Riskcare

The Objectively-Oriented Organisation

INTRODUCTION

1.1 THE IMPORTANCE OF STRUCTURE

How we structure an organisation is one of the most significant factors underpinning its prosperity. This matters as much as its business model, strategy and execution. A well-designed structure amplifies these; a badly designed structure works against the leadership.

1.2. IS THERE SUCH A THING AS A 'BEST' WAY?

Is there a 'right way' of designing organisations? If there is then it follows that there must also be a 'wrong way' of designing organisations. Can we measure how well our organisations are designed along the spectrum of 100% wrong through to 100% perfect?

1.3 POWER POLITICS STEM FROM THE LACK OF LEGITIMATE DESIGN

The fact that 'organisational design' is not a widely-recognised discipline means that designing a firm's structure falls to the most influential people rather than those best equipped to perform the task.

Worