to add value to the people working within the enterprise and the enterprise itself by using sophisticated learning methods and engineering principles.

The theme can be described as an exercise in building a smarter world.

Chapter 1

The world is a complex space

An analysis of this world quickly reveals the inherent complexities that constitute this real world. The real world is a world created by the accomplishments of humans through the knowledge and experiences gained through scientific thinking and socio-cultural behavior. Our observations however are strictly from an industrial & enterprise perspective and it involves a system's thinking methodology in analysis of this world. The reason why a system's thinking approach is the best qualified approach is because it uni-dimensionally connects and converges disparate or heterogeneous styles of scientific thinking.

By heterogeneous styles of thinking, we mean to suggest the division of information and knowledge into subdivisions of science and engineering for the convenience of comprehension and specializations.

A system's analysis of the world reveals complexities arising from three critical constituents of the enterprise ecosystem:

1. People
2. Work
3. Knowledge and information.

Let's first discuss the **people**'s end of the spectrum. People natively belong to different nations and cultures. They speak a wide variety of languages to communicate.

While we affirm this statement as a fact, it is important to understand how humans developed cognitive faculties over an evolutionary period of time. The human mind uses external symbolic storage and representation as a foundation to language and communication. Once humans had a modern language faculty, they had the potential to apply themselves in building a material culture.

You may understand that symbolic representation's interpretation in the form of a language meant drawing on a lexicon of sounds that we call words and each word standing for a significant idea. The critically important thing about language is that the sentence has more meaning than the sum of the meanings of the words of which it is composed. Words operate more significantly in relation to their individual signified meanings, and there are different ways, grammar and syntax among them which forms the complex web of symbolic relations and eventually, interpretations.

This makes languages a very complex medium of communication. In today's world, the language faculties have been extended to new functional skill sets and specializations, from the industry and enterprise point of view.

Now let's discuss **the work** end of the spectrum where work tasks range from low skilled mundane tasks to extremely specialized industrial tasks.

Work from an industries perspective is classified into a multitude of domains viz., software, aviation, petroleum, engineering, chemicals etc. Each of these domains requires its own level of specializations and skill sets. This makes it virtually impossible for the human mind to think holistically ignoring the specializations.

Therefore, the word specialist reflects a very narrow context of domain. Naturally, this gives an impression to the human mind that the whole world around, consisting of disparate domains is actually a very complex space.

Now let's discuss the **knowledge and information** end of the spectrum. Knowledge as we understand is structured information about learning and inferences gathered from scientific experiments. As in the work spectrum, knowledge exists in a multitude of subjects and specializations.

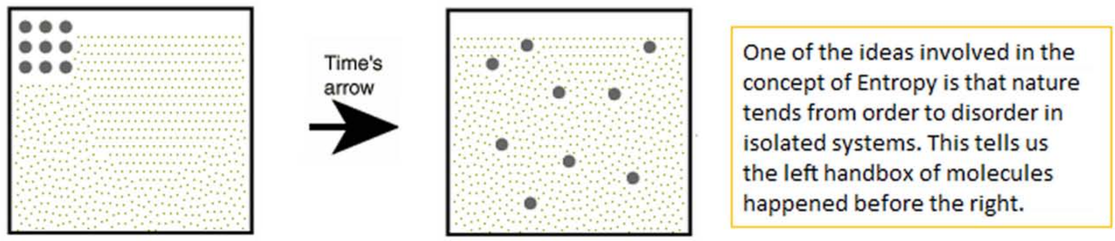
It is because of these inherent complexities of the world space that we use systems engineering technique to systematically simplify perspectives and impressions of the world.

Before we move further, we would like to introduce a scientific term called entropy which theoretically defines a degree of randomness or the disorderliness of a system. This sets us into thinking in terms of real world forces that either add to the disorderliness or tackle disorderliness.

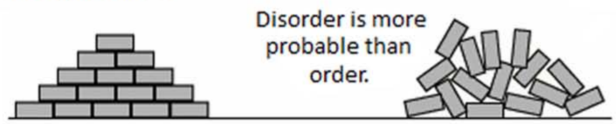
The forces that add to disorderliness can be called as decay forces. Similarly, the forces that fight disorder are called evolutionary forces. E.g. Fragmentation of nation states into smaller territories or emergence of a wide variety of language dialects can be considered as decay forces. Similarly, new inventions, discoveries, information systems and formation of larger nation states like the Eurozone can be considered as evolutionary forces.

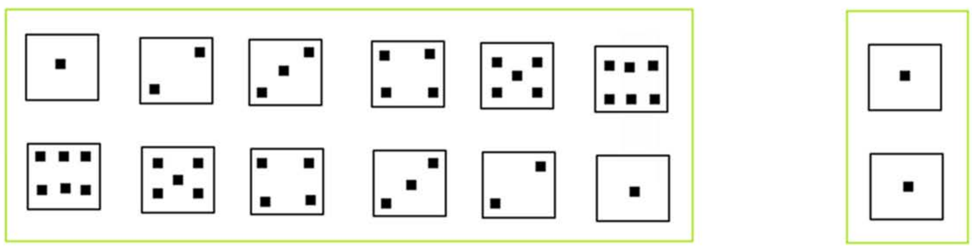
A more precise way to characterize entropy is to say that it is a measure of multiplicity associated with the state of objects and also a test of complexity.

The concept of Entropy - natural disorderness



If you tossed bricks off a truck, which kind of pile of bricks would you more likely produce?





when "throwing dice", throwing a seven is more probable than throwing a two because seven can be produced in six different ways and there is only one way to produce a two. So, we could say seven represents a higher disorder or higher entropy.

If a given state can be accomplished in many ways, it is more probable than one which can be accomplished in only a few ways. E.g. when "throwing dice", throwing a seven is more probable than throwing a two because seven can be produced in six different ways and there is only one way to produce a two. So, we could say seven represents a higher disorder or higher entropy.

Comparing this to the tasks we do at work, some tasks are more disordered because they can be accomplished in many alternative ways.

Why are we discussing this complex world space?
T&C Vectors has been set up to necessarily lay the foundation for a new beginning, wherein the complex world space as discussed can be systematically simplified to allow a simpler and clearer comprehension of the world, in turn instigating forces that are evolutionary in nature, leading to the creation of a smarter world.

The evolutionary forces that we've been discussing can be summed up by a simple word *"change"* – change in thought, action and existence. Physics students may interpret this word in the form of differentiation in world space with respect to change in time.

The objective statement of purpose in the T&C Vectors vision document states, "it intends to solve equations of changing times."

The word change is scientifically called a fluctuation function that occurs before a sea change is observed in the entire world space. In retrospect, consider the time before computers were invented and consider the time today. Consider a fluctuation function like the advent of internet or the world after the invention of search engines.

The reason why we reiterate these change functions is to inculcate patterns of historical consciousness in human mind. Collective memory only simplifies, sees events from a committed perspective and is impatient with ambiguities of any kind. It reduces events to mythic archetypes.

In contrast, historical consciousness by nature focuses on the historicity of events that took place then and not now, that they grew out of circumstances different from those that are now. Memory by contrast has no sense of passage of time. It denies the pastness of its objects and insists on their continuing presence. It is this behavior of the mind that disregards the significance of these change functions that occur over long periods of time.

Historical consciousness helps keep in perspective change functions over a discourse of time and aids in a prepared adaptation to change and gives every significant change a momentous recognition.

Chapter 2

Factorizing complexity by learning to observe

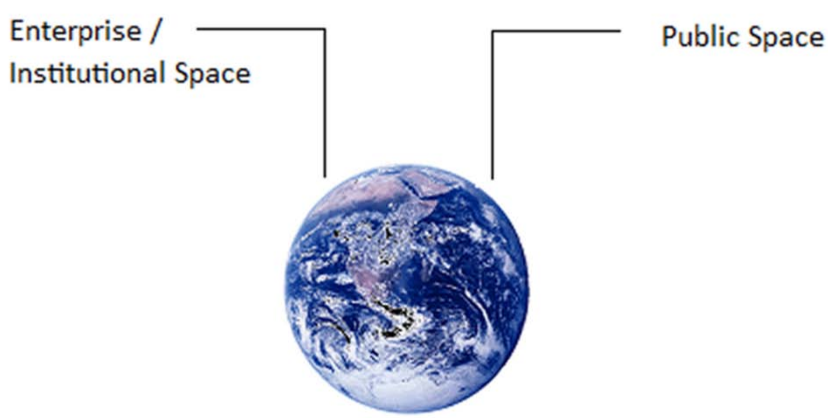
As the chapter name suggests, the goal of this chapter is to factorize the complexity that was discussed in the previous chapter. We used observation as recourse to the factorizing process.

Observation is an activity of assimilation/recording of data through human senses or scientific instruments. Scientific method requires observations of nature to formulate and test a hypothesis.

We also use the process of visualization or a visual paradigm to convey an inference. In this context, our observations would be visually represented using varieties of matrices.

By first observations the world can be divided into two spaces:

1. The enterprise/institutional space
2. The public space.



Enterprise space is the space occupied by individuals carrying out industrial or work activity e.g. factory.

Public space is the space occupied by an individual with respect to his private or personal living space e.g. home.

Our second step to factorizing complexity involves the classification of the enterprise space and the public space into minor entities which we define as factors.

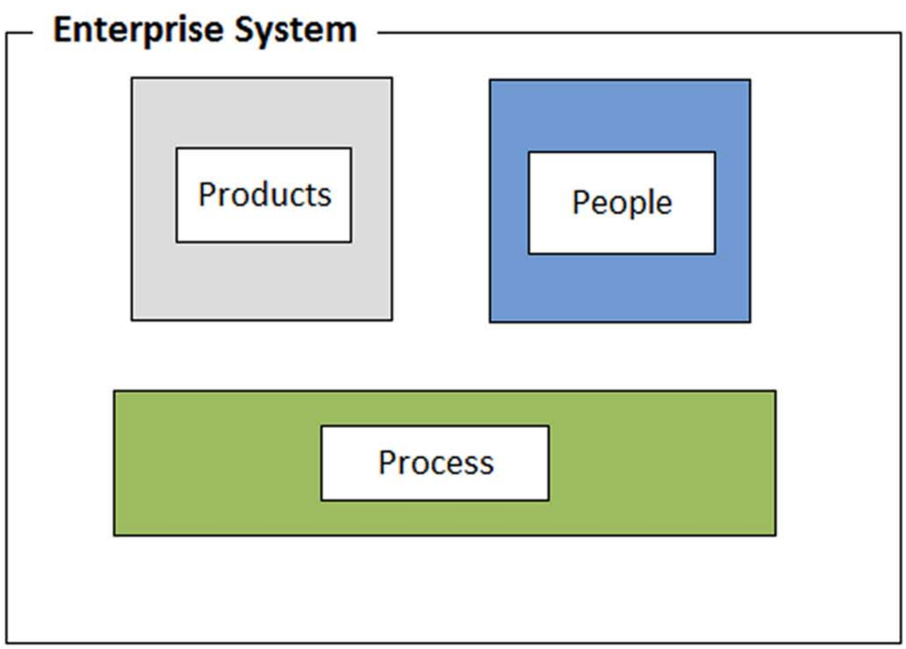
The enterprise space is factorized into the following factors:

1. People - people who work in enterprise/ institution

2. Product/service – products built by enterprises/ services offered by enterprises

3. Process – the methodology of carrying out any particular activity.

Enterprise System Constituent Entities



Public space is again factorized into:

1. People – people in the private/personal living space

2. Process – the process by which people receive information about products/services

3. Products – products that people purchase.

Further, enterprises are of three types:

1. Product/ OEM enterprises
2. Service enterprises
3. Distributor/Trade enterprises.

After our first level of factorization, we're now in a position to draw out the first matrix called the world matrix. The world matrix shows the possible interactions or transactions between the two spaces, the enterprise space and the public space.

World Matrix:

Enterprise space	Enterprise space
Public space	Public space

The inference from the world matrix is that the transactions that occur in the world space can be classified into:

1. Enterprise – Enterprise transactions
2. Enterprise – Public transactions
3. Public – Enterprise transactions
4. Public – Public transactions.

In our next step to factorizing complexity, we draw out another matrix called the transaction matrix. Transactions can be of many types as described below:

Transaction Matrix (Involving People, Processes and Products)

	Products/Service	People	Process
Enterprise to Enterprise	Buy and sell, Distribute.	Support, Knowledge transfer.	General communication channels.
Enterprise to Public	Buy and sell.	Hire and fire.	Advertising/Media/ other channels.
Public to Enterprise	Buy and sell.	X	General communication channels.
Public to Public	Buy and sell, exchange	Communicate, socialize, inform.	In person, socializing forums.

The Transaction matrix describes the possible types of transactions that may occur between the entities of Enterprise and Public spaces.

1. **Enterprise to Enterprise** [PRODUCTS/SERVICES]: This describes the Products/Services that may be bought and sold between enterprises.

2. **Enterprise to Public** [PRODUCTS/SERVICES]: This describes product buy and sell transactions between the enterprise space and the public space.

3. Similarly, **Enterprise to Public** [PEOPLE]: This describes the transactions between the enterprise space and public space wherein people are hired for jobs or fired from jobs

4. **Public to Public space** [PROCESS]: This describes transactions between people through socializing platforms and channels.

5. **Public to Public** [PRODUCTS]: This describes transactions where there is a purchase, sale or exchange of products or services between people.

   The other transaction types are self explanatory.

Analysis of the transaction matrix reveals the following examples of companies, individuals and their activities in the world space:

1. Enterprise to Enterprise [PRODUCTS/SERVICES] - product for use by other enterprises

    a. E.g. IBM – servers for datacenters and enterprises.
    b. GE – Aircraft engines for OEMs like Boeing/Airbus.
    c. CISCO – Networking equipment for enterprises.

2. Enterprise to Public [PRODUCTS/SERVICES] –products for use by people

    a. Microsoft – Windows operating systems.
    b. Apple – Ipods, Iphones, Ipad.
    c. Sony – Laptops, Video cameras.

3. Public to Enterprise [PRODUCTS/SERVICES]

    a. Specialized consultancy services.
    b. Art offerings.

4. Public to Public[PRODUCTS/SERVICES]

    a. Houses, cars - sale and exchange.

5. Enterprise to Enterprise [PEOPLE]

    a. Technical Support.
    b. Knowledge transfer.

6. Enterprise to Public [PEOPLE]

    a. Hire People.
    b. Fire People.

7. Public to Enterprise [PEOPLE]

    a. Not applicable.

8. Public to Public [PEOPLE]

    a. Communicate
    b. Socialize
    c. Inform

9. Enterprise to Enterprise [PROCESS]

    a. Direct communication channels.
    b. Advertising.

10. Enterprise to Public [PROCESS]

    a. Advertising
    b. Media, other channels.

11. Public to Enterprise [PROCESS]

    a. Direct communication channels.

12. Public to Public [PROCESS]

    a. In person – word of mouth.
    b. Socializing forums.

We now introduce you to a term called “Circle of influence”. It is the area or region in which each of the transactions described above have a valid scope.

For example, Sony sells laptops to the entire world. Therefore, its circle of influence is total world. Similarly, companies may be region specific or nation specific. In that case, their circle of influence would be regional or national.

We've now learned how a simplified world looks through the world matrix and the transaction matrix. However, let's not forget this simplification is only a first step. We're yet to deal with the inherent complexities associated with each of the entities of the world matrix or the transaction matrix viz., [Enterprise systems, Public systems, People, Products, Processes].

Let's quickly run an exercise to observe each of the entities described above, from a complexity perspective.

[**People**] – Belong to an array of nationalities, are native speakers of different languages, follow different cultural habits and must still co-exist in the globalised world.

[**Enterprise Systems**] – Exist as multi sized enterprises in a variety of domains. Enterprises exist as protected, public or private enterprises and each having its own level of innovations and specializations.

[**Products**] – Product constructs range from mildly complex to extremely complex systems. They range from a pen or a chair to highly complex systems like gas turbine engines, operating systems, nuclear power plants, aircrafts and naval ships. It may be wise to remember that each of these complex products or systems go through decades of iterative cycles of development in the evolutionary cycle.

[**Processes**] – Processes exist in the enterprise domain as domain specific processes or general enterprise processes. In the people's space, these processes exist as a learning process or a process by which an individual gains awareness about a particular product or service through a media method.

[**Public Systems**] – Consist of heterogeneous spread of public spaces into urban and semi-urban regions. It also consists of different regions having various degrees of public infrastructure and information system infrastructures. It also consists of differential educational systems imparting knowledge and skill for a significant contribution in the enterprise space. It also consists of government systems structured in a variety of ways. E.g. Center controlled or state controlled system. Public systems also consist of a variety of heterogeneous markets.

Chapter 3

Thinking objects & systems

As with our previous chapters, we begin our understanding with an exercise in observation. An observation of the world space reveals that the world around us can be classified into two:

1. Objects
2. Systems.

Observed from a distance, a pen is an object, a calculator is an object and a car is an object. Usually objects can be classified as simple or complex. Simple objects are self explanatory. Complex objects however involve interdependence of other simpler objects forming an integrated whole. Such complex objects are also called systems.

For example, a car is a complex object, but is also referred to as a system. The car's engine is itself a complex object and therefore is called an engine's system. Although computer science teaches us car is an object of the car class, from a system's engineering perspective, the car can behave as an object as well as a system. This is called polymorphicity of objects and systems. Thus the world has many such complex systems viz., an Airplane, a Ship or an industrial enterprise etc.

All complex systems have interactions of object entities as mentioned earlier. Therefore a system can be defined as a function as follows:

System = function of (internal objects, processes)

If we define an engine system, it has many internal objects like cylinders, sparkplugs, pistons etc., all working in a sequence of steps that is called a process.

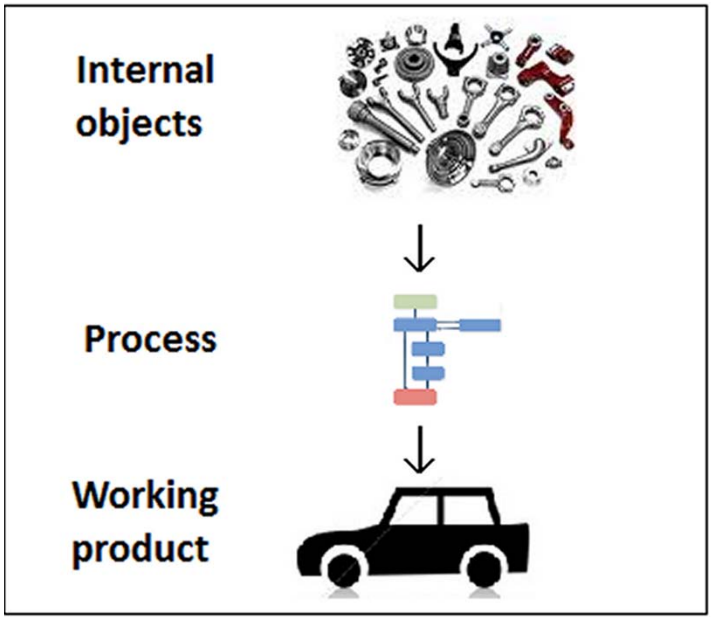
Similarly, in the world of computer science, internal entities work in a sequence of steps to complete a specified process or task.

This principle of systems can be extended to a variety of complex systems.

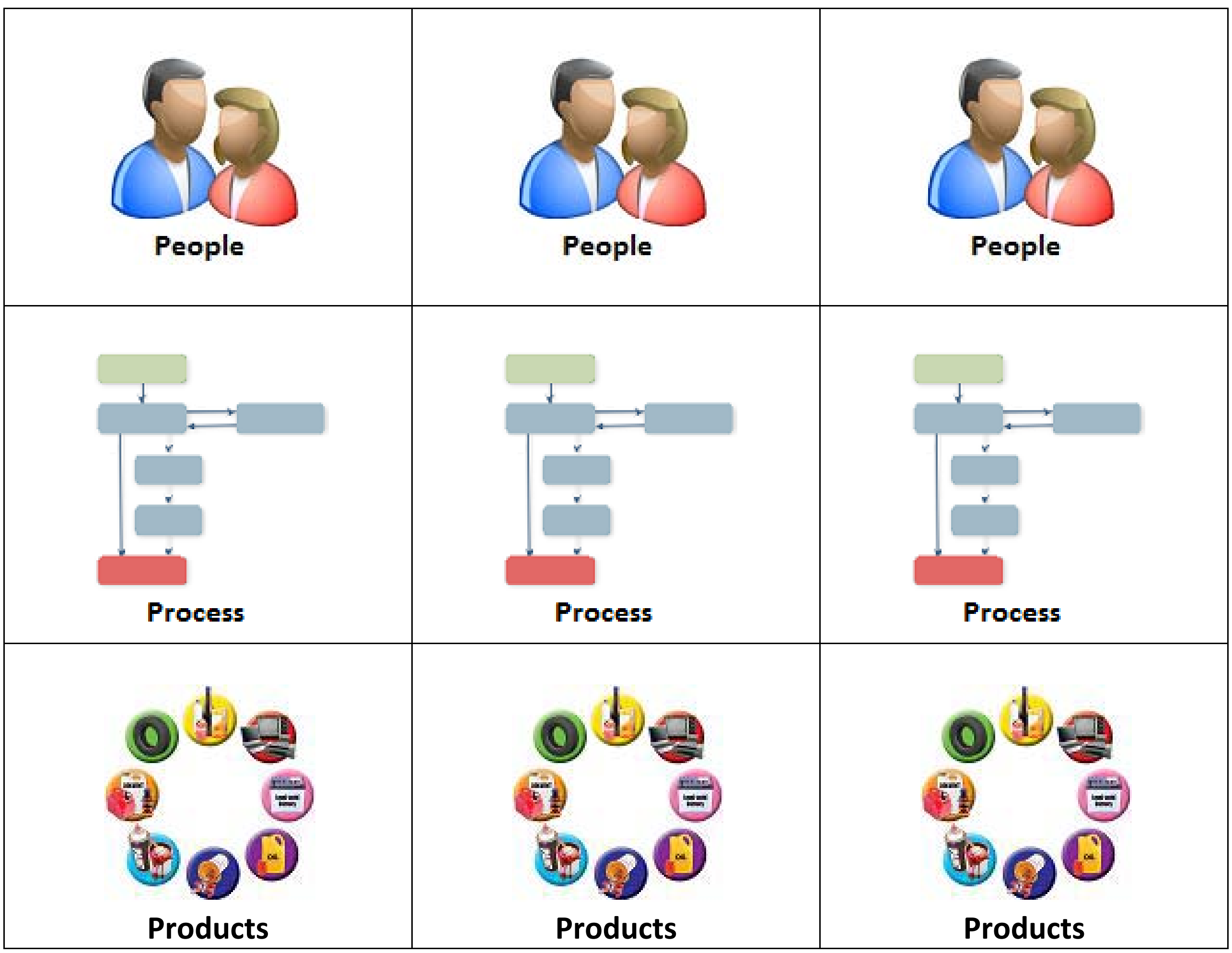
When we extend the system from a machine system to an enterprise system, another additional parameter is added to the function i.e. people. Therefore,

Enterprise System = function of (people, objects[products], processes)

In an enterprise system, people use processes to use products or to develop products.



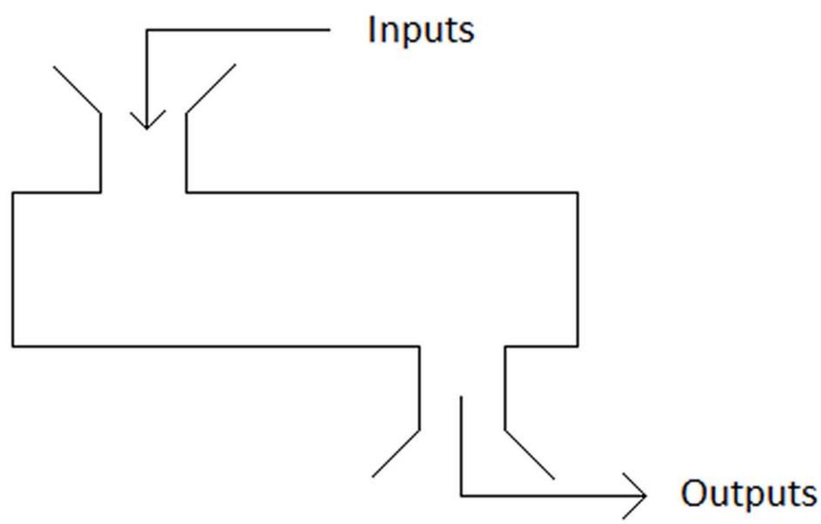
Machine System < Internal Objects, Process >



Enterprise System < People, Process, Products >

For example, in an enterprise system, people use the software development process to develop a software product or people use auto components to produce an automotive engine in a factory.

We’re nearly halfway down being declared as systems engineer but we’re not there yet. Lets study a few working systems to decide a scientific purpose or a design goal of a system.



Every system normally has an input and an output function. The input and output functions depend on the design goal of the system. This implies every functional system belongs to a specific domain and the input and Output depend on the domain.

	Input	Output
Aviation enterprise system	Assembly of Aviation components	Aircraft
Automotive enterprise system	Assembly of Auto components	Car
Petroleum refinery system	Crude oil, refining process	Fuels (petrol, diesel etc.)
Computer mfg system	Electronic components assembly	Computer

Every system has a process and these processes can be classified as:

1. Specialized Processes
2. Generic Processes.

Specialized processes are those processes that are specific to a particular domain, for example, automotive firms have a specialized process for the manufacture of cars. Petroleum refinery has a specialized process for refining crude oil into fuel.

Generic processes are those processes which are common to every kind of industry for example, marketing, sales, human resources (HR) etc.

Science and engineering foundations teach us to build the most efficient systems. In other words, every system has a primary design goal to be able to extract maximum useful work from the system and therefore, act as an efficient system.

We're now in a position to freely use the term enterprise engineering. Enterprise engineering is a discipline of systems engineering in that it applies the knowledge and methods of systems engineering to the system and architecture of enterprises. Its core purpose is to increase the value of:

1. People
2. Processes and
3. Products.

Our goals henceforth would be to:

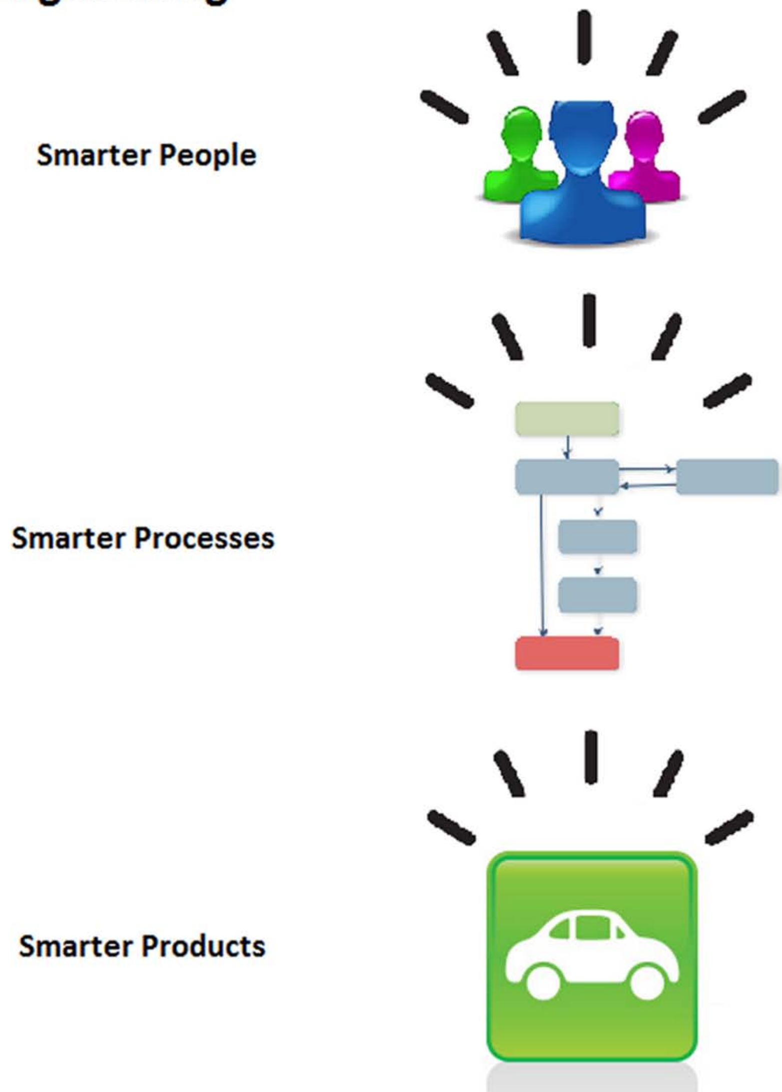
1. Build a smarter world by making or building smarter processes.

2. Making people smarter

    a. The way they learn, think, work.
    b. The way they communicate.

3. Making smarter products or services.

Enterprise Engineering



We've already learned how complex the people, process and the product ends of the spectrum are. So, let's work our way through.

Chapter 4

The Eight Steps to Brilliance

Before we move ahead, we need a ground up course in a function oriented thought. We presume you understand the definition of the term ‘function’.

For example,

Function **Draw Circle (Radius r)** – is a function that draws a circle with input parameter ‘r’.

Similarly, function **Eat (Item I and Weight Xgms)** – is a function that represents the process of consuming a particular food ‘I’ in quantity ‘X’gms.

Similarly, a function **Work (Task T, Hours H)** – is a function that represents a process of doing H hours of Task T.

From these functions, we infer that any activity or task performed by a machine or a human can be represented in a function oriented terminology. You may also infer that this terminology is a departure from the usual language constructs used by humans. It is this deviation in the thought process that differentiates a normal world from a programmed world.

Software professionals and scientific personnel may be aware or acquainted with this style of thought process. Those professionals who are in the software field may further recognize a variety of styles of programmed thinking or functional thinking in the form of a variety of programming languages that are available as an option today.

The programming languages differ from each other on a scale whose one end represents a verbose style and the other end represents a purely functional style.

Example - Visual Basic is a language that’s closer to the verbose end of the scale while the programming language C# is closer to the functional style of the scale.

This re-orientation in cognitive faculty is the first step into building smarter people. We can infer that people who have a background in functional thinking are better suited to be professional programmers.

The functional style of thinking actually disregards fluency in natural languages or traditional symbolic representation of language. It is with this assumption that the measurement of intellectual capacity differs from traditional methods of intellectual capacity measurement whose basis has a very strong traditional language orientation.

Having learnt basic theory of function oriented thinking, we move ahead to learn about a proprietary set of functions introduced by T&C Vectors. This set of functions is called the TEFS [T-Enterprise Free-Scaling Smart] Functions or the INTELLIGENT DESIGN function set. This function set has two representations:

1. Simple

2. Complex (Recursive)

Let us first define the simple function set and understand the meaning of each individual function.

1. **Execution Function** - The process of doing any task or activity. In the context of software development process, execution would mean programming. In the context of automotive industry execution would imply the process of manufacturing or assembling auto components to build a car or automobile.

2. **Design/Structural function** - The process of creating a blue print or a template using creativity, innovation, engineering and science.

   In the context of automobile industry, it would mean creating a template or design blue print of a car or an automobile. In a software context, it would mean architecting an application based on design theory or structural engineering theory. The same principle may be extended to other domains like architecture of a building.

3. **Data/Storage functions** – This function represents the process of storage in a physical world or storage of data in the digital world. In the context of an automotive industry, storage would imply storage of inventory, materials etc. In a digital context like software, information would be represented as data.

4. **Risk function** - The process of assessing risk or mitigating risk in any domain. In the software context, this would mean security functions in a software application or in the context of automotive industry, it would mean safety features and characteristics incorporated into manufacturing of a car or automobile.

5. **Optimization function** – This is the process of completing a task in the most efficient way. In the software context, it would imply optimization algorithms. In

automobiles context, it could mean fuel efficient way of running an automobile engine. From a logistics & transportation point of view, it could mean shortest route delivery channels.

6. **Enabler function** – This is a pre-requisite process or task required to complete the succeeding task. In the software context, it could mean a pre-requisite framework like the .NET framework for the running of an application. In a computer hardware context, an electrical power supply is an enabler for the hardware machine. In the automobile context, the robotics, welding machines and hoists are classified as enablers to the Manufacturing process.

7. **Rules and Boundary Condition Functions** – As the name implies, it is the set of rules or principles which govern the working of any process or system. In the software context, it would imply the business logic and rules of a software application. In the automotive context, the rules would imply the principles of Otto cycle in the functioning of an engine or the physics rules of vehicle stability.

   The boundary conditions imply the maximum permissible values of variables of a system or process. E.g. In the automobile context, it could mean the maximum weight of the vehicle or the maximum dimensions of the vehicle or its maximum speed. In a software application context, it could mean maximum number of users who are permitted to use the application or maximum speed of data flow.

8. **Exception Functions** – These functions represent an unusual occurrence or exceptional event occurring in a process or a system.

We have now completed the description of the simple form TEFS functions or the INTELLIGENT DESIGN functions. Our focus now shifts to what these function sets can be used for in different domains and systems.

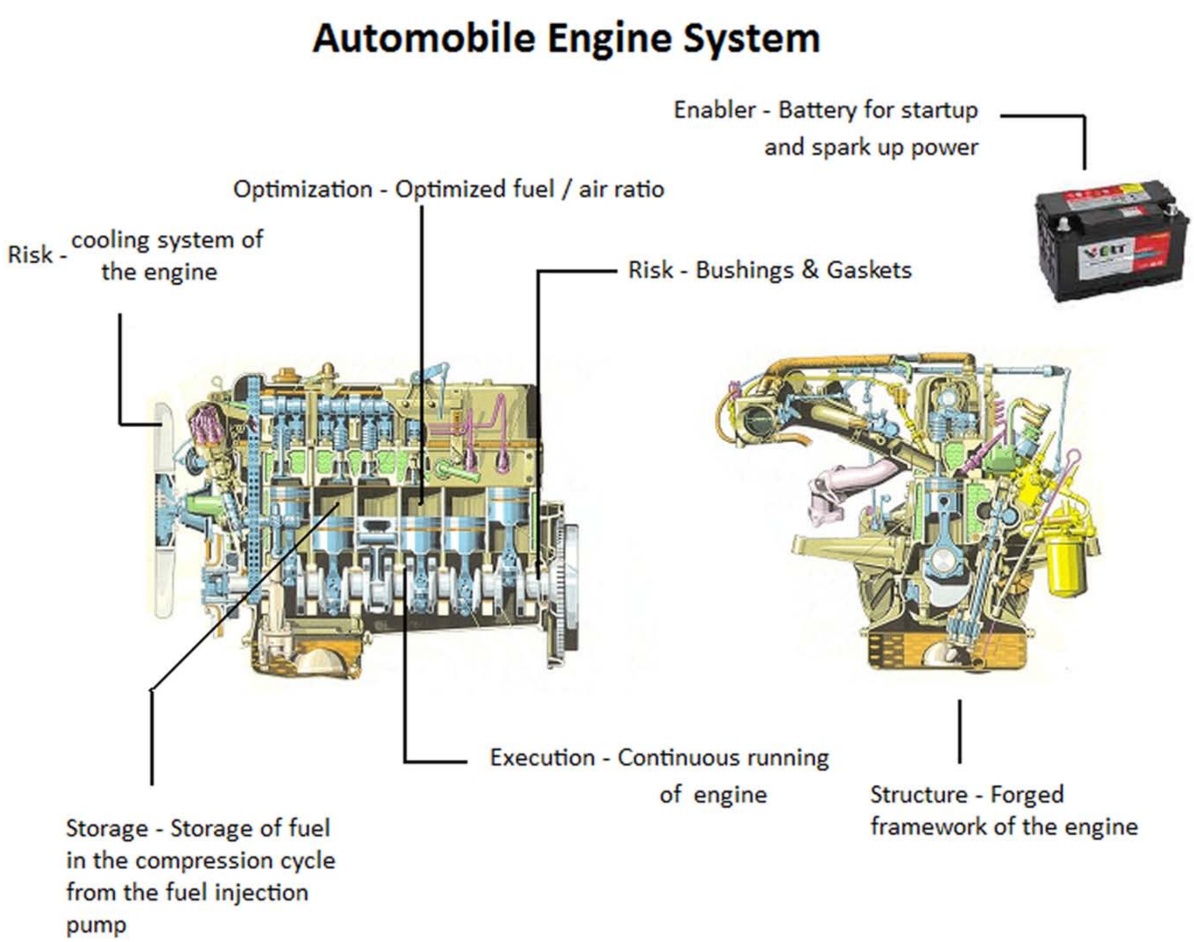
We understand we are in a pursuit to make the people, processes, products and the system on the whole a smarter space. This function set has the power to intelligently define products, people, processes and the system as a whole.

Before we move ahead, describing the complex form of the intelligent design function set, we would study the applications of the simple function set in a variety of domains.

Automotive Engine System:

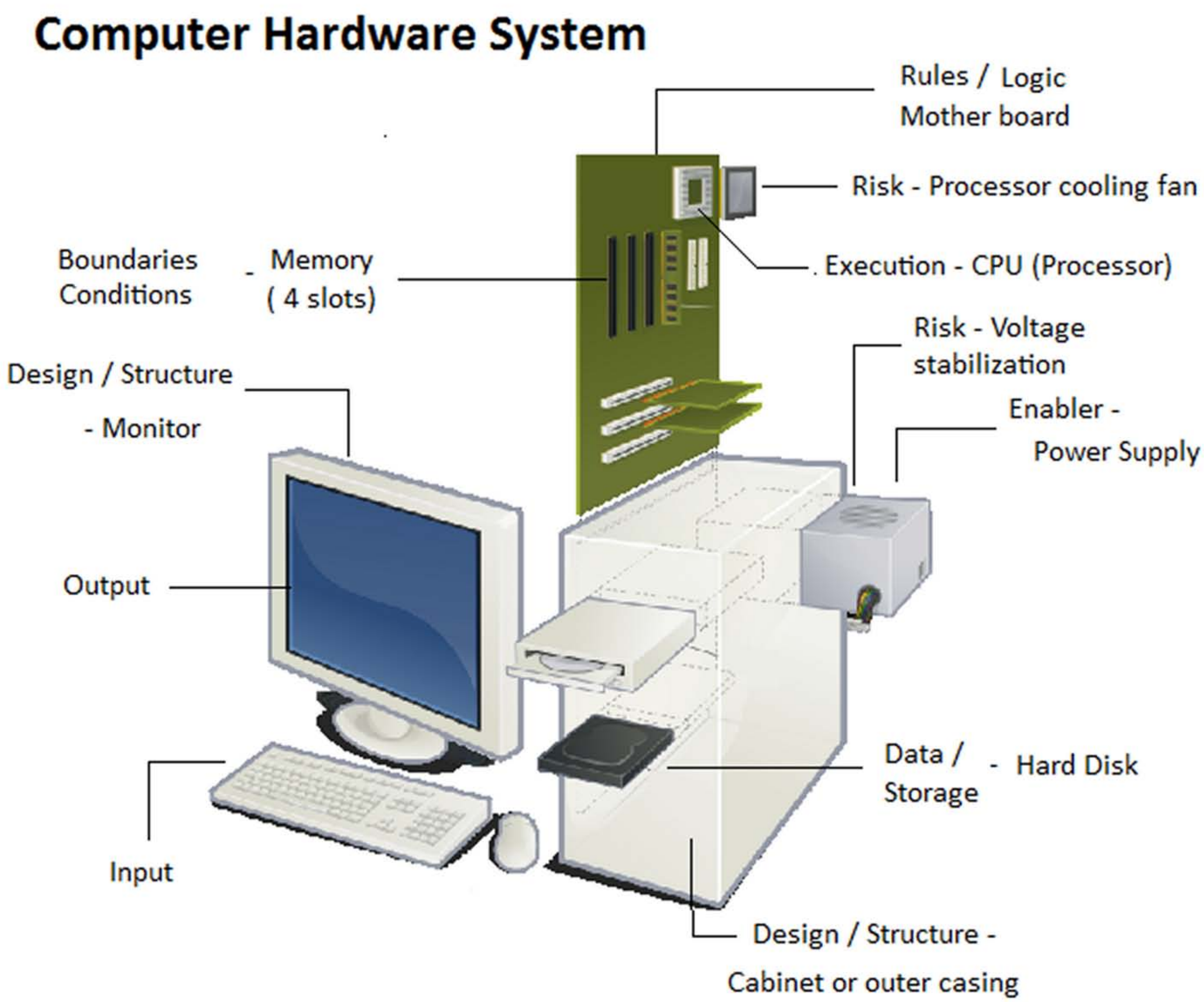
Execution – The continuous running of the engine.
Storage – The storage of fuel during fuel injection.
Structure – The forged framework of the engine where the pistons move up and down in the cylinder generating power.
Risk – Gaskets, bushings, washers to disallow the spillage of oils, hydraulics, fuels etc., or engine cooling system.
Optimization – Optimized fuel/air ratio, optimized sequential process of sparkler timing.
Enabler – Battery power to allow spark up or start up power.
Rules/Boundary conditions – Otto cycle principle/process of combustion using programmable processors. Boundary condition implies engine capacity of 1.2Ltr or 1.4Ltr.
Exceptions – Exceptional heat or temperature rise in the engine or exceptional fuel consumption.

The functions and the examples that we have described above are for a sample purpose only. A real engine function set would contain far more functions.



Computer Hardware Systems:

Execution - Operation of the computer processor for computation and calculation.
Design /Structure – The structure is represented by the cabinet or the outer casing.
Data/Storage – Storage of data is in the hard disk or in the memory, in the physical computer system
Risk - Cooling systems for the processor and the electronics. Or regulated voltage stabilization or RAID - replicated data in duplicate disks.
Optimization – Optimization of electrical power consumed by the processor, optimized storage and indexing of data in hard disks.
Enabler – Power source (AC source)
Rules/Boundary Conditions – Rules and logic are governed by the mother board or an Operating System to run the computer by management of memory/process scheduling.
Exception – Memory faults, hard disk failure, processor overheat.



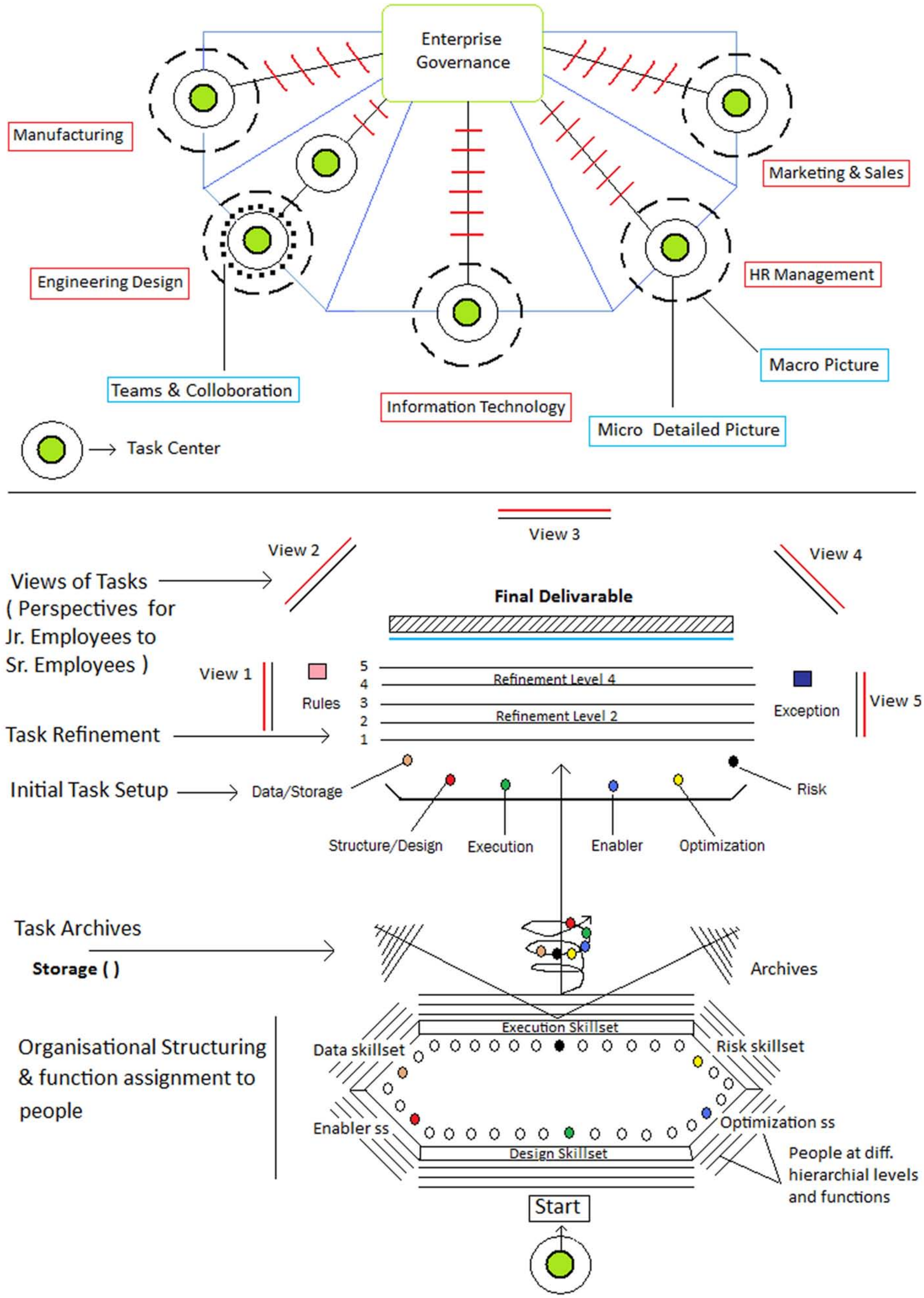
We have now seen two examples of using the TEFS Functions or the INTELLIGENT DESIGN Functions to define the working of two machine systems. A similar principle can be used to describe software systems and a variety of other systems.

We would now move ahead to learn the definition of an enterprise system.

Enterprise Systems:

Execution – Manufacturing/Product development by the people working in an enterprise.
Design/Structure – design of the enterprise building or organizational structure or the hierarchy of people.
Data/Storage – Storage involves storage of inventory, materials and finished products. Data involves information related to people, tasks, products etc.
Risk – Information security or production quality and physical security of the enterprise.
Optimization – Efficient process of manufacture or development of products e.g. Lean manufacturing.
Enabler – Office working space or working capital.
Rules – Enterprise, governance, rules and policies.
Exceptions – Unusual incidents in the enterprise space.

We can now represent this function set in a pictorial diagram representing the enterprise system:



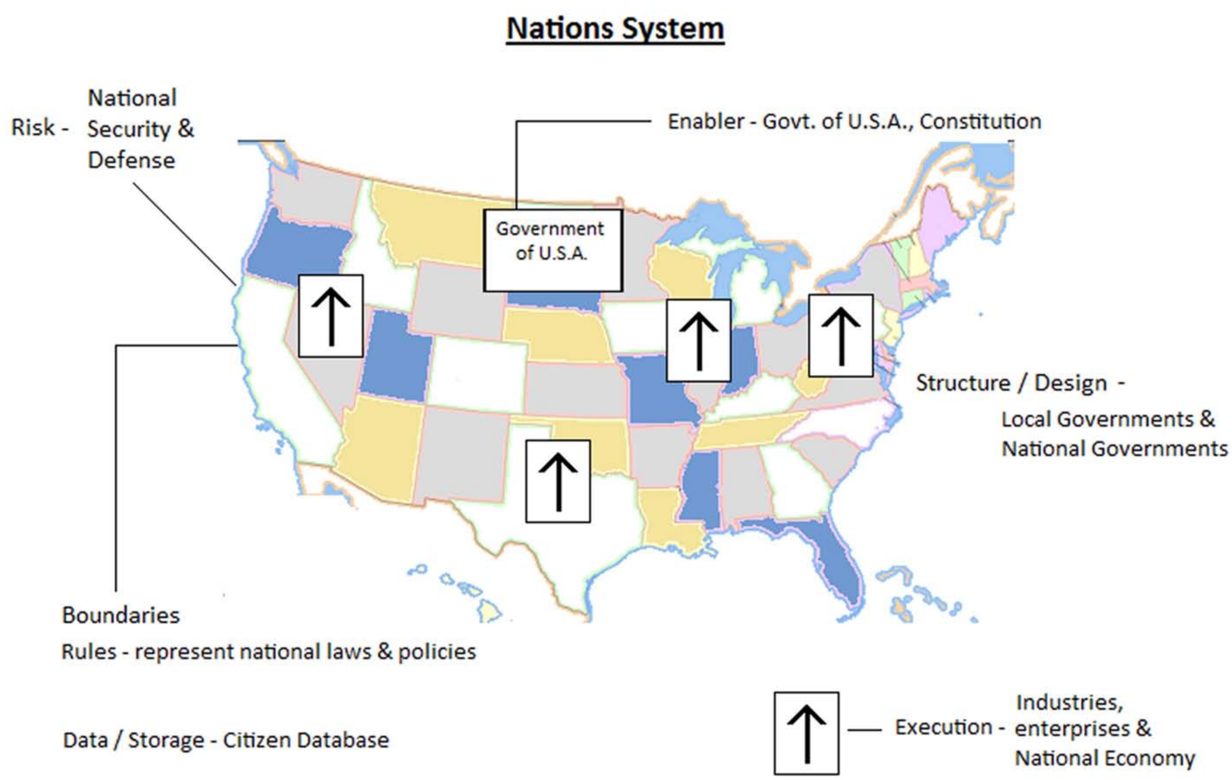
<u>Enterprise workflow system definition</u>

We now extend this intelligent design function set to define a nation system.

Nations Systems or Nations expressed in systems parlance:

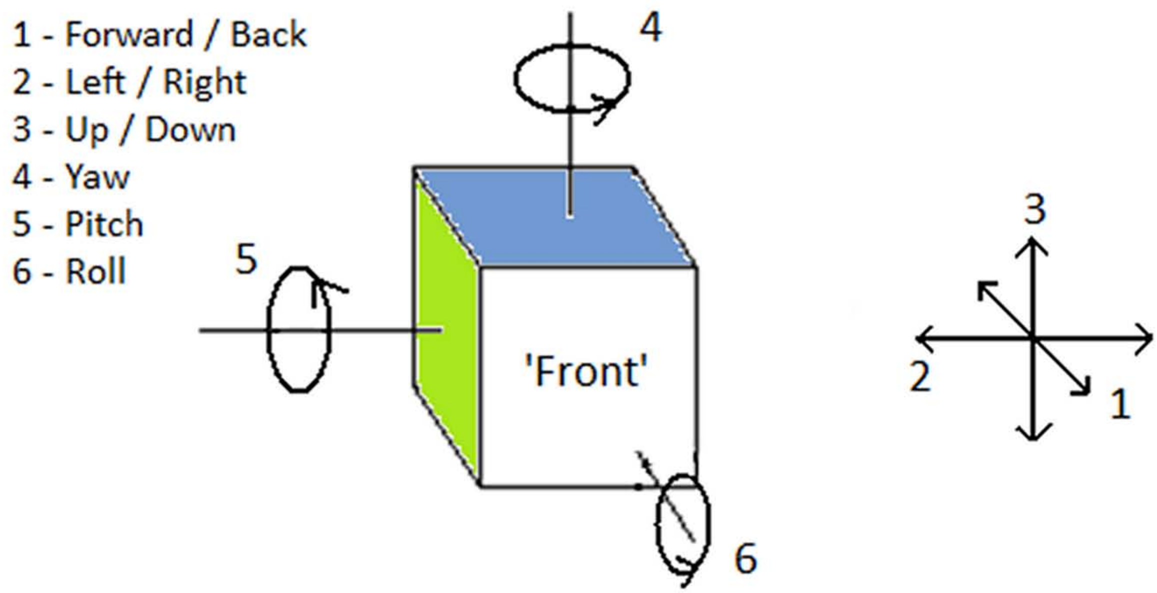
Let us take a country or a nation on earth. E.g. USA.

Execution – is represented by Industries, enterprises and national economy.
Design/Structure – Division of nations into nation states with governance at national level as well as local level.
Data/Storage – All the information related to citizens, establishments etc.
Risk – National security and defense services.
Enabler – Government frameworks and constitutions.
Rules/Boundary conditions – Geographical boundaries of the nation and various national laws for the public and private space.
Optimization – Optimization of industrial space, housing space and forest space.
Exceptions – Untoward incidents, Recessions or wars (External or internal).



We have worked out system definitions at machine level, enterprise level and Nation systems level. The same principles may be used to define systems of any other kind. Mathematics students may understand this approach through the principles of mathematical induction.

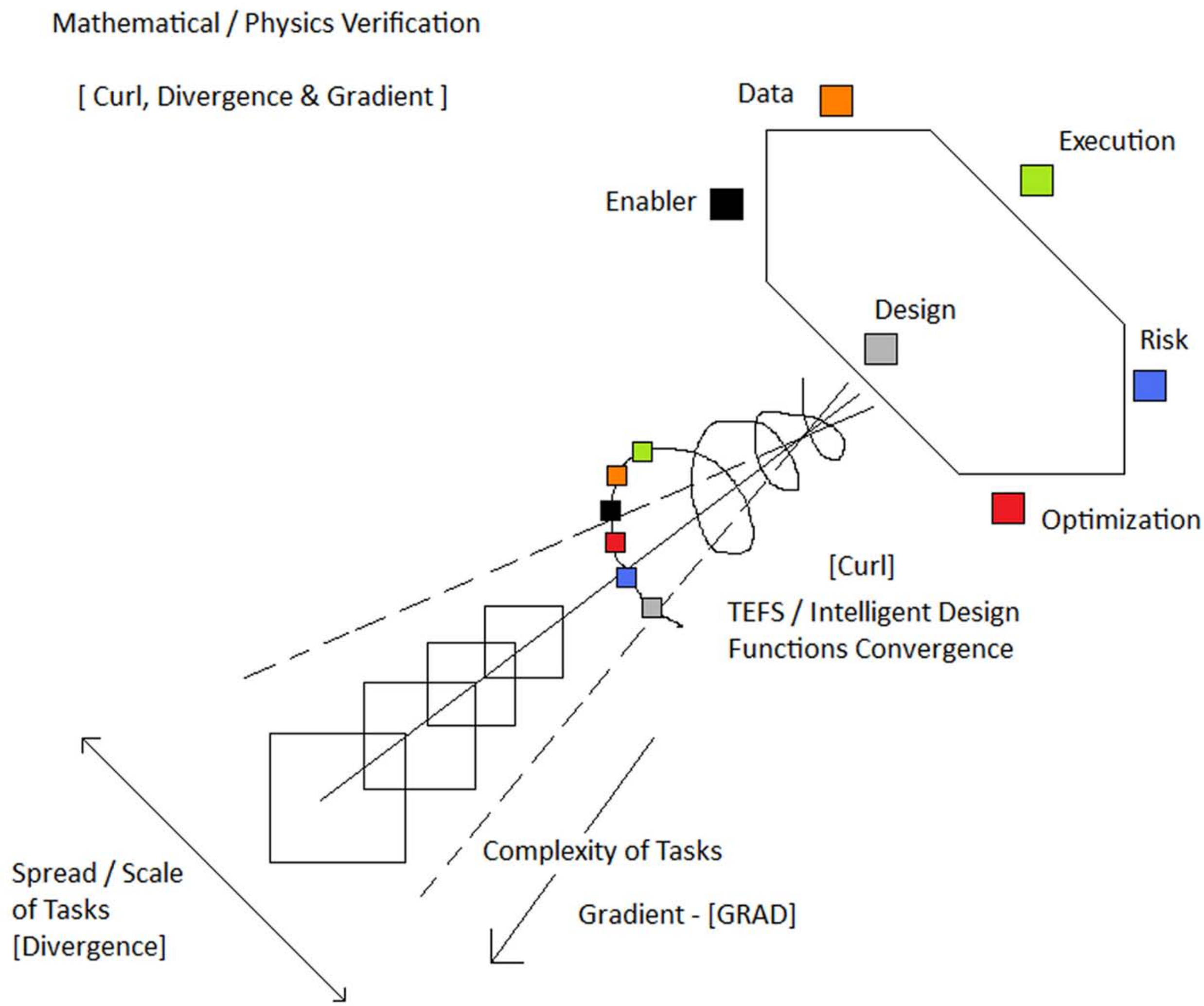
Physics students may be aware of the concept of six degrees of freedom which refers to the motion of a rigid body in three dimensional space viz., the ability to move forward/backward, up/down and left/right combined with rotation about three perpendicular axes (pitch, yaw, roll).



The Six Degrees of Freedom

Similarly any system whether a machine system or an enterprise system are governed by six degrees of freedom viz., < Execution, Risk, Optimization, Data/Storage, Design/Structure, Enabler >

Mathematics and Physics students may also understand the depiction of the Intelligent Design functions in the form of Vector Calculus functions – Gradient, Divergence and Curl.

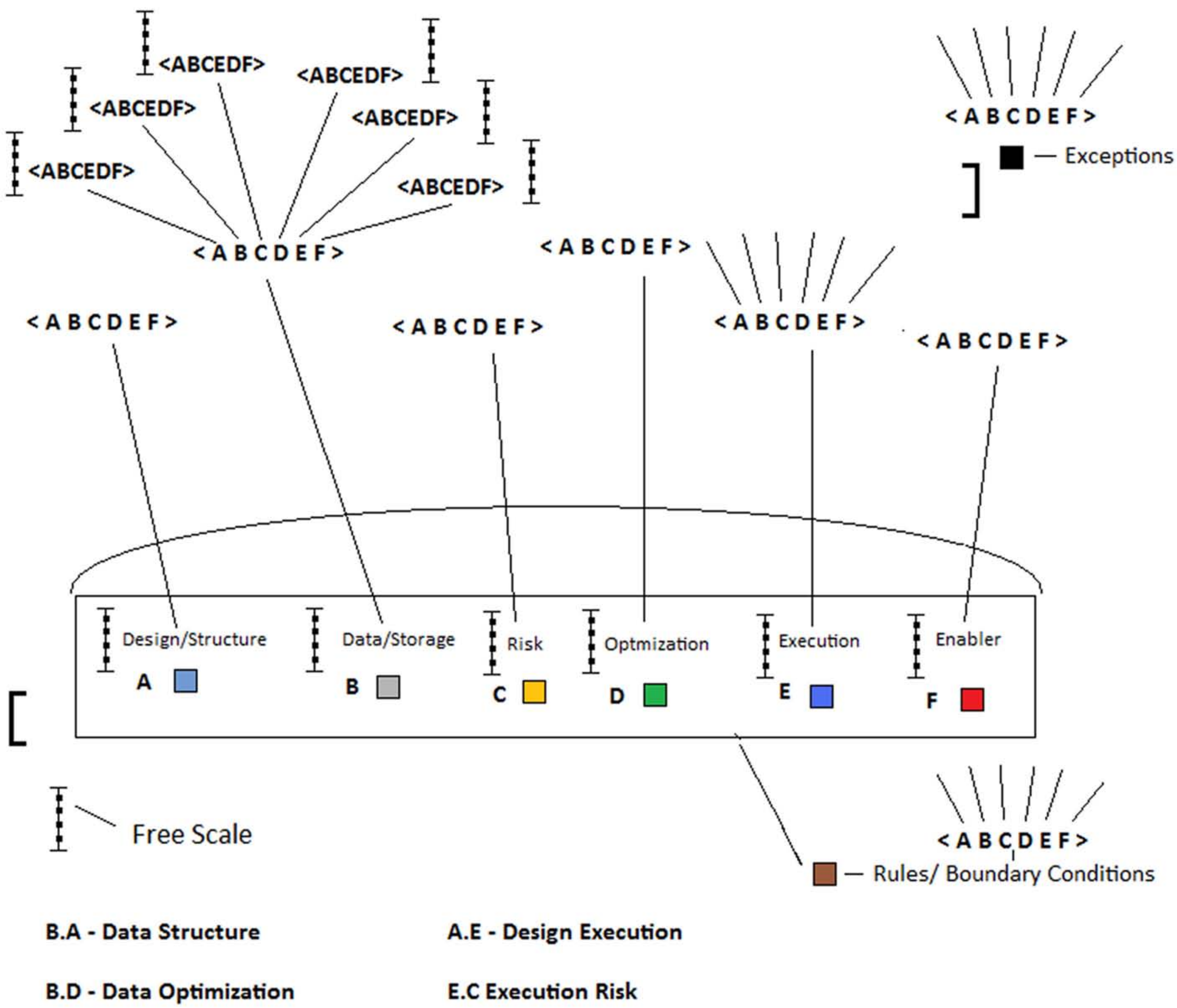


<u>Mathematical/Physics Concepts Demonstrating Intelligent Design or TEFS functions</u>

Please note that for all the examples that we have described, we have defined each of the systems, specifying only a few points for each intelligent design function e.g. Execution or risk etc. A detailed description may involve definition of multiple such execution or risk functions like e.g. execution1, execution2 and execution3 or risk1, risk2, risk3.

Till now we've been discussing the simple form of TEFS or INTELLIGENT DESIGN Function set to define systems. Let us now analyze the complex form (Recursive form) of the TEFS functions.

TEFS/Intelligent Design function set Complex form (Recursive format)



Intelligent Design Recursive Function Set

You may have noticed this recursive or complex function can be expressed as a second order, third order or nth order function set. Using the third order functions and higher order functions is not a very simple task unless used in the field of software programming. Therefore, to explain the concept, we go ahead using the second order TEFS functions/INTELLIGENT DESIGN Functions.

Since the recursive functions appear complex, our approach would be to transform the functions into a 8x8 matrix which we call as the INTELLIGENT DESIGN Matrix or the TEFS Matrix.

Intelligent Design Matrix or TEFS Matrix:

	Design/ Structure	Execution	Data/ Storage	Risk	Optimization	Enabler	Rules/ Boundary	Exceptions
Design/ Structure	Design-Design	Design-Execution	Design-Data	Design-Risk	Design-Optimization	Design-Enabler	Design-Rules	Design-Exceptions
Execution	Execution-Designed/Structured	Executable	Execution Data	Execution Risk	Execution Optimization	Execution Enabler	Execution Rules	Execution Exceptions
Data/ Storage	Data Structure	Data Execution	Meta Data	Data/ Storage Risk	Data Optimization	Data/Storage Enabler	Data rules	Data Exceptions
Risk	Risk Design/Structure	Risk Execution	Risk Data	Risk of risk	Risk Optimization	Risk Enablers	Risk rules	Risk Exceptions
Optimization	Optimization structure/design	Optimization execution	Optimization Data	Optimization Risk	Optimization of Optimization	Optimization enablers	Optimization rules	Optimization Exception
Enabler	Enabler Design	Enabler Execution	Enabler Data	Enabler Risk	Enabler optimization	Enabler of Enabler	Enabler Rules	Enabler Exceptions
Rules/ Boundary	Rules Design	Rules Execution	Rules Data	Rules Risk	Rules optimization	Rules Enabler	Rules about Rules	Rules Exception
Exceptions	Exceptions Design	Exceptions Execution	Exceptions Data	Exceptions Risk	Exceptions Optimization	Exceptions Enabler	Exceptions Boundaries	Exceptions among Exceptions

Once we execute the matrix, it may become very clear that the elemental functions of the matrix can be easily used to describe or define any concept in the enterprise space (By application of the functional oriented thought process).

Let us take a few of these elemental joins and see what they mean or represent in the real world space.

Data Structures – Subject in Computer Science, dealing with data or Structure of a product/Object or Database design in Software development or Algorithms representing a variety of data structures.

Data of Data – That is data about data, also called metadata. In computer science, it could mean header information about a data packet. In general context, it could mean the synopsis or a précis about some detailed information.

Execution Risk – Execution risk implies quality of execution or an executable. If execution risk is 0 (Zero), then it can be inferred that the deliverable of the product is of 100% quality.

Design Execution - drawing of graphics on a computer screen or any other media. Ina manufacturing context, it could mean development of a prototype.

Design Data – design data is the blue print or design templates which can be replicated during object creation or object manufacturing process.

Execution Optimization – represents the most efficient process or the sequence of executing a task. E.g. in a software development context, it could mean tuning of an application for performance or in an automobile context, tuning the engine for maximum efficiency.

Execution - **Execution** – It represents an executable or a deliverable. In computer science context, it represents an executable binary. In automotive space, it represents the assembly of a car or the manufacture of an auto component/machine part.

Structured Execution –It represents a systematic execution process. In computer science context, it could mean structured programming languages like PASCAL, C++ or Ada. In manufacturing process, it could represent Cellular Manufacturing Techniques.

Rule Design – It represents the design of systems rules for effective functioning of the system. In the context of a game, it could mean the design of rules of the game. In the context of an enterprise, it could mean the rules and policies of the enterprise.

Rule Execution – It is the process of compliance of rules in any system, machine or enterprise. In the software terminology, it represents functioning of a rule engine.

Execution exceptions – It represents an exceptional condition during the execution process. E.g. Machine failure. In software context, it could mean occurrence of an illogical statement like 'Divide by zero'.

Exception - exception - It represents an exception among exceptions. Out of hundred executions instances, if 5 values of exceptions are in the range 100-200 and one value of exception is in the range 1000-2000, such an exception would be termed as an exception among exceptions.

We have taken up a few cases of these matrix elemental joins and given a few examples. The rest of the elements could be worked out by you as an exercise.

You may observe that in some systems certain of these elemental functions appear as strong functions and some appear as weak functions.

Strong functions are those which have a strong significance in the particular system or process.

Weak functions are functions with less significance. For example, if we are talking in the context of building a high performance machine, then the execution optimization function may become less significant or may be classified as a weak function because we may focus on a higher performance of the machine rather than a more efficient performance.

As stated above, the third order and higher order INTELLIGENT DESIGN functions are not easily applicable. However, it appears feasible in software development process. An example is as follows:

```
System.Design.Draw Circle( )
        {
                System.Execute.Fetch_Radius ( );
                {
                        System. Data.Database_Call ( )
                        {
                                System. Data.Use_Connnection_Pooling ( ) { Conn.Open … }

                        }
                }
        }
```

We have now covered all the magical eight steps to system definition. You may now understand why the chapter name has been called so.

We now discuss a few ‘Not so Brilliant’ functions which support or act in conjunction with the primary TEFS or INTELLIGENT DESIGN Functions.

1. **State** – State functions help define that exact state of a system at a given time.

2. **Time** – Time is a sequence of timestamps at which any function may execute. E.g. time T1 - execution, Time T2 - optimization, time T3 – execution, time T4 – Data etc.

3. **Iterations** – Iterations represent number of cycles of design or execution functions.

4. **Threads/Parallels** - These represent simultaneous executions of any task or activity.

5. **Measurement** – It is a function to measure or quantify an execution or any other function. E.g. measurement of execution percentage or measurement of execution risks showing quality of deliverables.

6. **Medium** – Medium functions are used to convey the medium of execution or material medium. E.g. Medium of storage could be Hard Disks or tapes, medium of data could be print medium or digital medium on the web.

7. **Views** – Views represent a perspective of a process in execution from a variety of angles.eg Perspective of a junior engineer, Perspective of CEO etc.

8. **EDIC (Explicitly Distinct Implicitly Coherent)** – This function is used as a technique in intelligent design keeping the type of functions explicitly distinct like e.g. explicitly distinct execution functions ,optimization functions etc., yet keeping all the functions cohesively coherent.

We have now completed our discourse involving the TEFS functions or INTELLIGENT DESIGN functions.

Let's analyze a few Information Technology companies and the space they represent in the TEFS matrix or the INTELLIGENT DESIGN matrix:

Google is a formidable search engine company operating in the data space. It may be worthwhile to note that while there are many search engine companies around, Google has a lead in the search space because of its strength in the DATA-DATA (METADATA) / Indexes space. Over a period of time, Google has managed to translate large sets of search queries into huge indexes. While the search process remains nearly the same for all search engines, it is the metadata functions that give Google its strength.

Another company like Microsoft occupies the software enabler space. This space is represented by its operating system software which allows or is a pre-requisite for a wide variety of applications to run on it. E.g. Industry software, web applications, games etc.

Companies like Autodesk etc., occupy the structure design space.

Companies like IBM, Oracle and Microsoft occupy the Data Execution space represented by their products DB2, OracleDB and MSSQL.

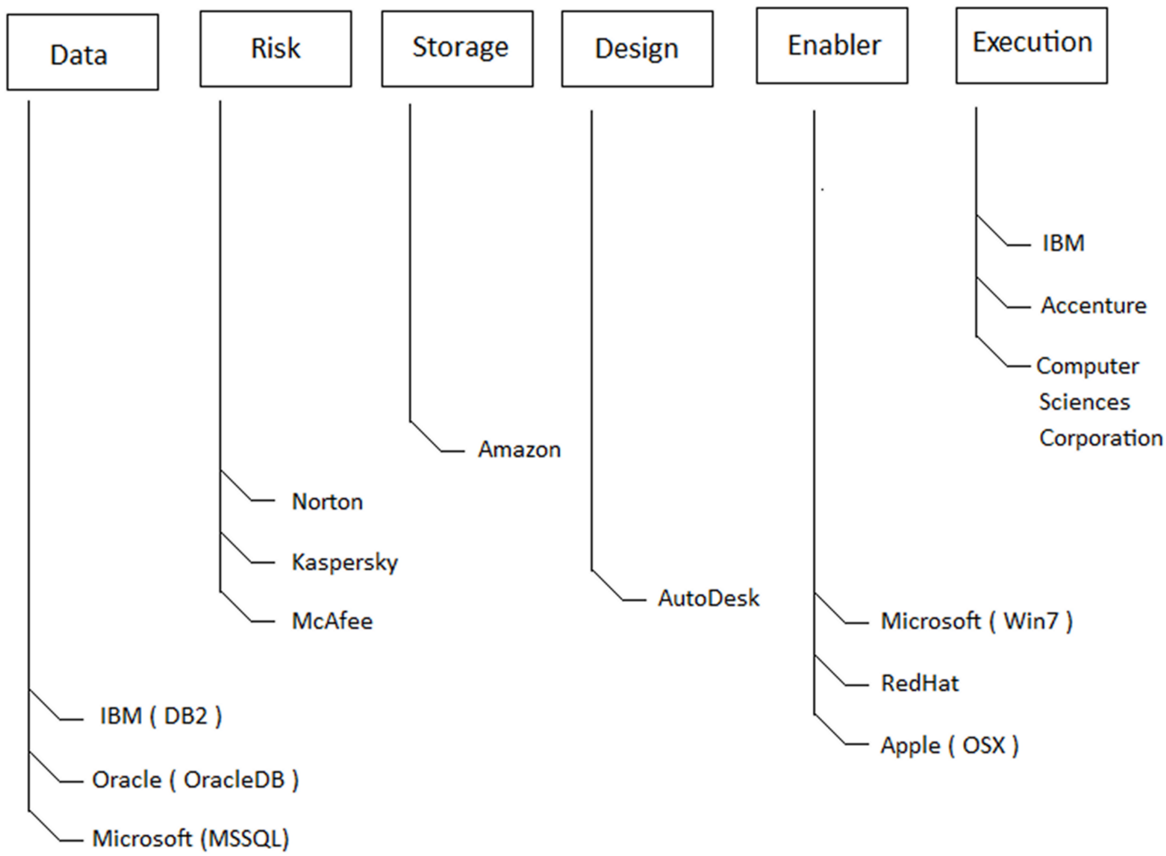
Companies like Norton, Kaspersky, and McAfee occupy the Risk Execution space represented by their products - Virus scanners.

Amazon is an e-commerce company occupying the Storage Execution space represented by its fulfillment services, although it is classified as a distribution or trade enterprise.

T&C Vectors is currently working on occupying two spaces, the first space occupying the entire enterprise systems space represented by its product “Enterprise Focus” and occupying the second space of structured search through its search engine “Inter-Xect”.

Until now, we had been discussing the application of TEFS functions in various kinds of systems. The succeeding chapters would deal with how the TEFS functions would deal with the specific constituent entities of the system space viz., People, Processes and Products.

Information Tecchnology Companies Worldmap



Chapter 5

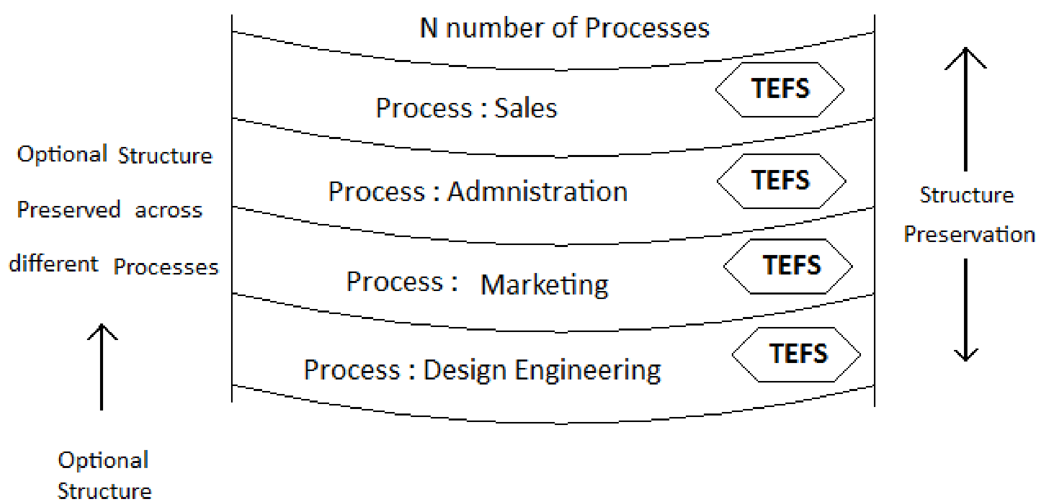
Breaking the IQ barriers by cognitive fluidity

This chapter deals with two constituent entities of the enterprise system:

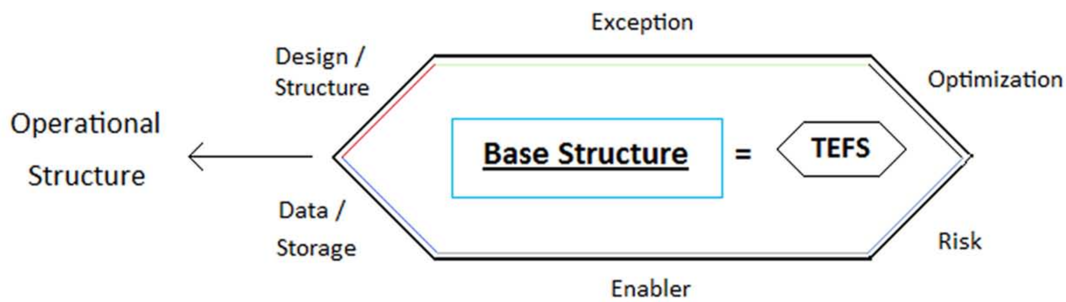
1. Processes
2. People.

Let's first deal with the process aspect of the enterprise space. You may remember processes are classified in the enterprise space as specialized and generic processes. Specialized processes are domain specific processes like Software development or automotive manufacturing process. Generic processes are processes which are typically common to all types of industry e.g. HR process, Marketing and sales.

Mathematics teaches us few concepts in abstract algebra. Abstract algebra studies sets with operations that generate interesting structure or properties on the sets. The most interesting functions are those that preserve the operations. These functions are called Homomorphism's. Homomorphism is a structure preserving Map between two algebraic structures. Applying these mathematical principles in the enterprise space leads us to the following enterprise process structures:



<u>Preservation of Intelligent Design functions across Processes</u>



The TEFS functions or The Intelligent Design Function set

We now are aware that the TEFS or INTELLIGENT DESIGN functions can be used to describe the specialized process as well as the generic process. To understand this better, we take up a process in marketing and see how the TEFS functions defines the process.

Marketing Processes:

Design/Structure – In the marketing process, design functions involve design of advertising campaigns.

Data/Storage – Data functions represent the process of collecting data from market research as well as product data.

Execution – It is the execution of the marketing campaigning.

Optimization – Allocation of capital to different mediums of advertising.

Risk – Risk is assessment of all the negative reactions that might originate as a part of the marketing campaigning.

Enabler – It is all the information about the products and the intended target audience.

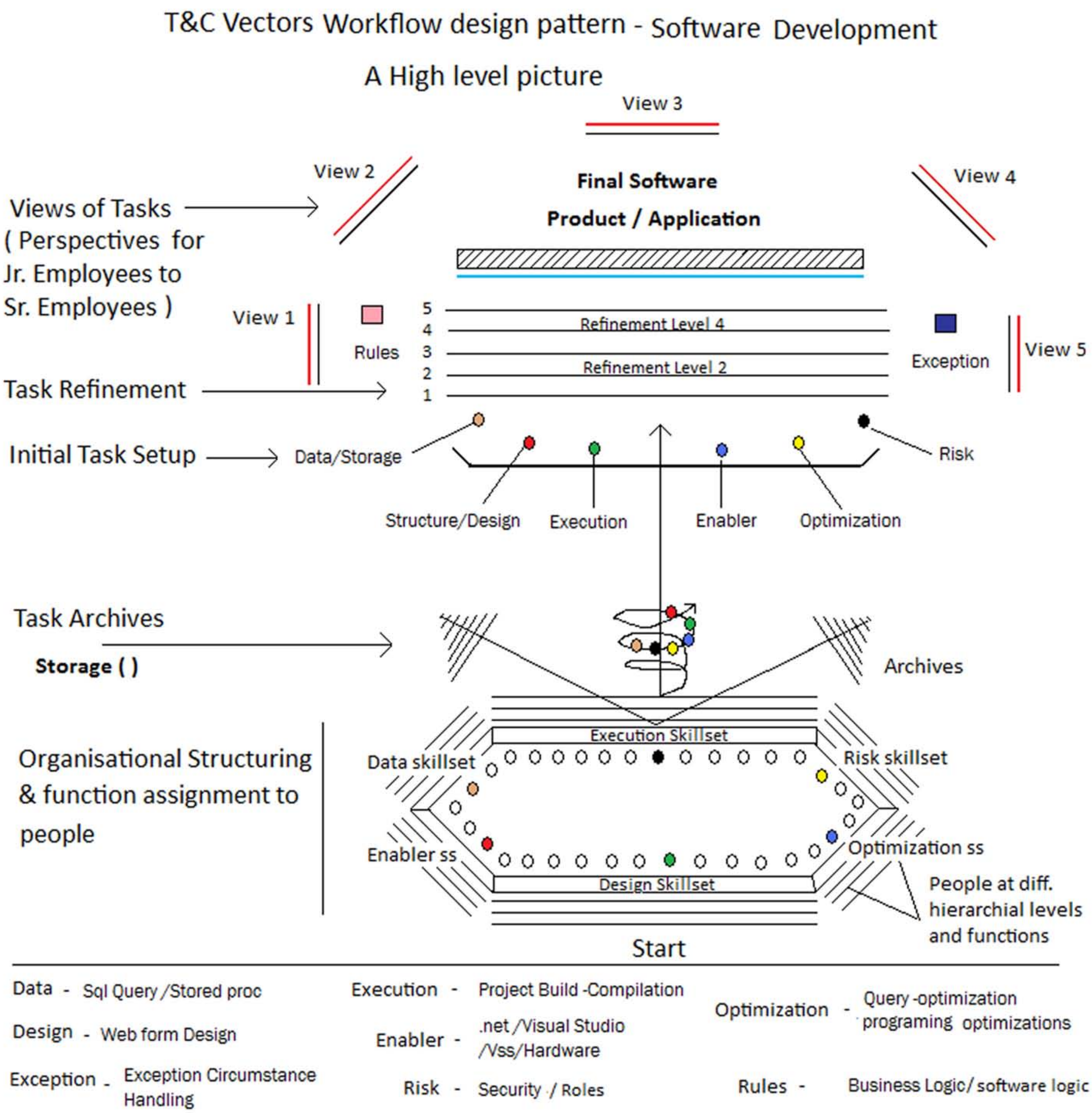
Rules/Boundary Conditions – Compliance with all advertising rules and laws, regulations existing in the enterprise space.

Exceptions – An unusual condition requiring out of the ordinary actions.

Now let us discuss the discipline of developing software.

Software Development Process – Development of a web/windows application:

Data/Storage – It involves database design, SQL queries and store procedures.
Design/Structure – Web form/ Windows form Design.
Execution – It is the project compilation and build.
Enabler – It includes hardware, .Net framework, integrated Development Environment (IDE) like Visual Studio.
Optimization – Programming optimizations involves .Net programming optimizations or Database query optimizations.
Risk – Involves all the security essentials related to the software.
Rules – Involves the codification of all business logic and process logic.
Exceptions – Exception handling within the application.

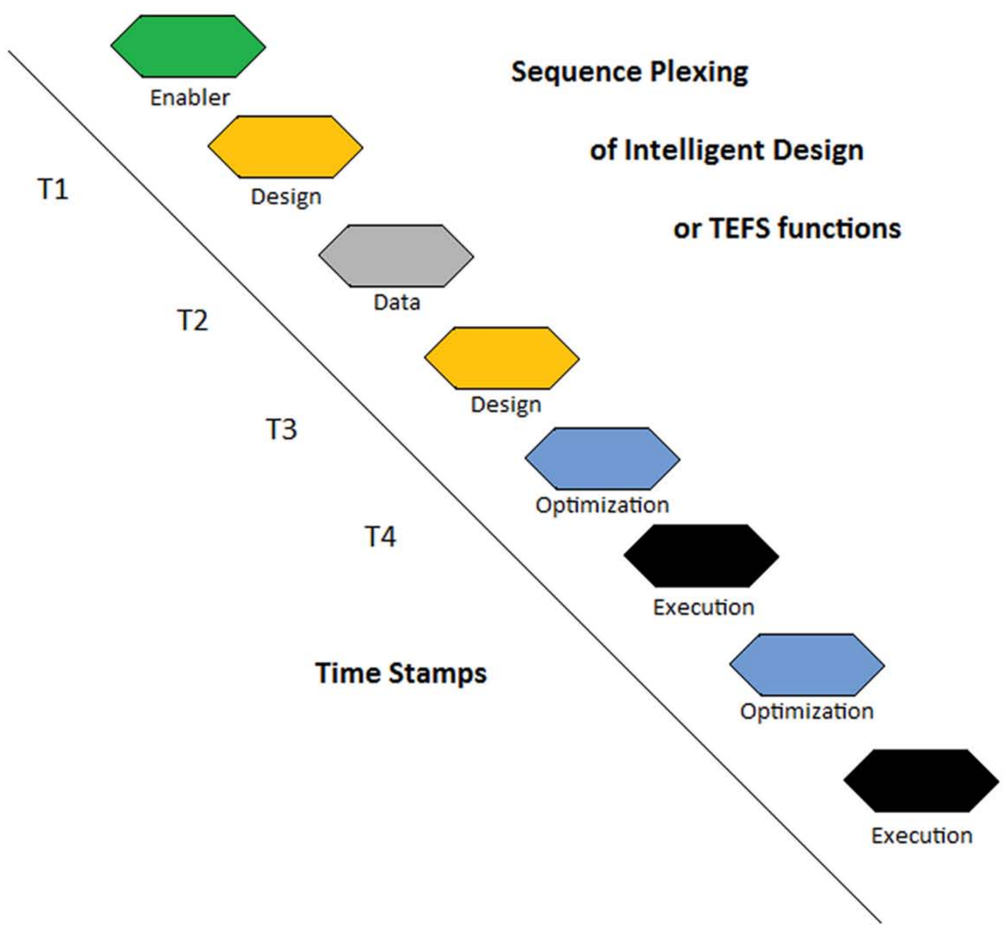


Until now, the standard software development methodologies included SDLC (Waterfall model, Spiral Model etc.) and techniques of Agile development. TEFS foundation suggests a next step in the software development evolutionary life cycle which would be known as Hyper Programming or Programming by Intelligent Design.

Initially, in SDLC, the focus lay on the process. In agile techniques, it shifted to people and teams. Hyper Programming or INTELLIGENT DESIGN methodology suggests a remarkable shift in both the process space and the people's space. It involves sequence plexing of TEFS functions or INTELLIGENT DESIGN functions.

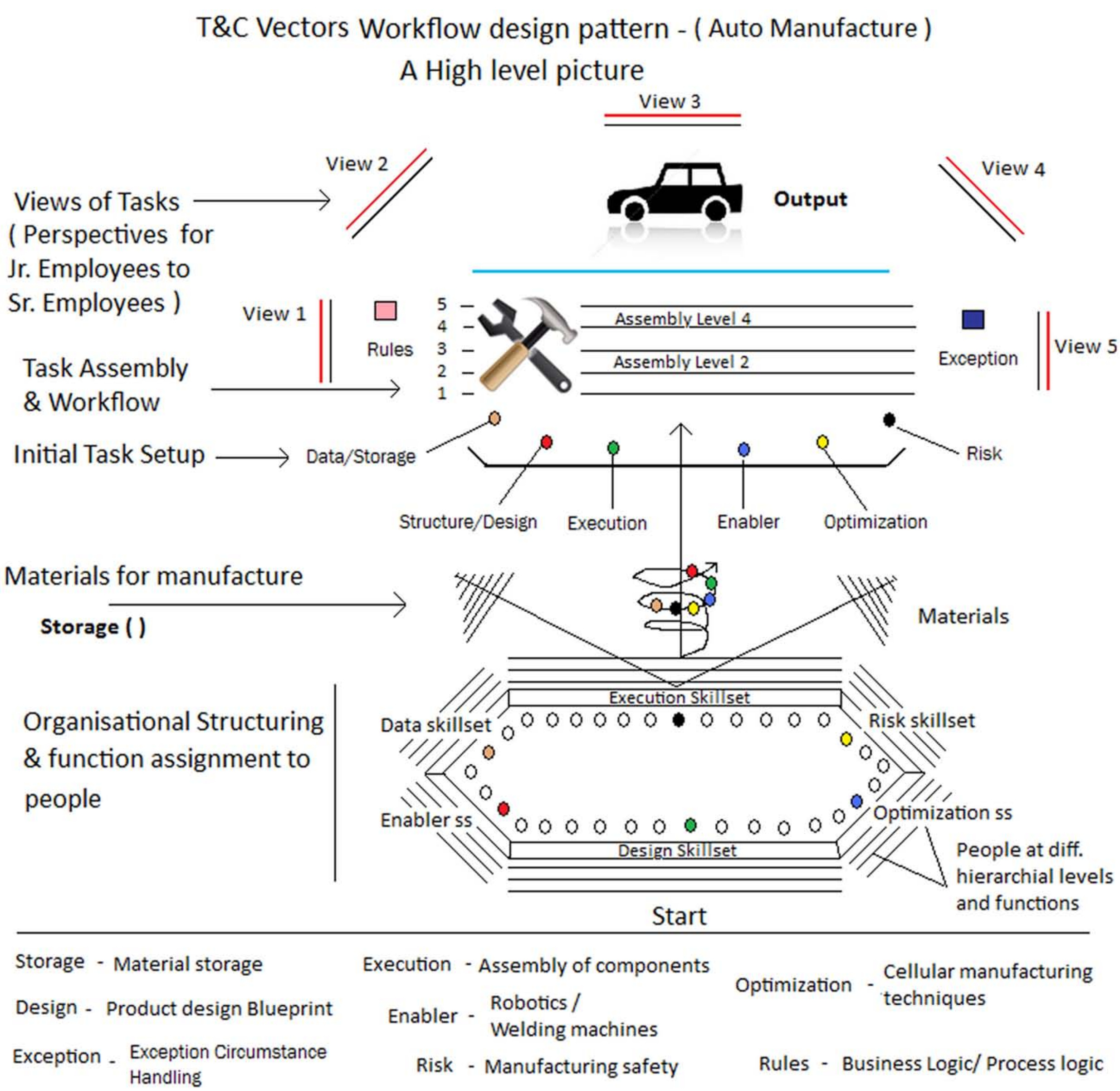
It also by principle necessitates entire team cohesion towards a task of common concern. The teams are structured so that the team members are split up into factored teams focusing on one particular aspect of the software development. For example, a team may be solely responsible for all tasks related to optimization of the software or risks aspects of software. The software development technique is represented by the sequence-plexing diagram as shown below:

Hyper Programming



Auto manufacturing Process:

Design/Structure – Involves Product design blue prints interior plus exterior.
Data/Storage – Storage involves material storage and machine component storage.
Execution – Involves assembly of all components to produce the final product.
Enabler – Includes Robotics and welding machines for the assembly process.
Optimization – Includes cellular manufacturing techniques where similar types of products are produced together to reduce material flow through the enterprise space leading to an optimized production.
Risk – Involves all activities related to manufacturing safety.
Rules – Represents Logical rules and sequence to building a product.
Exceptions – Exceptions are incidents or events that take place over the entire manufacturing cycle.



Enterprise architecture concepts:

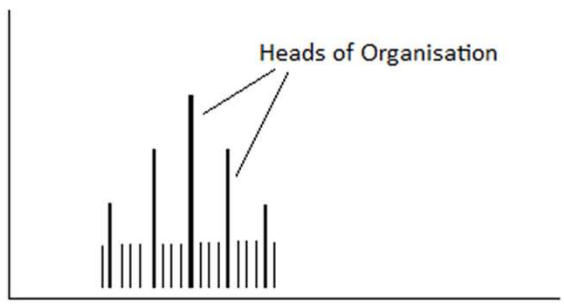
Enterprise architecture is a rigorous description of the structure of an enterprise, its decomposition into subsystems and the relationships between the subsystems. The description includes business processes, roles and organizational structures.

By the introduction of TEFS functions or the INTELLIGENT DESIGN functions we have simplified the complex terminology of enterprise architecture wherein every system consists of processes and every process, be its specialized or non-specialized, follows a design pattern in line with the TEFS functions. After observing processes, the other points necessary to complete an entire enterprise architecture process are organizational structures and roles.

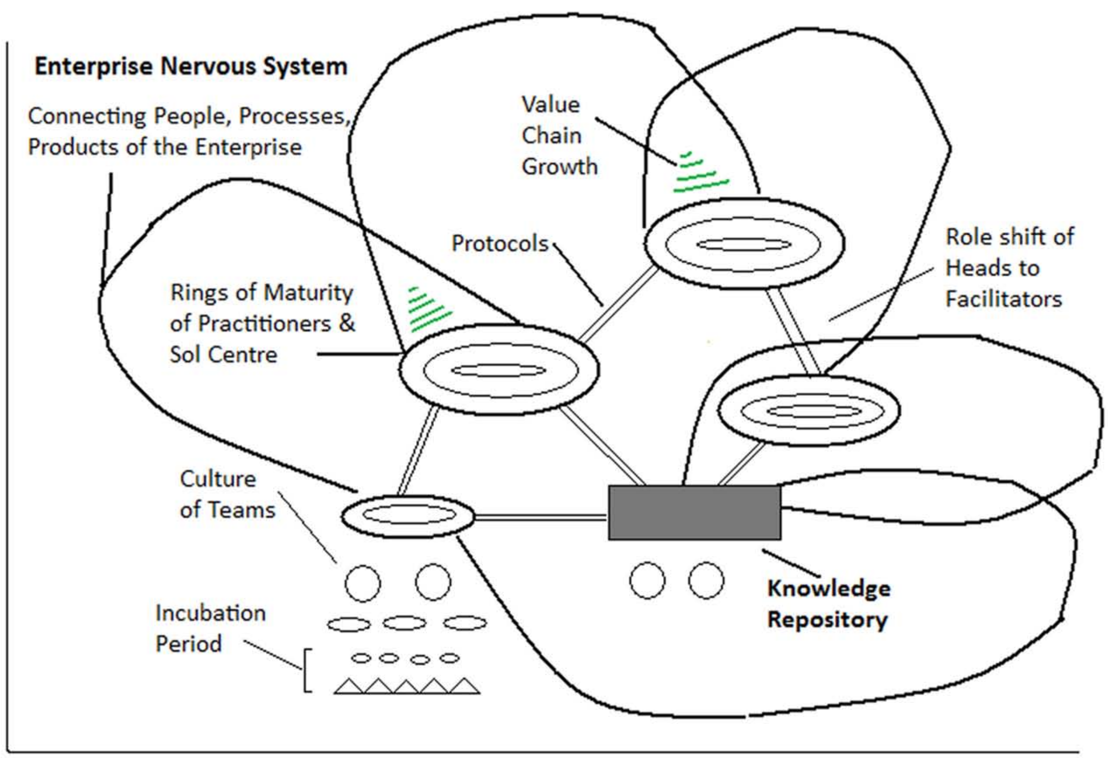
Excerpts from the work "The management of Innovation" by Burns and Stalker have provided the following characteristics of static and dynamic systems:

	Static Systems	Dynamic Systems
The Environment	Static	Dynamic / Ever changing.
Distribution of tasks	Specialized differentiation of functional tasks.	Contributive in nature to common task of concern.
Task Scope	Precise definition of rights and obligations and technical methods attached to a role.	Shedding of responsibility as a limited field of Rights. Problems may be posted Upwards, sideways, Downwards.
How is Task Conformance Ensured	Translation of rights and obligations and methods into responsibilities of a function.	Spread of commitment concerned beyond any technical definition.
Location of Knowledge	Reinforcement of the Hierarchic structure by the location of knowledge of actualities at top of Hierarchy.	Omniscience is no longer imputed to the head of concern.
Communication between Members	Vertical communication	Lateral communication
Values	Loyalty & Obedience	Commitment to the concern's tasks and methods of material progress and expansion values more.

You may observe from the diagram shown below that the Intelligent Design architecture inherently handles all the characteristic methodologies of dynamic systems. The most prominent of them being, the enablement of cohesive teams working towards a common goal or task of concern and an Enterprise nervous system connecting the people, processes and the products.



Static Enterprise System



Dynamic Enterprise System

Let’s move ahead from the process end of the spectrum to the people end of the spectrum. It is now imperative that we learn about two terms which may render our discussions much simpler:

1. **Intelligence Quotient** – is a metric to measure a person’s intellectual capabilities in comprehending knowledge or structured information to be able to effectively utilize it.

2. **Cognitive Fluidity** – is a term to describe the ability of a person to learn and comprehend knowledge or structured information belonging to diverse domains and processes.

The human cognitive process occurs at two levels:

1. Through the educational systems at school, college and university.

2. In the enterprise space through research and work experience.

It has been observed that the gaps that exist between the cognitive process at school, college and between those in the industry are fairly large. It is near impossible for enterprise knowledge and information to flow down to educational systems in a systematic way primarily because of lack of suitable channels and secondly because much of the intellectual learning that happened at the industries research and development environment are classified as proprietary intellectual property.

We at T&C Vectors are attempting to bridge this gap in a small way by imparting a functional thought process that may find relevance at the industry level. This portion of our exercise involves making people smarter. We have already introduced the concept of functional thinking paradigm in the previous chapter. Our attempt to make people smarter involves two steps:

1. Better learning and cognitive process

2. Better communication ability.

Better learning and cognitive process:

Better cognition or learning involves learning to specialize in a particular domain and additionally building enough cognitive fluidity to master associated domains. For Example, a software engineer may learn to specialize in software programming/ development, as well as learn other associated disciplines like enterprise architecture and enterprise engineering.

T&C Vectors proposes the TEFS based functional learning techniques in the cognition process while starting from ground up at school. The reason being this functional and systems oriented approach facilitates an absolutely clear learning technique in a variety of subjects ranging from mathematics, science to engineering.

Better communication ability:

We are very aware that people are native speakers of a multitude of languages prevalent in diverse geographical areas. We might agree that English is a globally accepted language that's used in most forms of technical communication today. But could there be a more technical oriented language that connects people from a variety of fields and specializations?

Our approach to introducing a new form of communication takes root from the fundamental behavior of how diverse and disparate machine systems communicate through language neutral formats of data e.g. XML.

People in the computer science industry may well be aware that today's programming language compilers translate a variety of programming languages like C#, Visual Basic, C++, Delphi to intermediate forms of language called an intermediate language.

The TEFS or the INTELLIGENT DESIGN functions again prove worthwhile in the construction of an intermediate technical language that people may utilize to communicate and convey a very sharp context.

The Intermediate Language Matrix:

	Design/ Structure	Execution	Data/ Storage	Risk	Optimization	Enabler	Rules/ Boundary	Exceptions
Design/ Structure	Design-Design	Design-Execution	Design-Data	Design-Risk	Design-Optimization	Design-Enabler	Design-Rules	Design-Exceptions
Execution	Execution-Designed/Structured	Executable	Execution Data	Execution Risk	Execution Optimization	Execution Enabler	Execution Rules	Execution Exceptions
Data/ Storage	Data Structure	Data Execution	Meta Data	Data/ Storage Risk	Data Optimization	Data/Storage Enabler	Data rules	Data Exceptions
Risk	Risk Design/Structure	Risk Execution	Risk Data	Risk of risk	Risk Optimization	Risk Enablers	Risk rules	Risk Exceptions
Optimization	Optimization structure/design	Optimization execution	Optimization Data	Optimization Risk	Optimization of Optimization	Optimization enablers	Optimization rules	Optimization Exception
Enabler	Enabler Design	Enabler Execution	Enabler Data	Enabler Risk	Enabler optimization	Enabler of Enabler	Enabler Rules	Enabler Exceptions
Rules/ Boundary	Rules Design	Rules Execution	Rules Data	Rules Risk	Rules optimization	Rules Enabler	Rules about Rules	Rules Exception
Exceptions	Exceptions Design	Exceptions Execution	Exceptions Data	Exceptions Risk	Exceptions Optimization	Exceptions Enabler	Exceptions Boundaries	Exceptions among Exceptions

Let us, for example, set up a few dialogues that showcase an intermediate technical language:

XYZ – ABC! What is the **EXECUTION** status of project X?

ABC – XYZ, project X **EXECUTION** is 80% complete. However, the cost percent is 95% due to advanced payments.

XYZ – I heard there were two critical **EXCEPTION**s last month. Had they been handled?

ABC – Yes! The first critical **EXCEPTION** was a server fault due to a faulty hard disk. The second critical **EXCEPTION** was due to failure of the internet gateway. Both the

EXCEPTIONS were handled in urgency adhering to all **RULES** of operations and also in adherence with all the **RISK** procedures.

XYZ - Other than the **EXCEPTIONS**, have you observed any **EXECUTION RISKS** in the application deployment and **EXECUTION** process?

ABC – We've collected all **EXECUTION DATA** and have not encountered even a single instance of **EXECUTION RISK**.

XYZ – Did you know, we've called in some IT Engineers to **DESIGN** the next module of the manufacturing process?

ABC – I heard that. We have the **DATA** ready. I guess they would be insisting on starting with the **DATA DESIGN.**

XYZ - Did you hear the latest story about the car crashes on Manhattan road due to faulty chassis?

ABC – Yes, I did hear. They must be taking in the cars for a **STRUCTURAL RISK** check. First report suggests a **DESIGN RISK** in the front suspension.

This ends our discussion of an intermediate technical language.

We now have insights into domain neutral and process neutral techniques theoretically explained with the help of TEFS functions or INTELLIGENT DESIGN functions. Additionally, we've also developed an intermediate technical language with the help of the TEFS function sets.

Our journey in learning has gone through the following discourse. In the chapter "Thinking Objects and Systems", we learnt about how to define a machine system and an enterprise system. At the end of the chapter we've set a goal to build a smarter world by making people, process and products ends of the spectrum smarter.

In the chapter "Eight Steps to brilliance", we learnt about the TEFS functions or the INTELLIGENT DESIGN function set and learnt its applicability in the systems space.

In this chapter we've fulfilled a part of our goal to build a smarter world by learning to apply the TEFS functional methodology to the process and people ends of the spectrum. The next succeeding chapter would deal with making the product end of the spectrum smarter.

Chapter 6

Engineering and Design of Products

Having covered the people and the process end of the spectrum, we move to the product end of the spectrum where we use advance techniques to construct or develop smarter products.

Products are basically tangible or intangible systems. We have already learnt about the concept of object to systems polymorphicity. Products that are not overly complex can be simply called as objects. Every product that has been invented or built has a singular design goal. A complex product may have multiple functionalities to present a slightly complex design goal.

A car is designed to transport people on the ground.
A commercial Aircraft is designed to transport people by air.

While both cars and aircrafts transport people, their medium of execution differ i.e. one uses road as a medium while the other uses air.

Let's take another example.

A phone is designed to help people communicate with voice.
A smartphone is designed to help people communicate with voice as well as data.

We again notice that while both the products are used for the purpose communication, they differ in their medium of execution.

Therefore it becomes very necessary that we clearly define the design goal of the product that we plan to develop or create.

For ease of understanding, products can be classified into three types:

1. Physical/Material products

2. Software products and

3. Hybrid products.

Further, each of these products can be classified again as

1. Static products (objects or systems)
2. Dynamic products (objects or systems)

These two classifications result in the formation of a product matrix:

	Static Products	Dynamic Products
Physical / Material Products	Physical Static products	Physical Dynamic products
Software Products	Software Static products	Software Dynamic products
Hybrid Products	Hybrid Static products	Hybrid Dynamic products

Physical/Material products are products that are tangible, have a physical presence and their workings or creations are governed by scientific laws of physics, chemistry and engineering.

Software products are intangible products which run in a computer or server system and provide information/systems solutions in a specific domain or in a specialized process.

Hybrid products are products which have hardware as well as a software element to them. Examples of hybrid systems are smartphones and computers.

All products naturally go through the basic evolutionary cycles of design and execution. When constructing products in the physical world, the rules of science and engineering are so rigid that it inherently never tolerates any fundamental design flaws which go against the scientific laws. If flaws exist, they very “observably” show up as exceptions and failures.

Our approach to building physical world objects uses the TEFS functions or the INTELLIGENT DESIGN functions to cover all functional aspects of development or construction.

Let’s for example, take the construction of a physical world building. Let us now use the TEFS functions or the INTELLIGENT DESIGN function set in the creation of the building.

Structure/design - Structural functions take care of all the materials and procedures required to create a stable structural foundation. Design functions cover all the user friendly aspects as well as the aesthetic and sanitation aspects.

Risk functions – Risk functions cover all the risk factors and security functions related to the building.

Data/Storage – The data functions cover all the information and data channels to the building. The storage functions cover all aspects of external storage spaces like warehouses for material storage.

Enabler functions - The enabler functions cover the aspects of suitable soil tested land availability worthy of construction activity as well as Power/Energy supply.

Optimization functions – Optimization functions cover all aspects of creating as much open space or work space within the limited boundaries of the building.

Execution – Execution functions cover tested material and construction procedures to erect a building.

Exceptions – Exceptions functions cover all the untoward incidents or events that may occur during building construction or execution.

Rules/ Boundary conditions - Rules/ Boundary conditions cover all the aspects of compliance to building construction norms.

You may notice that many formal design and construction techniques inherently adhere to some, if not all functions of Intelligent Design. While this is so noticeable in physical world products, the software world products do not necessarily follow such a pattern.

If we approach the building of software products through an Intelligent Design approach, then this software would naturally adhere to all principles of systems theory and such software may be called a software system. Software systems which do not follow or cover all the aspects of intelligent design may be called merely software.

Let us now move our discussion to hybrid systems.

Hybrid systems as we mention have software aspects as well as hardware aspects. But for a stable system to function, the software and the hardware must co-exist and co execute compatibly.

Ill designed software can easily ruin the best hardware systems. We may agree that while designing hardware systems or hardware aspects, some, if not all, rules of Intelligent Design are naturally adhered to while, this is not so in case of software. The reason being the design and structural flaws are more observable in physical or hardware systems than in software. It is for this reason that software bug is a very acceptable term while discussing software.

Another point to be noted is that hardware product design is a more cohesive activity than a software development activity because the surface area of flaws that may occur in hardware systems is comparatively less than while building software systems.

While building hybrid systems our approach should follow the process of intelligent design in both, the aspects of hardware and software.

Design Sense:

Design is a very frequently used term which very naturally conveys the meaning of creating a template or blueprint for a product, based on aesthetic scales. However, Design in the form of Intelligent Design is actually a rarity in this world because the intelligent design approach dwells into establishing the science of art or aesthetics.

Talking about the software world, we may agree that the software world has a very strong foundation with the usage of elements of abstraction in the software development cycles.

Abstractions very frequently signify an artistic aspect. Therefore, many a time, the activity of software development has been described as – “The Art of Programming”.

The T&C Vectors approach to building software suggests a far more scientific design behavior of software, a deviation from the artistic or abstract behavior. The reason being, art or abstraction is “not always meant to be understood”.

Many a time, we might have heard debates “comparing Windows operating systems to Apple Macs” or “Software as a science or an art”. Such debates necessarily use the art scale of measurement to differentiate.

If, however, software were developed as software systems, like hardware systems, the room for debates would narrow down to scientific scales of measurements like Execution Performance, Execution Risk or Risk mitigation ability etc.

Interface based Technique and Visual Paradigm:

We will now move to apply the interface based technique to product development.

Every object or system is necessarily perceived through its interface. Therefore, it may not be hard to convince that an interface is really the object. It is with this foundation statement that the software world is acquainted with the term or concept of "Interfaces". For clarity in discussion, we address interfaces as object interfaces. Though the use of the word interface conveys user interface, we must be careful with the terminology. The user interfaces may be treated as explicit object interfaces for the purpose of showing graphics or design on a visual screen.

For those of you who are acquainted with the software terminologies may know the definition of an interface object which is nothing more than a collection of abstract functions, which gain definition and logic during class definition.

For those of you who are not acquainted with the software world, you may understand that an interface object in the software terminology is actually a file which contains a collection of functions that the object or the system is capable of doing. It is merely the depiction of an object or a complex system in the form of a set of functions.

Interfaces can be classified into two types:

1. Macro interfaces
2. Micro interfaces.

Macro interfaces are interfaces which contain macro functions or functions which are abstract in nature.

Micro interfaces are interfaces which contain functions at a micro level or a detailed level.

You may observe that in the present stage of technical progress, design systems like Computer Aided Design (CAD) or Computer Aided Machining (CAM) use the technique of translating software blueprints into physical world products/objects. It is because of this principle, that we are essentially discussing this interface based approach to designing far more precise, sophisticated and yet simpler products.

Let us now move ahead in describing a few real world interfaces.

Example of a Macro interface: A car engine.
The macro functions that the car engine interface contains are as follows:

Start()
Stop()
Accelerate()
Decelerate()

Now a micro interface of a car engine is as follows:

cranking()
Fuel_injection()
Air_intake()
Compression_cycle()
Sparkler_ignition()
Combustion()
Piston_cycle()
Exhaust()
Accelerated_engine_cycle()
Decelarated_engine_cycle()

Now let's discuss an Aircraft interface.

Aircraft_taxing( )
Aircraft_Takeoff( )
Aircraft_fly_forward( )
Aircraft_dive( )
Aircraft_roll( )
Aircraft_yaw( )
Aircraft_carry_goods( )
Aircraft_seat_passengers( )
Aircraft_maintain_ambient_temperature( )

We have now learnt the concept of defining an interface for any product or system. Our goal now is to make these interfaces smarter using the Intelligent Design techniques.

The intelligent design approach uses the "what you see is what you get" principle to design fully factorized interfaces. Please note that we used the term fully factorized interface to convey every aspect of the systems functionality.

In the interfaces that we have defined above, the interface functions are ad hoc in nature and there is no way for us to decipher whether the function was an execution function, a data storage function or a risk function.

As described in the product matrix, the systems may be classified as static or dynamic. This suggests that the interfaces that build up to a system may also be static or dynamic in nature. Let us take an example of a dynamic system using the example of an aircraft.

The wings of an Aircraft in today's technology can be classified as dynamic systems. This is so because the aircraft wings are designed to dynamically increase or decrease surface area of the wings during takeoff or landing.

Similarly, automobiles or cars may have the functionality of retracting and folding away the car roof as in a convertible.

This leads us to the concept of fully factorized and free scaling interfaces. In simple words, free scales are the dynamic functionality added to the functioning of any system or product. Free scales are dynamic scaling ability of the intelligent design functions defined in any interface.

E.g. A server/Computer system may have free scaling / expandable memory slots to accommodate more memory for the hardware as and when the need for scalability arises

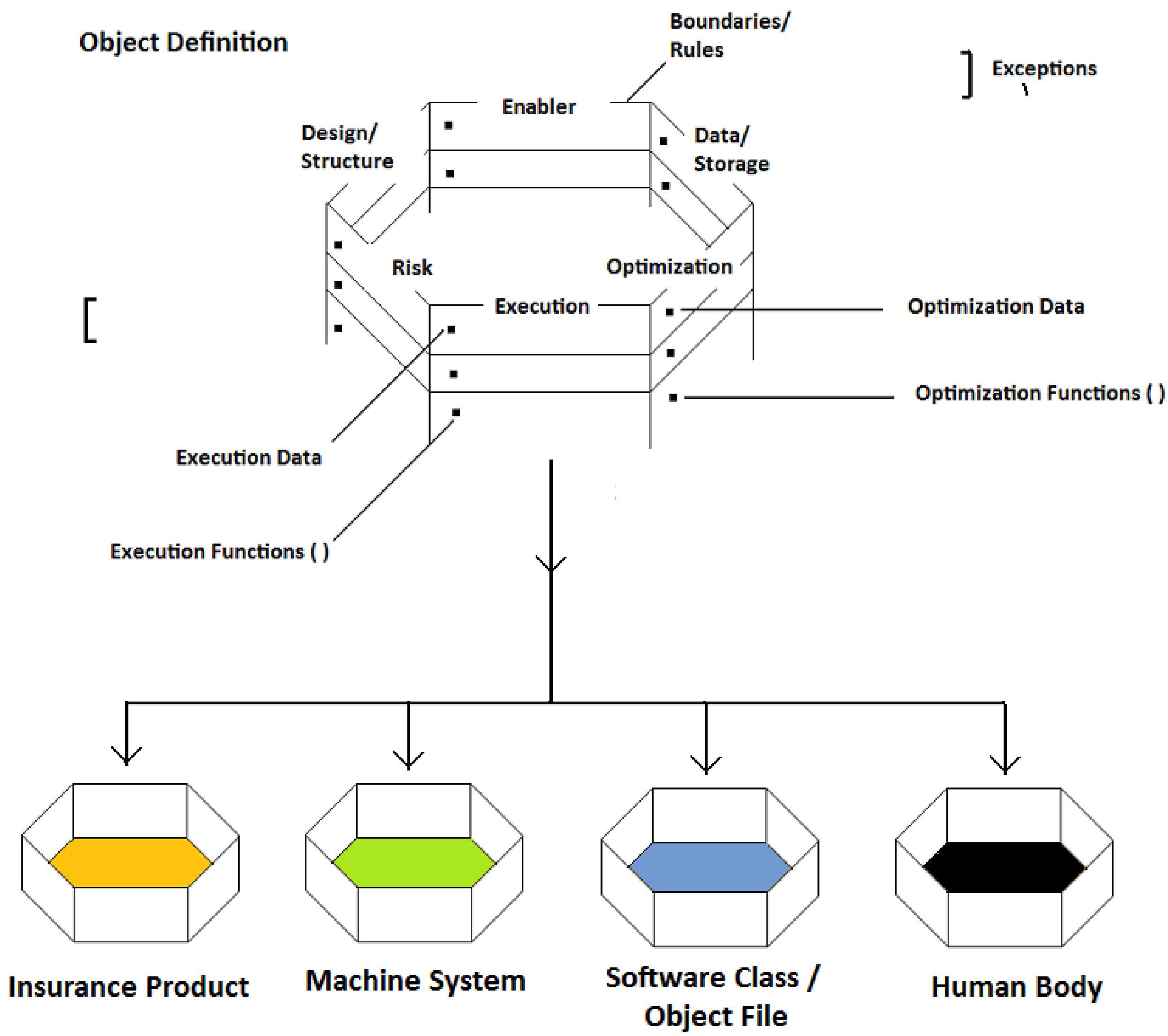
Similarly Boot space of a car may be increased dynamically by folding away seats.

Polymorphicity of objects/interfaces:

Those acquainted to the computer science world may be aware that in the object oriented world, all objects inherit from the parent or base object. This is true even in the case of the interface based technique that we are presently discussing.

The Intelligent Design fully factorized free scaling interfaces form a formal interface or the base interface for every or any kind of product or system.

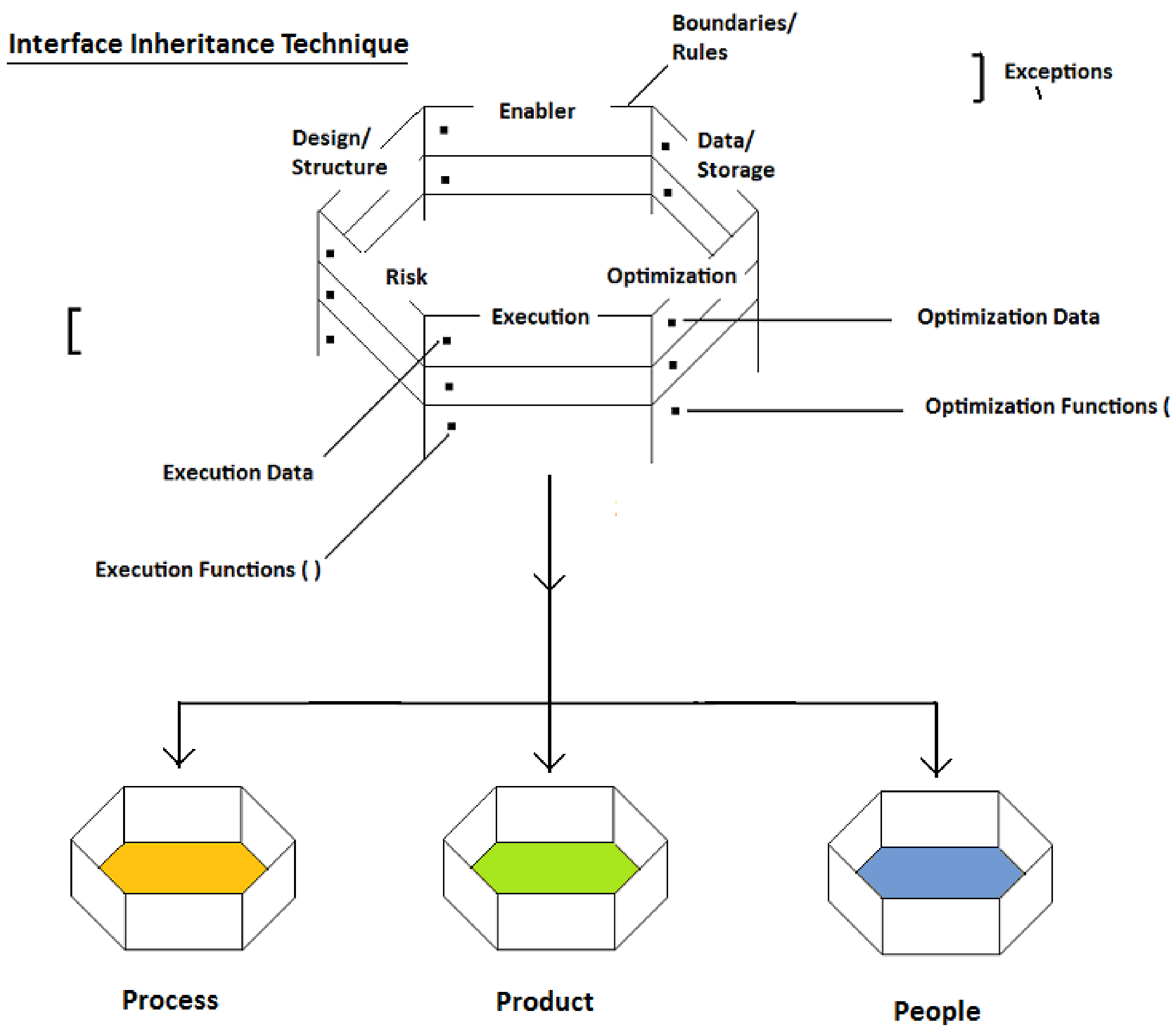
A pictorial representation is shown in the next page:



Polymorphicity of objects/interfaces

:

We have already discussed how the TEFS function sets or the intelligent design function sets are used in the people and process ends of the spectrum. This interface technique may aptly represent how this Intelligent Design interface describes all the three ends of the spectrum. The pictorial representation is given below



Interface Inheritance Patterns

Let us now create a few interfaces based on the fully factorized functionality of the intelligent design interface. The simple example of the car engine interface that we discussed earlier translates to the following:

The fully factorized car engine interface:

System.Execute.Start()
System.Execute.Stop()
System.Execute.Accelerate()
System.Execute.Decelerate()

The full factorized interface of software that draws a circle on the computer screen is as below:

System.Design.Draw_Circle( )
System.Data.Fetch_Radius ( );
System.Execution.Database_Call ( )
System.Execution.Use_Connnection_Pooling ( )

Now let's use the fully factorized interface of a software web application (Macro Interface):

System.Design.Web_form_design ()

System.Data.SQL_Query()
System.Data.Stored_procedures()

System.Execution.Project_compilation ()
System.Execution.Project_build()

System.Enabler.Visual_studio_IDE()
System.Enabler.DotNet_software_framework()

System.Optimization.Program_code()
System.Optimization.Queries()

System.Risk.Security()
System.Risk.Roles()

System.Exception.Exception_circumstance_handling()

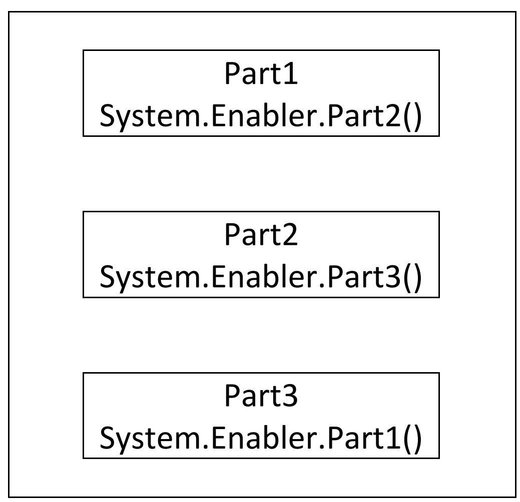
System.Rules.Business_logic()

We have now dealt with interfaces of a variety of types. Now we move ahead into discussing the build or construction of systems which consists of individual, interdependent parts.

The theory of reductionism is an approach that is normally used to understand the nature of complex systems by reducing them into interactions of the subparts. The Intelligent Design functions very easily handle reductionism because they have a built-in function type called the enabler function.

In a hypothetical machine with three constituent parts, if part1 is dependent on part2 and part2 is dependent on part3 and Part3 is dependent on Part1, the interface enabler functions would be as follows:

Complex Machine with three constituent interdependent parts:



Verifications with Science:

We have already learnt about the concept of entropy in one of our previous chapters. The phenomenon of entropy is the process by which disorder sets in naturally in any system. We also learnt about the multiplicity effect, where the probability of throwing a dice and achieving a seven is far more probable than achieving a two.

The Intelligent Design approach is a value added approach similar in analogy to achieving a two with a throw of dice (i.e. there is one way to the best solution).

Products normally follow evolutionary iterative cycles of development. And this leads to a process where there is a chance or probability of missing aspects or functionalities conveyed by the intelligent design approach.

This approach is normally associated in analogy with the throw of dice and achievement of a total of seven in a multiple ways. The end product could be characterized by elements of disorder or complexity.

Therefore we can conclude that the intelligent design approach helps in systematically tackling the disorder forces and allowing the building of sophisticated interfaces which are naturally very revealing about their functionalities.

These functions would thus lead to building smarter and far more systematic products.

Conclusion:

We have now come a long way in our learning exercise – having dealt with the three core ends of the enterprise spectrum viz., People, Processes and Products.

In our successive chapters, we would be outlining various examples of smarter systems that may be built in the distant future.

Chapter 7

A Brief History of Time – A Historical Analysis of Innovation

Until now, we have been discussing Intelligent Design techniques to build smarter people, processes, products and smarter enterprise ecosystems. This can be summarized by a single word – Innovation.

This chapter deals with analyzing a brief history of innovation or rather the evolution of modern life per se.

Innovation is a term we frequently used to represent or identify inventions, but it may not always be the case. The process of bringing about a change by introducing new processes or methods, products and higher forms of communication and thought in individuals is a more apt definition of Innovation. We may recollect that we used the term fluctuation functions that signify an advent of change.

But change itself is not a new phenomenon. Change can be more aptly put forth as constant phenomenon over the ages or cycles of time. We as stated, dwell into historical insights of great changes that took place in society over a period of time. The eventual purpose is to go back in time and celebrate the past, because that's the only way that can prepare us for an appreciation of the future.

T&C Vectors is only attempting to enlighten its readers to recognize these forces of change and align with them for the primary goal of evolution of the individual and the society on the whole.

We may make a note that change can easily be classified as evolutionary or degenerating in nature. Evolutionary change is usually accompanied by forces that make the world we live in a more systematic system. The degenerating forces are those forces that do not have substantial purpose or a higher goal of contributing to the system on the whole.

Although man is classified as a social animal and social percussions have a very strong role in human society, individual perseverance and pursuit have for long contributed to evolution of the masses in societal systems.

From a historical perspective, the above statement stands true very strongly because the innovations that took place then could be achieved by singular effort. However in the modern age the innovations and systems that are created today are so complex that it requires individual as well as a team effort in the process of creation.

While we could easily identify the inventor of a ball point pen, it is nearly impossible to identify the inventors of the computer. For complex systems like computers were invented over a period of time by contributions from many individuals.

Before we go into studying historical timelines of innovation, let's identify a few critical innovations that change the world and impacted society in a very strong way. Research by FPRI had concluded the most important innovations of all times.

1. True semantic, syntactic and phonetic languages that allowed humans to communicate and co-operate with each other.

2. The taming of fire allowed humans to inhabit cold places, work after dark, scare away wild animals and cook food.

3. Simple tools that allowed people to multiply their mechanical effort to getting the work done.

4. Farming was a long chain of activities that involved growing crops and domestication of animals.

5. Clothing permitted people to protect themselves from inclement weather.

6. Earthenware and pottery when ancients discovered hardened clay in their fire pits.

7. Metallurgy permitted people to make hardened tools and spears.

8. Food preservation meant humans to store the food for later consumption.

9. The system of writing that made possible written records and calculations.

10. Religion shaped up human culture and behavior.

11. Invention of wheel changed how man could move goods from one place to another.

12. The manufacture of paper led to explosion in publication and spread of knowledge.

13. Specialization of labor work force classified people into various skilled classes.

14. Discovery of fossil fuels permitted people to harness energy.

15. The concept of science came in only by the 19th century before which it was only considered as natural philosophy.

16. Harnessing of water power or water mills permitted one to carryout heavy work.

17. The invention of steam engine multiplied man's ability to do work and made practical the activity of coal mining which provided fuel for the steam engines.

18. Electromagnetism led to the harnessing of electricity and the running of electric motors.

19. Diesel and fuel based engines converted potential energy stored in fuels to work.

20. The gas turbine engines allowed propulsion of large transportation systems like ships and aircrafts.

21. Invention of computers created the foundation of a digital world which runs in parallel to the physical/material world.

22. The advent of internet that allowed information to flow freely across geographical distances.

This analysis gives us a fair glimpse of how the world changed over a period of time and gives us insights about how the future could play out in due course.

We could conclude that from a few of the points above, the most important directions of innovation that changed the world over a period of time could be classified as:

1. Information and communication.

2. Energy and utilization.

3. Invention of transportation.

4. Creation of the digital world.

We now study the timelines of each of these individual directions of innovation to draw conclusions and, if possible, extrapolate to the future.

Historical timeline of Information and Communication:

3500 B.C. - Phoenicians develop and alphabet. Sumerians develop cuneiform writing - pictographs written on clay tablets.

1775 B.C. – Greeks use a phonetic alphabet written from left to write.

1400 B.C. – Record of writing in China on bones.

1270 B.C. – The first encyclopedia is written in Syria.

900 B.C. – The first postal service for government use in China.

776 B.C. – First record of homing pigeons used to send message.

530 B.C. – Greeks started the very first library.

500 B.C. to 170 B.C. – papyrus rolls and early parchments made of dried reeds – first portable and writing surfaces.

200B.C. to 100B.C. – Human messengers on foot or horseback common in Egypt and china.

14 A.D. – Romans established postal services.

100 A.D. – First bound books.

105 A.D. – Tsai of China invents paper as we know it.

305 A.D. – First wooden printing press.

1450 A.D. – News papers appear in Europe.

1455 A.D. – Guttenberg invents the printing press.

1560 A.D. – Camera Obscura invented – primitive image making.

1650 A.D. – First daily newspaper.

1714 A.D. – The invention of the typewriter.

1793 A.D. – First long distance semaphore telegraph line.

1814 A.D. – First photographic image.

1821 A.D. – First microphone.

1835 A.D. – invention of Morse code.

1876A.D. – The invention of the telephone.

1888A.D. – First role film camera.

1899 A.D. – first recordings using magnetized steel tape.

1910A.D. – First talking motion picture.

1923A.D. – First television camera.

1925A.D. – First experimental television signals.

1944A.D. – Computer like *Harvard's mark I* – the age of information science begins.

1969A.D. – The first internet started – ARPANET.

1979A.D. – First mobile phone.

1994A.D. – World Wide Web (WWW) taking communication at light speed.

We may note that it was form of communication as well as the medium of communication that changed over a period of time. If we extrapolate the patterns of communication into the future, we are likely to find only two possible forms of change.

1. Likely change in Format of information / Form of communication.

2. Likely Change in the Medium of communication.

The format of information of communication that exists as of today exists in the form of language, written and spoken. The T&C Vectors Intelligent Design methodology is working on deriving a new form of intermediate technical language that we've already suggested. The format or structure of the information that we convey through education and learning, in a variety of subjects could also use the intelligent design methodology.

As for the medium of communication is concerned, the world is moving ahead into digital classrooms and Visual communication, further extrapolations may lead to more sophisticated digital mediums of communications.

T&C Vectors however suggests that great value added work lies in the conversion of unstructured forms of information and theory into structured, intelligently designed theory. It may also be noted that if the format of information or theory is closer to a functional based approach as suggested by the Intelligently Designed theory of intermediate language, the process of conveying or relaying this structured information through a digitized medium may be far more feasible.

Historical timeline of Transportation:

3500 B.C. – First wheeled vehicles in history.

3500B.C - River boats with oars.

2000B.C. – Domesticated horses used for transportation.

770 A.D. – iron horseshoes improved transportation by horse.

1492A.D. – First illustrated theories of flight by Leonardo Da Vinci.

1783A.D. – Invention of the first steam boat.

1783A.D. – Invention of Hot air balloons by Montgolfier brothers.

1790 A.D. – Modern bicycles.

1801A.D. – First steam powered locomotives.

1862A.D. – Gasoline engine automobile.

1867A.D. – The first motor cycle invented.

1885A.D. – The first practical automobile to be powered by an internal combustion engine.

1903A.D. – The first engine airplanes.

1926A.D. – Liquid propelled rockets.

1940A.D. – Modern helicopters invented.

1947A.D. – Supersonic Jet flight.

1970A.D. – First jumbo jet to carry passengers.

1990A.D. – Electric hybrid automobiles.

Having seen the timelines of innovation and transportation, it may be inferred that the transportation products domain is now fairly matured with availability of very efficient engines and transportation systems. With further research in the Intelligent Design approach, superior transportation systems could be developed. .Again, transportation can be divided into:

1. The core product / medium of transportation e.g. Cars, rail, flight.
2. The processes that govern the Transportation system.

The Intelligent Design approach could easily work out new structured processes that optimize the entire domain of transportation processes. We may note that algorithmic flight scheduling and rail system scheduling already exist in many developed nation systems, however there is still scope for an intelligent design approach in optimizing transportation and traffic systems.

Historical timelines of harnessing of energy:

1500 B.C. – Geothermal power used for cooking, bathing and heating.

200B.C. – Mining of coal.

644A.D. – Vertical axes windmill recorded in Iran.

1765A.D. – Invention of steam engine.

1780A.D. – Coal is used to generate electricity.

Till 1800 A.D., mankind's use of energy relied of muscular and biomass sources. Energy was provided by manual labor, animals and biomass (wood).

1800 A.D – Invention of the Battery

1821A.D. – First natural gas drilling in New York.

1885A.D.- Invention of Gasoline powered automobile engine.

1890A.D. – Solar energy used to run a printing press.

1890A.D. – First operating wind turbine.

1891A.D. – Invention of Tesla Electric coil by Nicholas Tesla.

1892A.D. – Invention of Diesel engine.

1938A.D. – invention of Nuclear fission (Otto Hann and Fritz Strassman).

1944A.D. – First nuclear reactor.

1950A.D. – First alternating current wind mills.

An analysis of timeline of energy reveals maturing of energy producing techniques capable of generating enough power to sustain working of large enterprises, societies and nations. Scope for innovation in the field of Energy could be classified into:

1. Production / Generating techniques.

2. Energy consumption patterns and processes.

The Intelligent Design techniques as suggested can help in finer research in the energy generation space, but could be far more effective in improving energy consumption patterns or processes.

We have discussed three critical timelines of historical innovation that covered the different ages of history viz., The Early age, The Medieval age and The Modern age. The present age that we live in today is often addressed to as the information age.

If patterns of history could convey what the future beholds, we could well be moving into what could be called as The Intelligent Age or The Systems Age.

T&C Vectors is envisioning the advent of a systems age, or more commonly a smarter world.

The systems approach concentrates on structured and intelligent governance of systems and its constituent entities in a manner, so as to allow a methodical, structured and sustainable future, a future that encourages multiple enterprise systems co-existing in equilibrium, through standard methodologies and protocols of techniques and behavior.

Having discussed the nuances of intelligent design and historical consciousness, we are in a position to successfully prepare for the times to come.

Chapter 8

A Systematic World Space: Patterns of systematic Innovation

For the purpose of a systematic comprehension of the world, we have already discussed the people, the process and the product ends of the spectrum. However, the comprehension is incomplete without a chapter on systematic innovation that deals with innovation at the people level, the product level and the process level and is measurable through a generic framework of innovation.

Industry always faced a challenge of each industry measuring innovation in its own way. It is because of this reason that there are no standard metrics that define innovation or measure innovation. T&C Vectors has developed the innovation framework based on the intelligent design theory or the TEFS function set that gives a standard framework for the enterprise in any domain to measure its level or accomplishment in innovation.

Locating a particular place in a geographical map requires two co-ordinates, the latitude and the longitude. Similarly, the procedure for measuring innovation involves two core steps:

1. Identifying the innovation space that the enterprise occupies.

2. The level of or the degree of innovation in < products, people and processes space >.

Identifying the innovation space:
To evaluate the first point of identifying the innovation space, we outline two simple matrices. They are as follows:

Products, OEM (Original Equipment Manufacturers)
Services
Trade or distribution

The Enterprise Matrix

The enterprise matrix divides an enterprise into three distinct types:

a. **Product/OEM** – Companies which build products or equipment for use by public or other enterprises.

b. **Services** – Companies that offer some form of service e.g. accounting services, financial services, software services or hotel accommodation services.

c. **Trade or Distribution** – Companies that distribute products from manufacturers to the public or other enterprises e.g. E-Commerce stores, retailers, distributors, shops.

Enterprise	Software/Digital world	Physical world
People	a	b
Process (internal or external)	c	d
Products	e	f

The Enterprise Medium – Software World or Physical World

The interpretation of the above Enterprise Medium matrix is as follows:

a. Enterprises that have “people” as its core entities e.g. Social networking and social media sites like Facebook, Twitter, Orkut etc.

b. Enterprises that have “people” as its core entities, but conduct their exercise in the physical world by group gathering and one to one, in person meeting.

c. These are enterprises which have “process” as their core entity of offering, in the software world.

d. These are enterprises which have “process” as their core entity of offering, in the physical world.

e. These enterprises are “product” companies which build software products e.g. Microsoft, Apple etc.

f. These enterprises are “product/OEM” companies which make products which are tangible, in the physical world.

We use the first matrix (the Enterprise matrix) and the second matrix (The Enterprise Medium matrix) to pinpoint the space that the enterprise occupies in the systematic world map.

The level or the degree of innovation:

Having identified the innovation space that the enterprise occupies, we move into the process of identifying the degree of maturity of innovation of the individual entities - people, processes and products.

To solve this, we outline the second degree intelligent design or the TEFS functions and call the same as Innovation matrix.

This innovation matrix forms the framework that we mentioned earlier, that can be used to measure innovation in different domains and enterprise spaces.

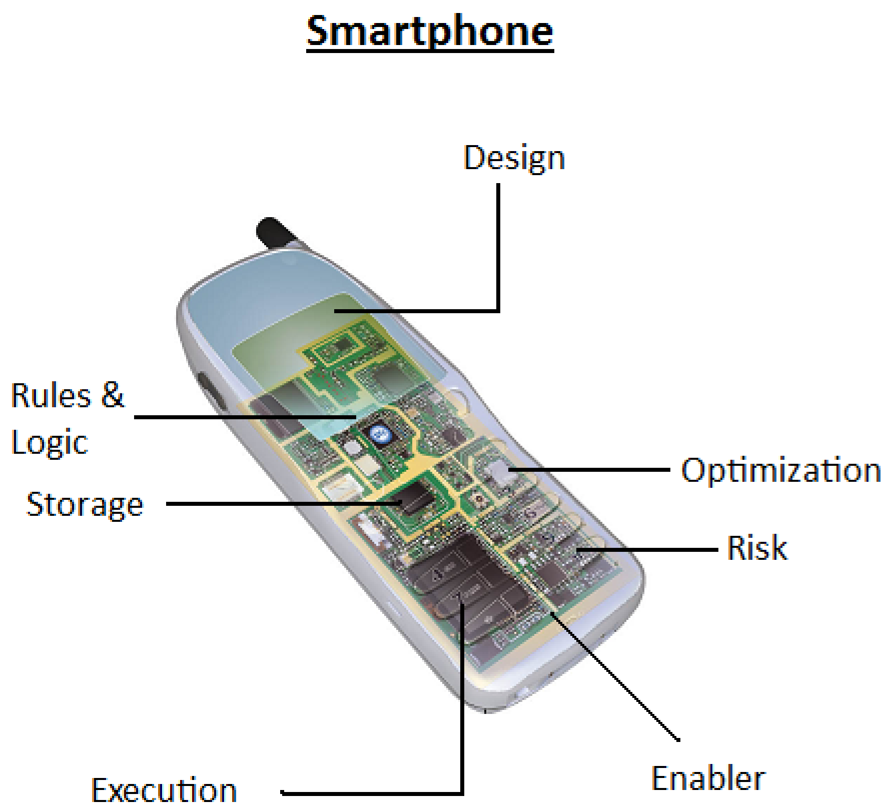
Systematic Innovation Framework

	Design/ Structure	Execution	Data/ Storage	Risk	Optimization	Enabler	Rules/ Boundary	Exceptions
Design/ Structure	Design-Design	Design-Execution	Design-Data	Design-Risk	Design-Optimization	Design-Enabler	Design-Rules	Design-Exceptions
Execution	Execution-Designed/Structured	Executable	Execution Data	Execution Risk	Execution Optimization	Execution Enabler	Execution Rules	Execution Exceptions
Data/ Storage	Data Structure	Data Execution	Meta Data	Data/ Storage Risk	Data Optimization	Data/Storage Enabler	Data rules	Data Exceptions
Risk	Risk Design/Structure	Risk Execution	Risk Data	Risk of risk	Risk Optimization	Risk Enablers	Risk rules	Risk Exceptions
Optimization	Optimization structure/design	Optimization execution	Optimization Data	Optimization Risk	Optimization of Optimization	Optimization enablers	Optimization rules	Optimization Exception
Enabler	Enabler Design	Enabler Execution	Enabler Data	Enabler Risk	Enabler optimization	Enabler of Enabler	Enabler Rules	Enabler Exceptions
Rules/ Boundary	Rules Design	Rules Execution	Rules Data	Rules Risk	Rules optimization	Rules Enabler	Rules about Rules	Rules Exception
Exceptions	Exceptions Design	Exceptions Execution	Exceptions Data	Exceptions Risk	Exceptions Optimization	Exceptions Enabler	Exceptions Boundaries	Exceptions among Exceptions

In conjunction with this matrix, we also use a measurement scale to measure the degree of each function of innovation.eg 70% Execution optimization or 10% Execution Risk.

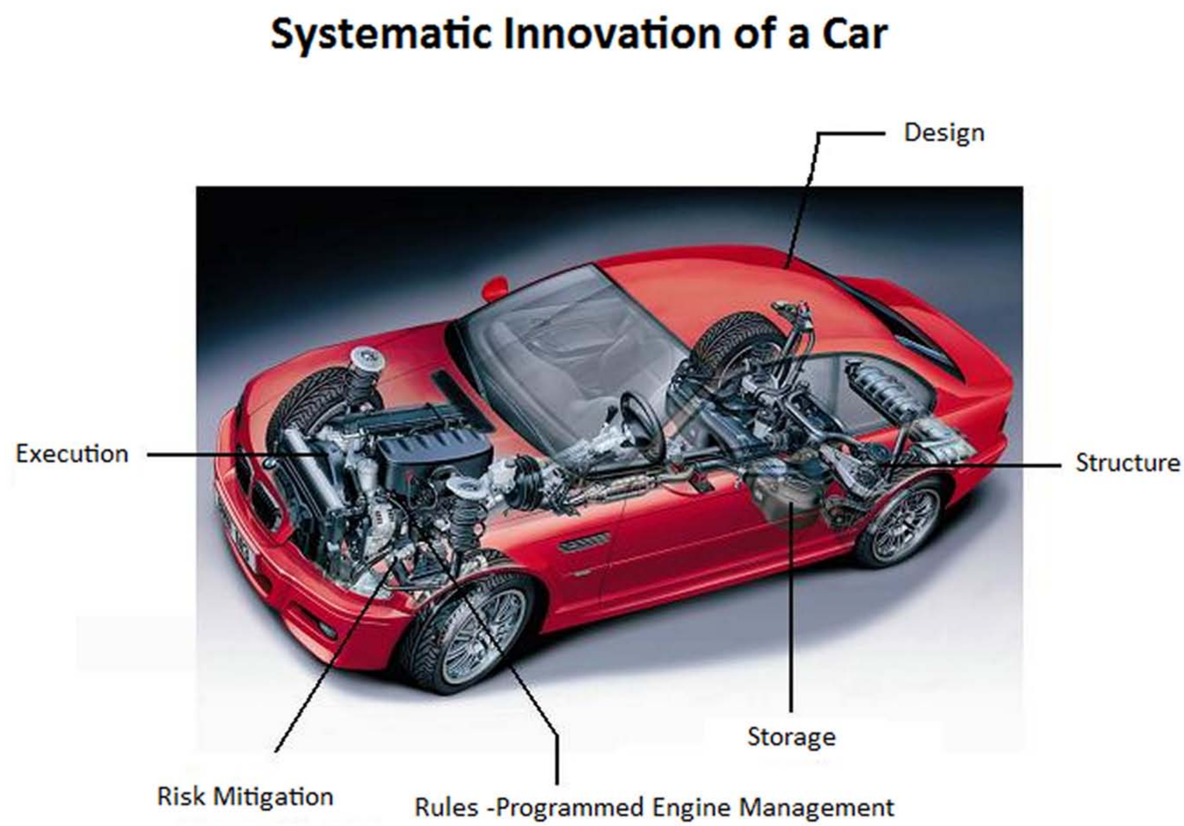
Let us work out a few examples to see how the systematic innovation framework helps in the measurement of innovation and outlining new directions of innovation. We discuss systematic innovation in the product space and the Enterprise space.

The systematic innovation approach to building a smartphone:



Execution – Building a smartphone with a faster processor.
Execution Optimization – Building a smartphone using an optimized power saving processor.
Storage – Building a smartphone with higher capacity of data storage e.g. 1GB, 2GB, 4GB etc.
Design – Building a smart interface for the phone like a tiled interface or a touch interface etc.
Structure – Building a smartphone form factor with a variety of blends of plastic, carbon fiber or metal.
Risk – Risk mitigation in a smartphone by incorporating bio-metric security for authentication.
Enablers – Building a smartphone with long lasting, high capacity batteries.
Rules – Building a smartphone with proprietary or open operating systems that run all the system logic and functionality.
Risk Execution – In case of smartphone theft, the system would lock up and send alerts.

Example 2 - Systematic Innovation of a car:



Execution – Building a car with more sophisticated, powerful engine.

Execution Optimization – Building a car with a fuel efficient engine.

Storage – Fuel storage and boot space or luggage storage space.

Design – The exterior design of the car.

Structure – Building the car chassis out of sophisticated metal alloys or carbon fiber reinforcements.

Risk – Design crumple zones in the front and exteriors.

Rules – Efficient engine programming and management processors.

The innovation process can be classified into:

1. Fundamental out of the box innovation.

2. Systematic Iterative innovation.

Fundamental out-of-the-box innovation involves building a breakthrough product from scratch while, systematic innovation involves innovating in cycles or iterations to bring out the next version of the product.

Let us analyze the software world space to get a perspective on how things have evolved systematically over a period of time.

Talking about computer programming languages – languages were initially unstructured (unstructured execution) succeeded by procedural and structured programming languages (Structured execution). It was further followed by the object orientation methodology. If patterns could forecast a future - it could well be an “object” programming paradigm in the near future.

Talking about evolutions of nations – Nations were formed after long histories of war. People learnt to form Governments and created law. Nations evolved to build national security. Nations built education systems to spread knowledge. Industries and Enterprises were built over time. Nations engaged in global Commerce. If patterns could again guide a future – it could well be the formation and governance of “nation systems”, through sophisticated frameworks of intelligent design.

Today, data analytics and data search is a common place term. However, the data or information that is dealt with is largely in the form of unstructured data. A transition or evolution in data spectrum would extrapolate to analytics and search of “structured” data. An example of a structured data paradigm is represented by the Wikipedia.

Search engines like Google, Bing operate on unstructured data. T&C Vectors is currently working on a world space intelligent search engine that operates on structured data.

According to the systematic Innovation framework, progress in innovation could be possible in any of the directions guided by the 64 Elemental Functions.

In Enterprise systems – Innovation could take place in the redesigning of the rules of the enterprise.

Avionics systems could be programmed to allow structured exception handling sequences in cases of emergency.

Search engines could operate on meta-data (Data about Data) instead of operating purely on data.

Project Management systems could encompass the entire spectrum of intelligent design activities defined in the Innovation function set.

In fact T&C Vectors has a product Enterprise Focus in its portfolio that allows advanced Project Management and governance of the entire enterprise systems.

Thus we can conclude this chapter by inferring that innovation is the process of inventing the next new by breaking the boundaries set by the previous iteration.

Innovation actually follows patterns and trends represented by the innovation matrix and applications of the innovation matrix in specialized domains may yield systematic results.

Chapter 9

Rudimentary Economics and Financials

Our discussions till now have been following a systems perspective in an analysis of the world. It goes unsaid that commerce and finance are equally important functions related to the enterprise space that usually gather attention as subjects of study in courses like business studies, economics etc.

From the systems engineering angle, these courses and fields of study are classified as abstract fields of study. Although subjects like economics used in conjunction with subjects like statistics have a slightly higher basis of scientific foundation, they are still considered imprecise by many factors. The reason being, these subjects deal with heterogeneous data OR Imprecise data.

The intelligent design approach or the systems engineering approach to studying economics and commerce follows one single tenet that, it classifies and structures data, to a synthesized smarter data, that eventually leads to precise measurements of the state of a system.

The world that we live in is actually classified into a variety of ecosystems or systems like the enterprise system or nation system. Each of these systems actually follows very scientific working principles of nature for their execution and governance. In other words, a system runs well when people, processes and products follow efficient trends of evolution in execution and are governed by systematic rules of governance.

Any deviation or inefficiency in processes, people or products leads to a deterioration in the working of the system. It is because of this reason that we observe breakdowns in multitude ecosystems resembling a phenomenon called a recession. We may highlight that enterprise systems and public systems are interdependent for their mutual growth (intellectual growth as well as commercial growth). Whenever, either of these spaces, the enterprise system or the public system undergoes significant deterioration, they lead to a cascading breakdown in the world ecosystem. This is a systems interpretation of the economic term recession.

This ultimate goal of any system therefore, translates to operating in equilibrium with other interdependent systems. It is with this reason that a systems engineered enterprise differs very largely from a business enterprise. Business enterprises allow room for human emotions like greed and capitalism to prevail over the larger principles of eco-systemic equilibrium, while systems engineered enterprises allow no room for forces that instigate deterioration in equilibrium.

The world of Finance:

Since finance is the foundation of commerce, it is an imperative subject for one and all occupying the enterprise space or the public space. Though finance is extremely complex subject of study today, a systems engineering approach simplifies finance into three simple categories:

1. Product Finance.

2. People Finance.

3. Process Finance.

Product finance deals with all the cost price in building a product and selling price of the product.

People finance deals with cost of people required to building products.

And process finance deals with financial aspects associated with the processes that lead to the manufacture and supply of products.

Let us take the building of a computer as an example.

Product cost – cost of individual components.

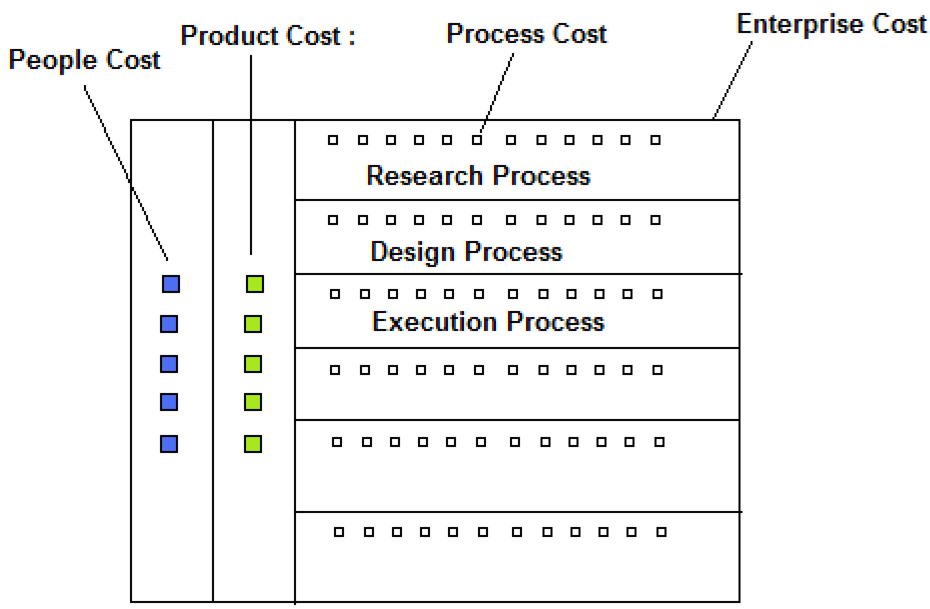
Process cost – Cost of the assembly process.

People cost – Cost of the employing people for various tasks.

The summation of these three costs gives us the total input cost. The price that the product sells at multiplied by its volume gives us the output cost. The difference between the input cost and the output cost is the amount of wealth created by the enterprise.

Enterprise Finance and Accounts can be easily conveyed through the following diagram.

Enterprise Finance : A Systems Approach



People cost = Salaries + Perks

Enabler Cost:

Product cost = Machines for Manufacturing ,Servers,Computers ,Furniture,Assemply Components,Materials, etc etc .Enterprise Cost = Office space ,factory building etc..

Process cost =cost of assembly,cost of Manufacturing,cost of designing and product prototyping etc

each process is further subdivided in to the following costs :
Design cost , Data/Storage Cost ,Risk Cost , Execution Cost ,Optimization Cost , Rules-Process-logic costs , Exceptions Cost

Costing is Further Divided into two types

 # OneTime Cost

 # Recurring Costs

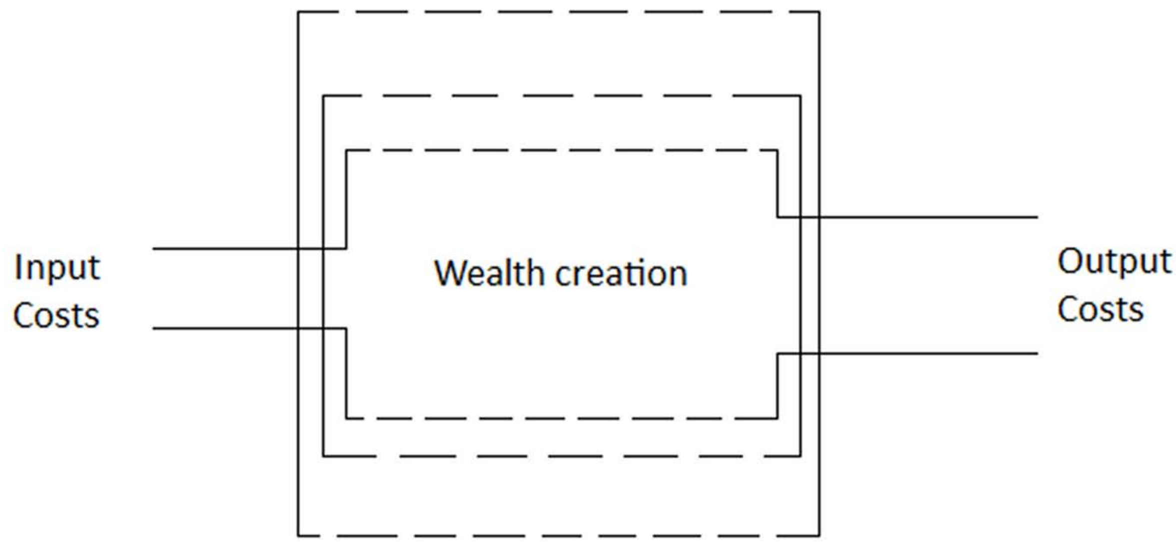
Time Value : Value or cost may

1) Appreciate with Time

2) Depreciate with Time

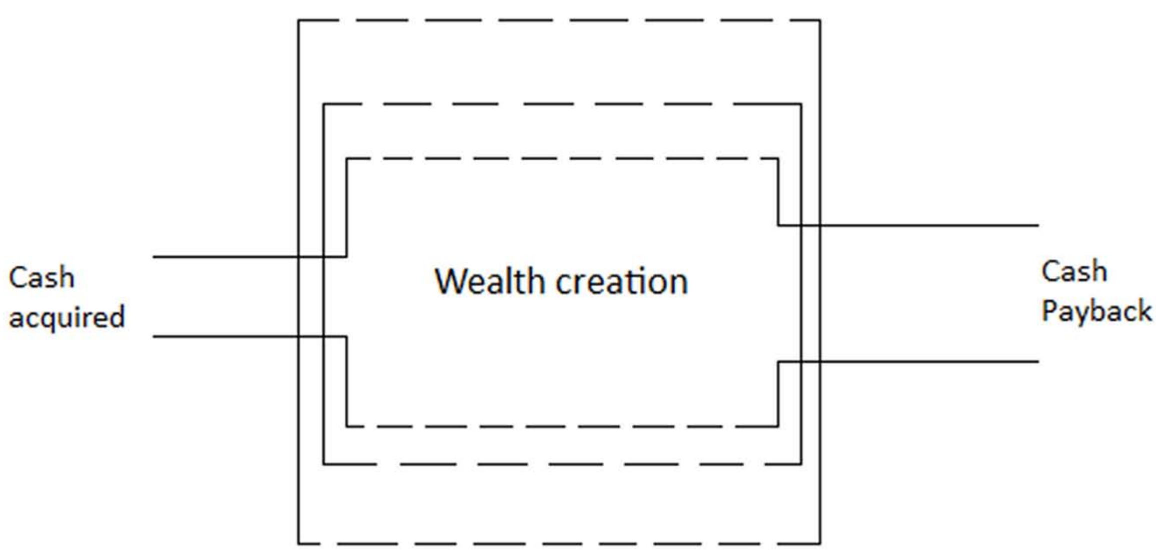
The Pictorial diagram shown in the next page explains a simple systems analysis approach to finance.

Financial analysis as a systems approach



A few other points about the financial world – Industry in the real world operates through stock markets where stock markets are a medium for an enterprise system to acquire cash from the public system for its operations and capital needs. From a systems engineer's perspective, the cash or a payout by any enterprise system back to the public system defines a payback.

A systems analysis of the stock market principle



A simple analysis of the stock market principle leads us to infer that the price of a stock should have relevant dependence on the payout of the enterprise rather than on the notional values or price to equity multiples that the stock sells at or is valued at.

From a systems engineering perspective, terms like market capitalization and wealth created are false financial jargons. The only true measurement that ultimately qualifies is the amount of cash acquired and amount of cash paid back by an enterprise. In this context, if an enterprise promises not to give away any payback, the value of the stock in the stock market can only be considered notional .There is an alarming trend of firms in the world today who do not contribute to paying back to the public systems.

Sometimes, it is these sorts of imbalanced working or governance principles that lead to deterioration of commercial wealth in the world ecosystem. E.g. Thus if a company borrows debt and never pays back, the public system that loans the debt may find itself in a financially deteriorated position,unable to further support newer capital to upcoming enterprises.

It may also be inferred that notional wealth or market value of stock prices is only created by successive generations of investors forcing up or escalating stock prices rather than the payout from an enterprise system doing so.

Keeping this in mind we wind up our rudimentary yet complete analysis of economics and finance and understand all the important governing parameters that lead to ecosystem stability or instability.

In the Next Chapter we wold introduce the Concept of an Innovative Financial Tool called the “**Community MutualFund**”

Chapter 10

Outlining the future – Array of Innovative Possibilities

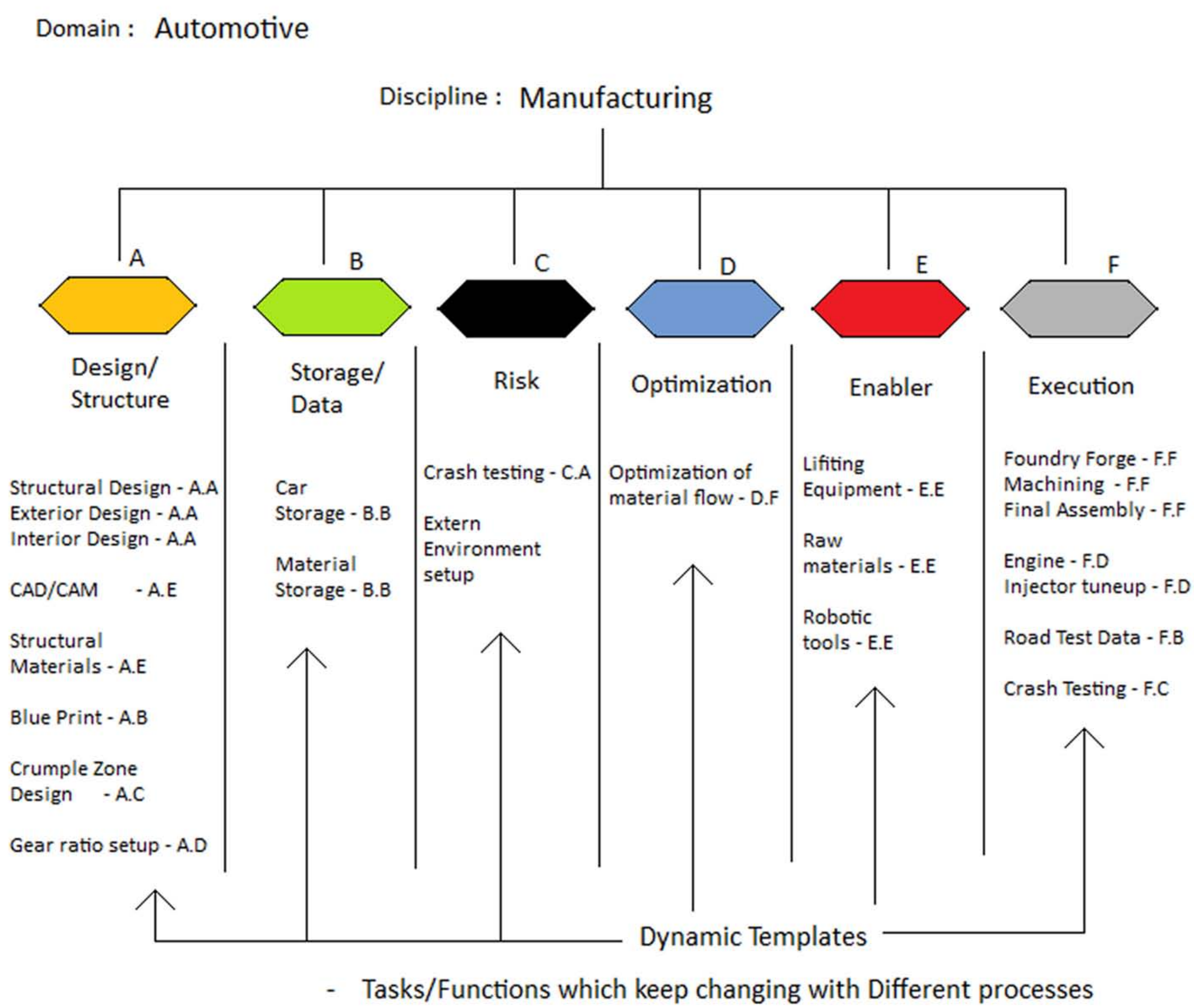
This chapter is a collection of possible areas of innovation that could possibly be demonstrated in the future. Since T&C Vectors does not have specialization in each of the possible areas of innovation, it is only outlining a template or guiding theme. The areas that are outlined are as follows:

- **ERP Systems – Multi Domain / Multi Discipline.**
- **System Software – Development e.g. compilers, operating systems.**
- **Object Programming Paradigm – Extension to OOP Methodology.**
- **Object Based Database Systems.**
- **Game Design.**
- **Artificial Intelligence.**
- **Computer Hardware Systems.**
- **Applications in Government/Nations Systems.**
- **Standards framework.**
- **Applications in Machine Systems.**
- **Enterprise to enterprise, Enterprise to public Digital engagement.**
- **Digital Economic Lifeline Systems .**
- **Precision Engineered Strategic Data to Manage the world –Enforce.**
- **Community Mutual Fund**

ERP Systems – Multi Domain / Multi Discipline:

According to our systems engineering theory, we have divided the enterprise space into people, processes and products. ERP systems are enterprise resource planning systems which help in managing the processes of an enterprise. ERP systems as of today manage specialized processes like human resources, payrolls, manufacturing etc.

Future ERP systems are envisioned to be not only domain neutral but also process neutral i.e. the ERP systems could fit into any kind of industry and any kind of process. We have already demonstrated how domain neutrality and process neutrality are feasible through the introduction of the intelligent generic process design. The following picture would convey the generic methodology.

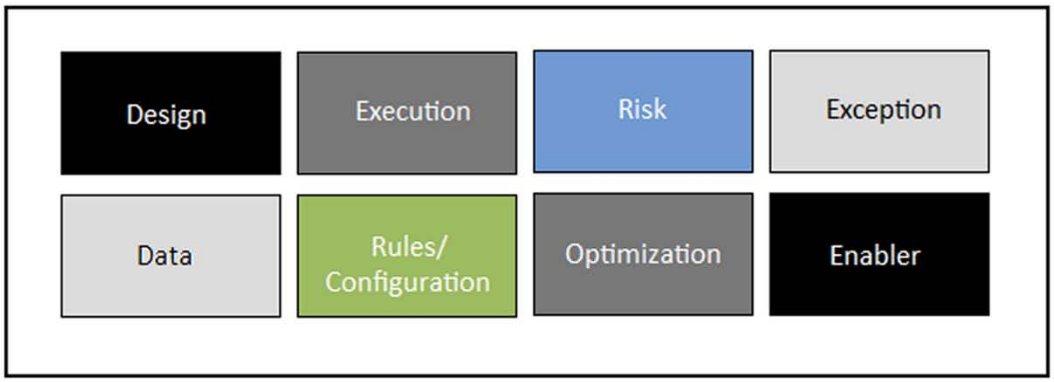


<u>Future ERP Systems</u>

System Software – Development:

We have a wide variety of system software available today, but most of the system software is proprietary in nature and in features. The Intelligent Design approach helps in eliminating proprietary standards and leads to building standards based software, each varying from the other in scientific functional features. We have compiled a sample operating system interface for an appreciation.

Operating System Interface Sample



- Design Software
  - D.E - CAD/CAM
  - D.E - Photoshop
  - E.D - Visual Studio
- Data Software
  - B.E - DB2
  - B.A - Winword / Adobe PDF
  - B.B - File Storage
- Risk Software
  - C.E - Bitdefender Antivirus / Norton
- Enabler Software
  - F.A - DirectX / OpenGL / PhysX Libraries
  - F.E - .NET / JAVA Framework
  - F.C - BioMetric Authentication
- Optimization Software
  - Harddisk Defragmentation s/w
  - Clean up software

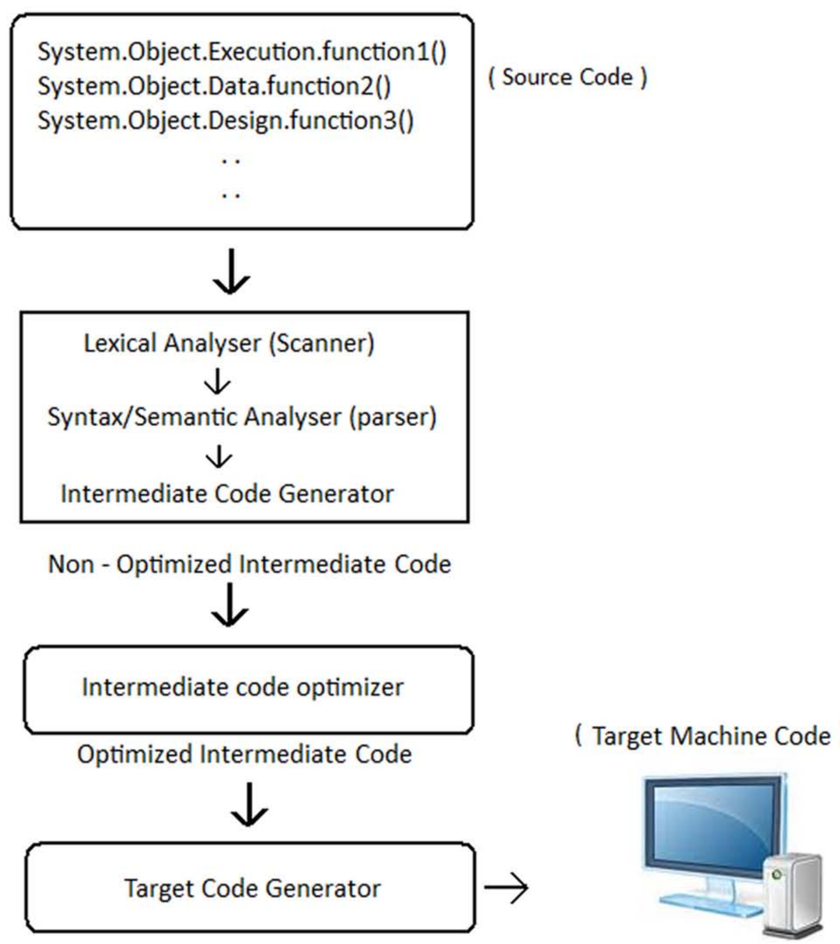
- Execution (General Purpose)
  - Communication - Mail / Messenger
  - Modem Connect
  - Exception Software (Not default to an Operating System)
    - Games
- Rules / Boundaries
  - OS configuration
- Exception — Events / Errors Viewers

Design / Structure **A**	Data / Storage **B**
Risk **C**	Optmization **D**
Execution **E**	Enabler **F**

Compilers can be designed adhering to the principles of intelligent design such that all functions describing a class can be represented in a factorized manner as suggested by the interface based approach. In a truly object oriented software, all objects inherit from the base class object called “**System.Object**”. In case of Intelligent Design Software, the class files would be further classified as “**System.Object.Execution.xyz_function**” depending on the classification of the function.

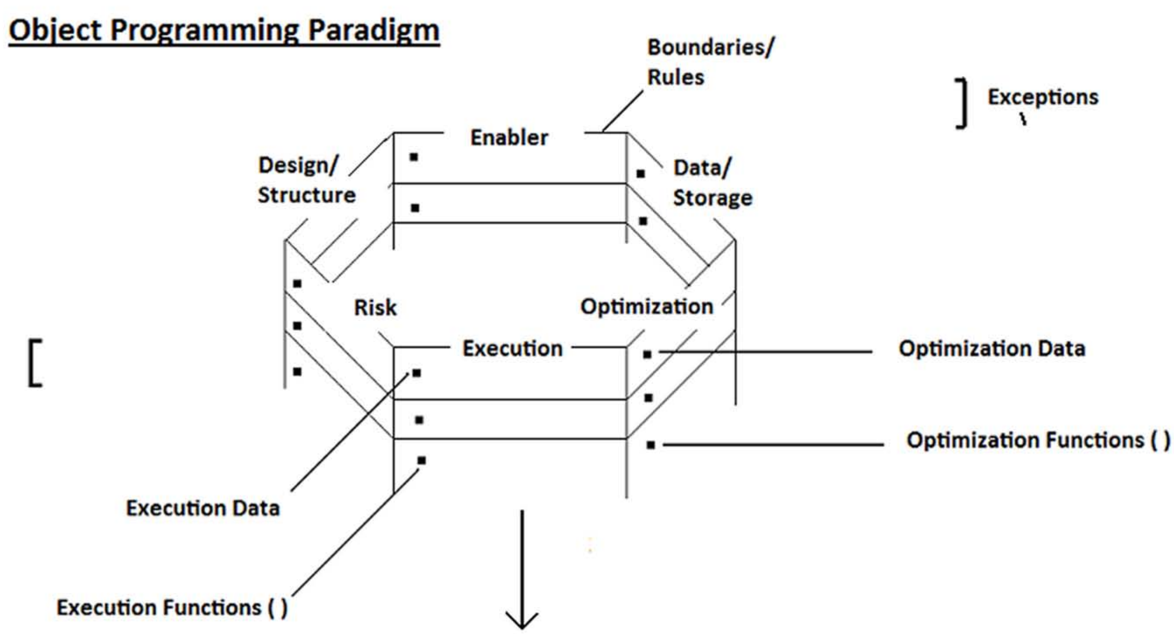
This would improve readability of the code as well as allow certain types of functions to be dedicated to specialized hardware such as the GPU (Graphics Processing Unit). The “**System.Object.Design**” functions can be directly allocated to the GPU. Mathematical functions of execution could be allocated to the CPU (Central Processing Unit).

Compiler Operation



Object Programming Paradigm:

As of today, the highest programming paradigm that exists is the Object Oriented Methodology or Object Oriented Paradigm (OOP). The programming technique has been described as “object oriented” rather than “object” programming because, structured programming evolved into the methodology of describing objects or systems through classes. With the breakthrough in intelligent design approach, the object oriented paradigm may be evolved further into object programming paradigm where all the functions and data of object oriented design are fully factorized and represent pure objects.



Software class file

```
Namespace
{
        Class_name class
        {
                System.Object.Execution.function_name();
                System.Object.Design.function_name();

        }
}
```

Object Based Database Systems;

A large variety of Database systems are available today namely relational databases, object oriented databases etc. The Intelligent Design approach can lead to the development of object databases where the primary entity of storage is a free scaling object. Let us take an example of relational database where we need to map a list of URL links to a specified key word, for example, the keyword computer should map to links www.abcd.com, www.xyz.com, www.123.com etc. The way the database would be structured is as follows (ignoring XML key field methodology):

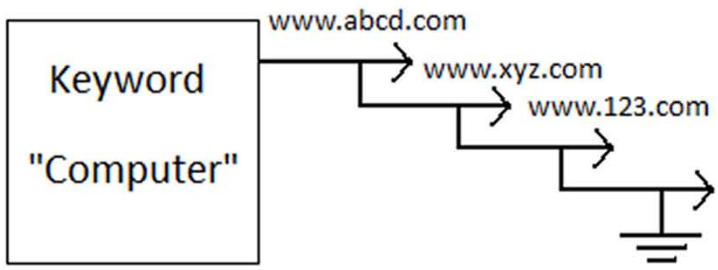
Primary key (Keyword)
Computer
Server
Interface

Keyword	URL Link
Computer	www.abcd.com
Computer	www.xyz.com
Computer	www.123.com
Server	www.abcd.com
Server	www.a1b2.com

RDBMS table structure representing keywords to their URL Links

If we approach the intelligent free scaling approach to design an object database, the database structure would function as follows:

Free Scaling Column - Object Database Design



The database field consists of scalable linked lists that store data in an optimized way.

Game Design:

Today's game design software's are graphically astounding. And there are a wide range of titles available with various types of game play. The intelligent design approach could improve upon the game play of a variety of games by involving all the TEFS functions or the Intelligent Design functions in a game sequence.

The game play sequence could be evaluated such that structured execution maneuvers yield better results than an unstructured maneuver.

Games could be designed for simulating enterprise systems and world systems .Younger Players could be introduced to advanced intelligent approach to learning rather than games acting as only a source of entertainment.

Artificial Intelligence:

Artificial Intelligence is the field of Engineering involving, building of intelligent systems that resemble Human Intelligence or human intellectual capabilities.

The Intelligent design Functions are very useful in the design of intelligent systems, because the functions help in building a very sharp context.

Much has been criticized about machine systems not being able to think like humans do. We beg to differ. Human thinking is not structured in many ways. Humans make assumptions that their thought process is superior to the ways of the machine because they use intuition for judgment. It may be noted that intuition is nothing but an emotional process. E.g. an assessment of Risk in a scenario can lead to a variety of judgments by different individuals, depending on their emotional coefficients. A conservative nature or emotion would lead to a different assessment from that of a risk taker.

In contrast Machine thinking or logical thinking infers judgment through an assessment of all the variables of data it has in context.

It may therefore be improper to draw inferences of superiority of human intelligence over machine thinking.

Given a proper context machine logic is far superior. T&C Vectors is currently working on a Search Engine (Inter-Xect) that tends to use smarter Data to build contexts and offer relevant results, a process similar to artificial Intelligence.

Artificial Intelligence

Simple Artificial Intelligence

Cognition → Inference

Artificial Intelligence involving building of a context.

Context Level 1

Which Domain does it belong to?

- Aviation
- Computer Science
- Electronics
- Engineering
- Marine / Naval
- Rail Transport
- Automotive
- . .
- . .

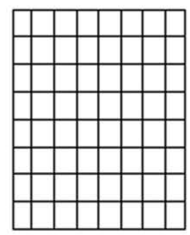
Context Level 2

What are you talking about?

- Entire system
- People
- Process
  - Marketing
  - Petroleum Refining
  - Software Development
  - Manufacturing
- Product

Context Level 3

What exactly are you talking about?



64 Intelligent Design / TEFS functions

- Execution
- Optimization
- Execution Risk
- Data-Data
- Data Design
- . .
- . .

Examples of Artificially Intelligent Queries :

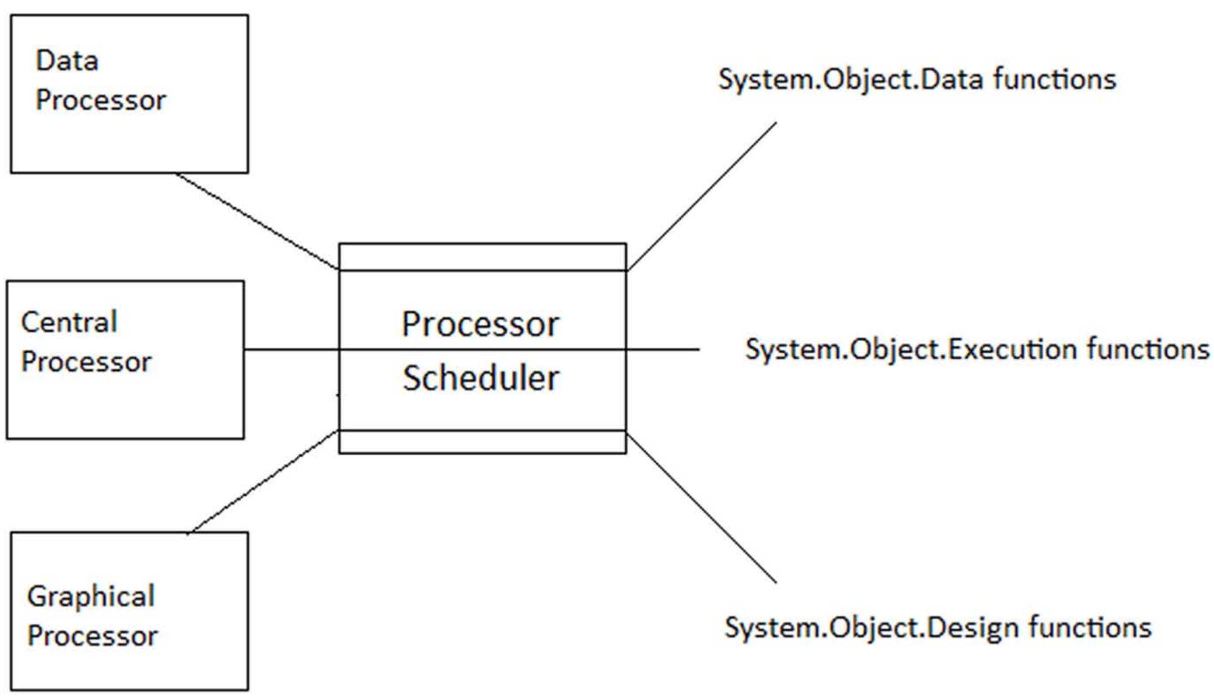
1) Software - People - Execution => We are talking about people's programming skills in software like C#, C++.

2) Aviation - Product - Execution => The process how the product works. If the product is an aircraft, the inference is "how an aircraft flies?".

3) If you are talking about Marine / Naval - Process - Design => Architecture and design of Ships / Marine products.

Computer Hardware Systems:

Computer architecture could follow the EDIC (Explicitly Distinct Implicitly Coherent) principles of design such that we might have specialized processors for graphical processing, for mathematical execution and for data fetch and store functionality. The processors could be coordinated by the help of schedulers allocating data tasks to the data processors, execution tasks to execution CPUs and Design tasks to Graphical Processors. The reason for such a split or such an explicitly distinct design could be because different kinds of tasks require different frequencies of operation or execution, for example, data fetching tasks are much slower compared to GPU tasks. Such architecture could be useful in Servers.



Applications in Government/Nations Systems:

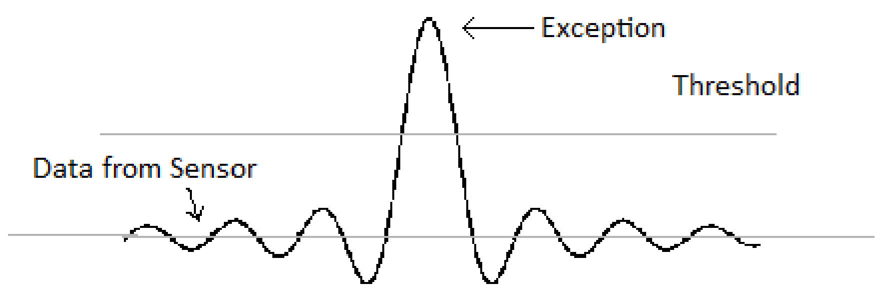
The TEFS functions have wide range of applications in the Government and Nations systems space. The applications could involve codification of national laws searchable by search engines or structuring of patent data in a format searchable by search engines. Other applications could be right to information systems that broadcast important information to the public.

Standards Framework:

The TEFS functions or the Intelligent Design functions can be used in an open standard framework that can be utilized by industries belonging to varied domains. For example, it could be used by manufacturing industry or software industry as a standard.

Applications in Machine Systems:

a. Applications of Sensors in Machine Systems to measure exceptions occurring during machine runs. Similar principles could be applied in a variety of systems where measurement of data is feasible.

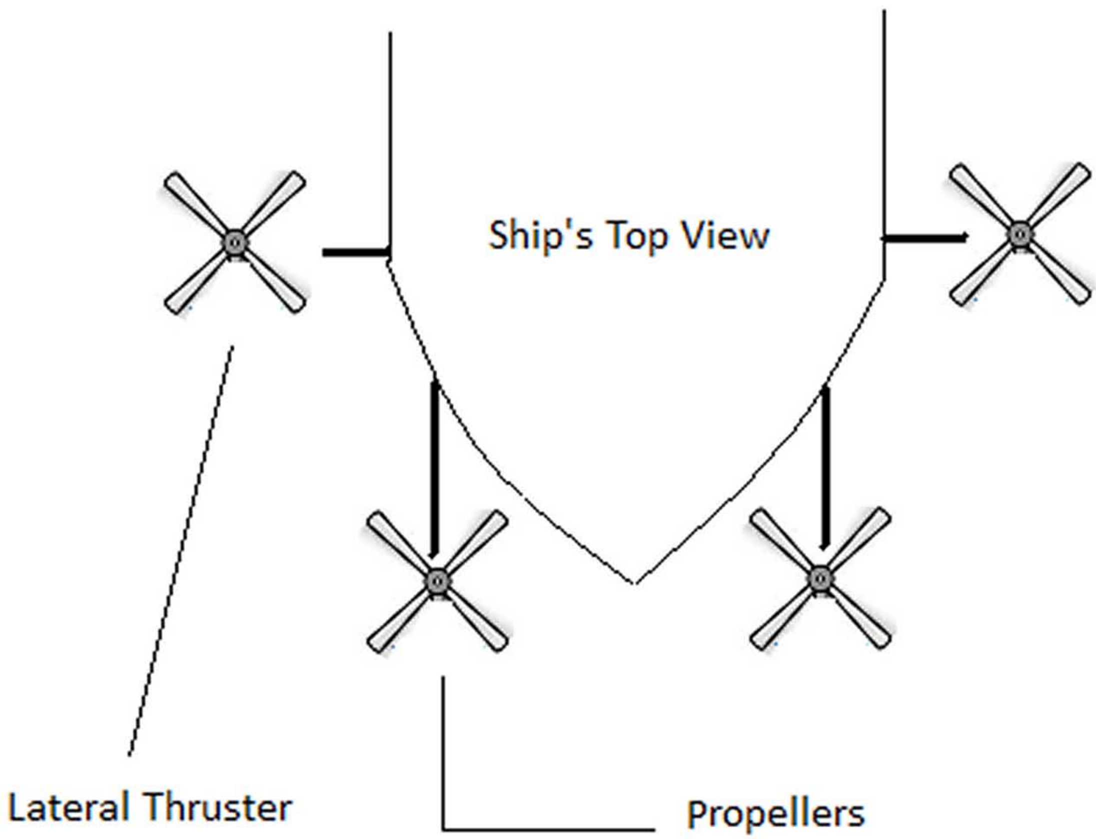


b. Possible futuristic fuel injection systems in automobile engines where the engines could be programmed to fire on individual cylinders or multiple cylinders depending on the need of the situation. For example:

engine RPM	Executing cylinders
Idle engine RPM e.g. 900RPM or below	Execution of only cylinder 1 sustaining minimum idling of engines.
>900 RPM up to 2000RPM	Higher fuel air mixture in cylinders 1,2 &3
>3000 RPM	Highest fuel air mixture in cylinders 1,2,3 & 4

c. Tilting seats in automobiles – The seats of an automobile or a car could be designed to accommodate higher degrees of freedom so as to allow the rotation of seats during centrifugal push while taking steeper turns or maneuvering higher uphill and downhill climbs.

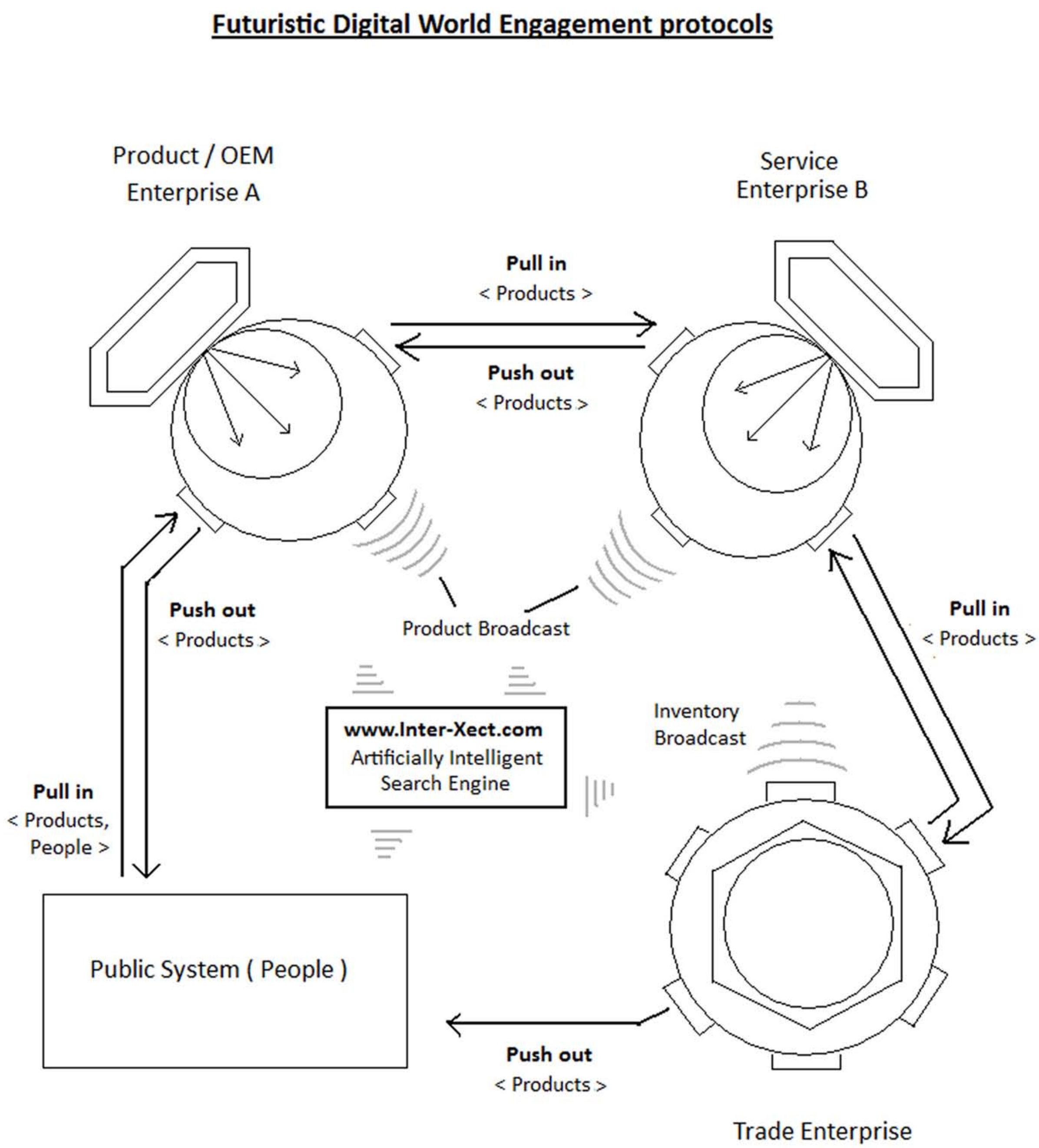
d. Horizontal lateral thrusters in ships would allow easier maneuverability of ships during yaw or turn motions.



e. Applications in braking of trains – Harmonic exponential timed brakes can be used in train wheel brakes while reducing wear and tear of brakes. Either the time parameter could be used or the surface area parameters can be used while exposing the brake pads to the wheel. Timing or surface area Patterns could be like ($x + \frac{x^2}{2!} + \frac{x^3}{3!}$).

f. In computer hardware machinery if a drop in threads or tasks is observed multiple cores of processors could be shutdown so as to reduce power consumption .This is different from the “Always on” cores of processors.

Digital World Engagement - Enterprise to enterprise, Enterprise to public:



The digital world engagement treats each space, the enterprise space and public space as a closed individual system in its own right. Each of these systems operates on three core parameters – People, Products and Processes.

The systems operate through push-out or pull-in functions which are used to transfer people and products across systems using various protocols and processes.

The enterprise systems also broadcast critical structured information to other enterprises or to trade and distribution systems to facilitate transfer of products to the public systems in a direct or indirect manner.

The digital engagement protocols use special artificially intelligent search engines to allow information exchange for efficient engagements between enterprises and public systems.

T&C Vectors is currently developing one such artificially intelligent search engine called "Inter-Xect" (www.Inter-Xect.com), where people from public systems and enterprises intersect in real time to facilitate commerce and engagement transactions.

A pictorial diagram representing the future world space is shown as a diagram in the succeeding page.

This school of thought is the principle foundation or basis of T&C Vectors vision for the future. Such a world system would enable sophisticated enterprise governance and societal growth.

Digital Economic Lifeline systems:

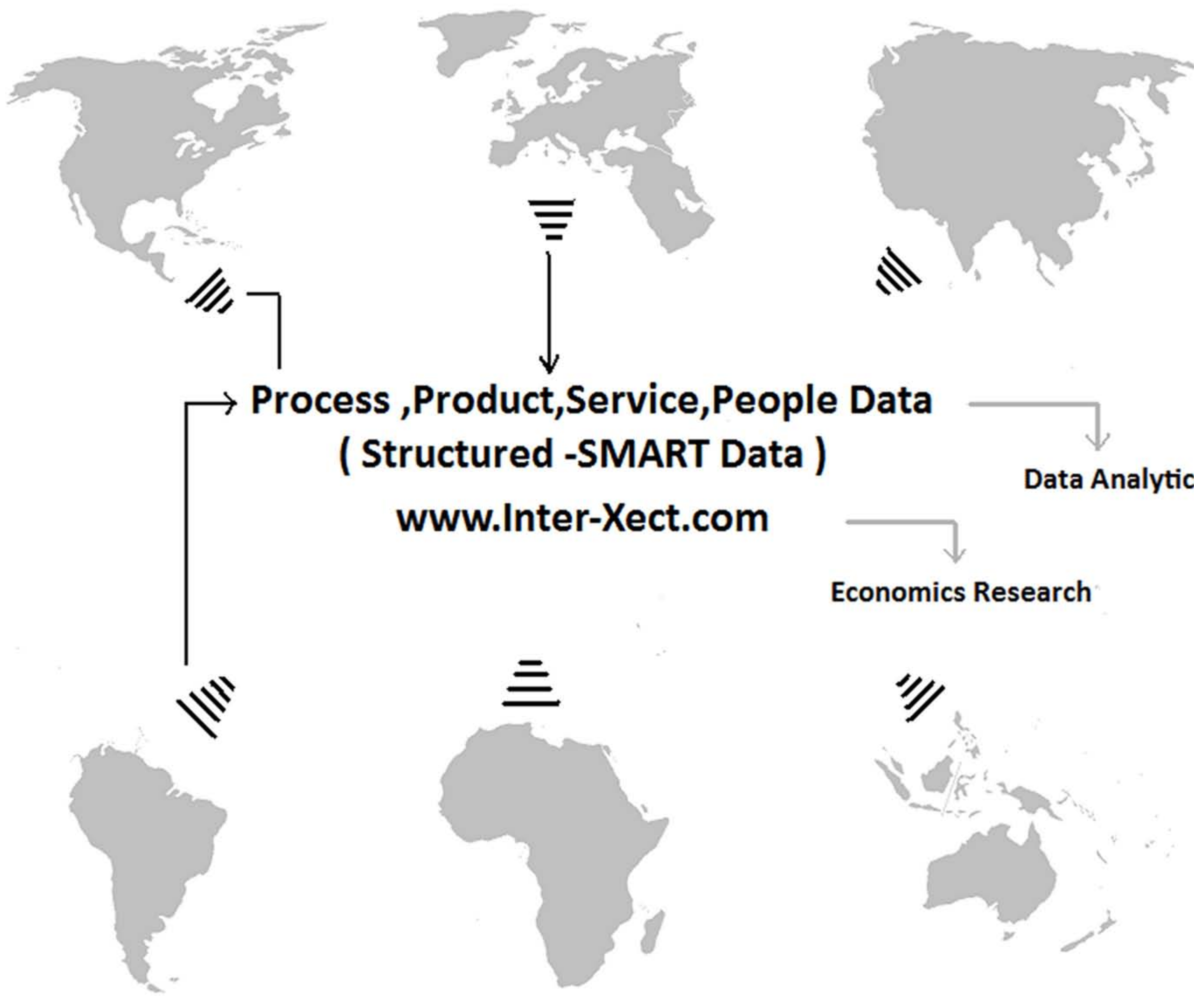
This concept envisages the entire world's economies operating as Digital Economic Lifeline systems, broadcasting SMART – Precise Data that can reveal the true state of a nation's economic system.

The Constructs for the same are not overly complex. It relies on structured data or information classified according to the systems engineering theory described in this handbook.

In fact T&C Vectors Artificially intelligent search and engage engines InterXect (www.Inter-Xect.com) have been designed to provide critical data for the measurement of the state of the economic systems of the entire world, including all countries and continents.

A pictorial Diagram representing a digital world Economic system is described in the succeeding page.

World Digital Economic Lifeline Systems

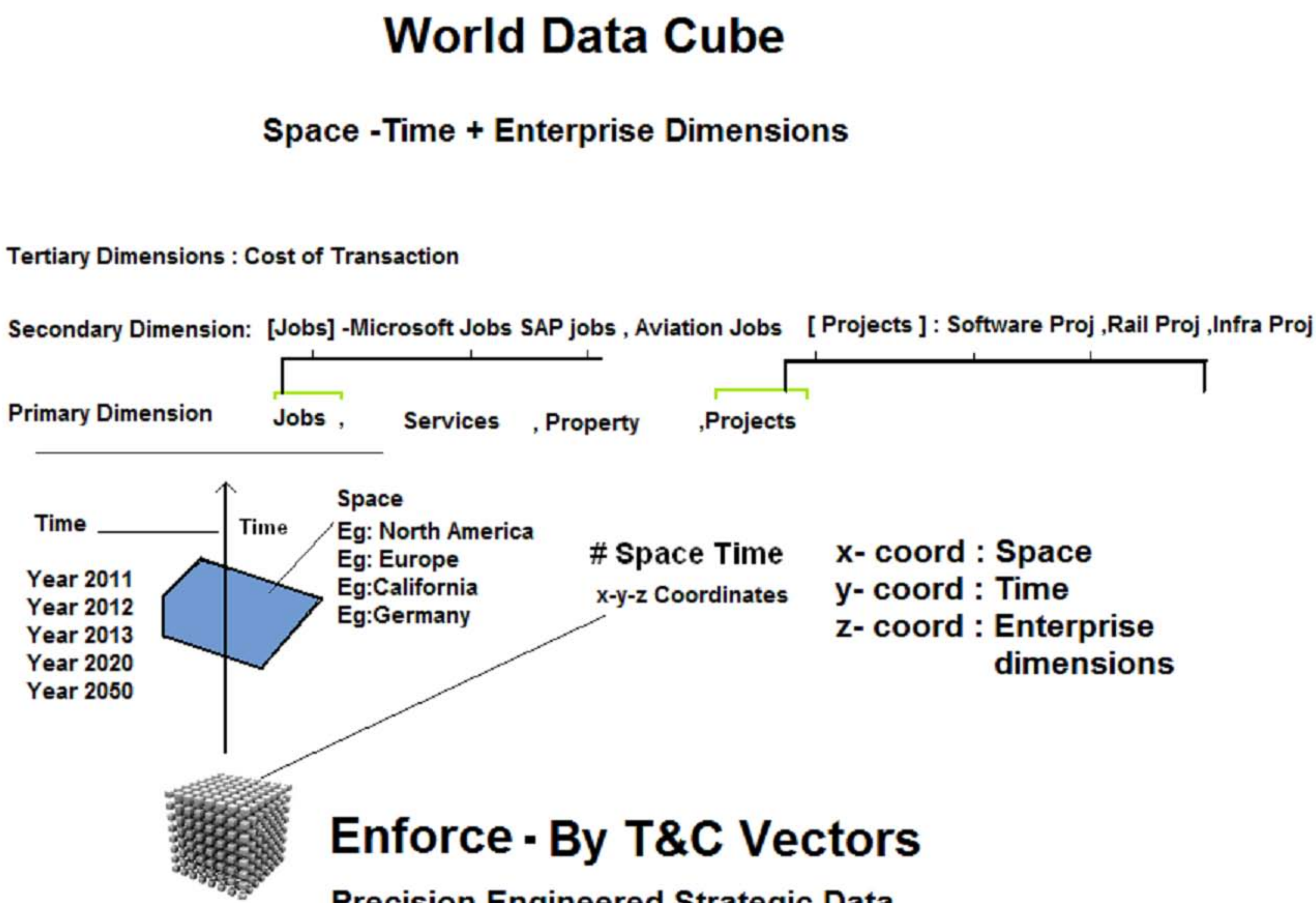


Precision Engineered Strategic Data to Manage the world – Enforce

The Digital Lifeline or the Inter-xect search engine creates usable precision strategic data for the enterprises to run their strategies.The concept is simple.The data or rather the SMART Data obtained from Inter-xect would be processed to create strategic data for the enterprises.

The structured data “Enforce” uses the Physics principles of Space Time Coordinates where the x,y and z coordinates are occupied by the dimensions – Space or geography , Time month+ year and Enterprise Data (like Jobs,Projects,products).

The following Diagram should be able to explain the working of Enforce Data.



Community MutualFunds: This concept Introduces a Brand New Financial Innovation or an Innovative Financial Product called the "Community Mutual Fund".In traditional industry a large number of Financial firms offer a product called the Mutual Fund.The Mutual Funds are funds created by collecting investments from a large number of people.The funds are invested in the stock market to create gains for the investors.
It may well be remembered that these mutual funds escalate the price of stocks without Serving any real investment purpose.It is so because the stock market doesn't have instruments to allow these funds to be used by companies to grow further.These funds only increase the notional value of the stock and introduce new Financial jargons like "Market Capitalization".

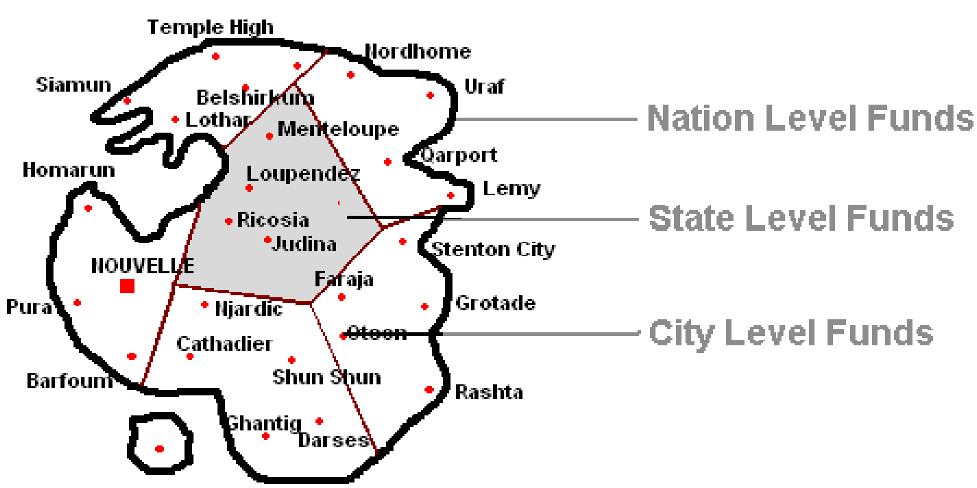
T&C Vectors envisages partnering with Leading Financial Institutions to introduce the concept of theCommunity Mutual Fund.

The Community Mutual funds are funds collected as National Level Funds,State Level Funds and City level Funds.

These Funds would be used to build infrastructure , create industries, create transportation infrastructure , job opportunities etc at the National Level , State Level and at the city level. The wealth created from these investments would be shared by the Investors.

One may remember that, in the StockMarket principle when "one" gains ,some one else loses wealth.This encourages predatory practices to create wealth rather than priciples that create mutual growth and Benefits.

These concept of the Community Mutual Fund serves a real purpose of investment and growth facilitation.The Community Mutual funds can be used in conjunction with funds from Consortiums to build new modern ecocosystems with the responsibilities of governance being shared by the industry and the Community.



Chapter 11

An Introduction to Enterprise Focus & Inter-Xect

This Chapter introduces the two Core products from T&C Vectors –

1. Enterprise Focus – An Enterprise Nervous System

2. Inter-Xect – An intelligent World search & Engage Engine

Enterprise Focus & InterXect have one Unified Purpose, to create a Managed world.ie to manage the Enterprise space as well as the Public space of the entire world systematically, using sophisticated Enterprise Engineering principles.

Enterprise Focus:

The Enterprise Focus Nervous system is a Scalable Nervous system for the Entire Enterprise –connecting People Processes and products seamlessly to create a very fluid and agile environment in the enterprise space.

Just as a nervous system of the Human Body, the Nervous system of the enterprise creates a managed world in enterprise space. This managed world is not built from general management techniques, rather is principled on the Intelligent Design Philosophy.

It Creates the Clearest Macro Pictures and the Micro Pictures of the entire enterprise, its processes and may be classified as a classical breakthrough incorporating the most advanced Programming theory and the most sophisticated Advanced Management Functionality.

It by default includes the modern practices of Design Patterns and Enterprise Architecture, and Advanced Project Management based on the Intelligent Design Philosophy.

It is principled to extract the most amount of useful work from an enterprise system connecting all people of the enterprise to a common purpose and Singular Vision.

Unlike ERP systems that predominantly cater to the processes of the enterprise, Enterprise Focus connects all the three entities of the enterprise <People, Products &

Processes >.It may be noted that Enterprise Focus is a software used in conjunction with ERP systems and does not act as a replacement.
In fact an Enterprise Nervous system is a Brand new concept being introduced, a breakthrough product representing a systems engineering discipline.

Inter-Xect:

InterXect could be the Most Sophisticated and smartest Search and Engagement Engine the world is to see.

Unlike traditional search Engines- which help in searching for information available on the web, Inter-Xect is a World search engine that searches for the Products, Processes, services and people mapping the entire earth (Continents, Countries, States, Cities, and Villages) based on data available through Enterprise Focus or Independent data feeding systems.

It is a Proprietary Concept and its vision extends to Forming a Digital Lifeline for the Entire earth with a Principle foundation of Operation on “Structured Data”.

It connects the enterprise to enterprise space, the Enterprise to Public space, disparate domains and various types of industries.

The search Engine can accurately pinpoint the farmer in an Indian village offering agricultural produce, a large Engineering firm requiring engineering components or a software firm requiring skilled software professionals or, Millions of such queries.

It has been designed to work like One Large Artificially Intelligent Brain that could answer the questions put forward by the world at large. Like any machine, its functioning depends on the user’s inputs of Data in A Proprietary Format.

It can be described as a digital Brain center for the Entire earth.

T&C Vectors is Confident that a system so Deep and Wide in scope has never been constructed on earth before. The core principles of Operation are not based on complex technical’s, but rather on the Intelligent design Theory.

It’s been Named inter-Xect to convey a notion, where the entire world intersects to conduct their Life on earth.

T&C Vectors anticipates that the people from all walks of life would align with this next step in Human Evolution, in creation of a world without borders.

Chapter 12

Intellectual Property Rights

This Entire Set of Concepts discussed in this book is a fruit of 7 years of Labor in Research and systems development (Year 2003 to year 2010).

T&C Vectors claims all rights to the Intellectual matter that's been conveyed through this book.

The intellectual material is copyrighted and ways are being considered to file patents for the same. Since the scope of the Intellectual property is very wide it would lead to a very large no's of patent claims. To address this issue it's been tentatively decided to Designate all intellectual property as –T Enterprise Freescaling smart Algorithms.

T&C Vectors encourages Enterprises and people to effectively align with T&C Vectors for acquiring rights to building a smarter world.

T&C Vectors anticipates it would not come across cases of, purposeful intellectual Infringement.

All intellectual material may be used for self-development purposes within the enterprise space or the private space only.

Any application of the Intellectual property in development of a product for commercialization would require garnering the rights for the same, from T&C Vectors.

Exercise space/ Notes

Exercise space/ Notes

Exercise space/ Notes

Exercise space/ Notes

Exercise space/ Notes

Exercise space/ Notes

Exercise space/ Notes

Sophisticated , Simple and Absolutely to the Point

This Sophisticated Book Introduces you to breakthrough concepts of T&C Vectors principles of Intelligent Design & Systems Engineering .

Audience

- CEO's
- Top Management Executives
- Innovation Leads
- Solution Architects
- Technical Architects
- Project Managers
- Operations Executives
- Enterprise Architects
- Research Scientists
- Mfg/Production Executives
- Marketing Professionals
- Financial Engineers
- Designers
- SW/HW Engineers
- Programers
- Students

It Simplifies this complex world , Deals with Engineering and Architecture purely from a Systems Perspective & sums up decades of theoretical learnings into simple concise pellets of Knowledge .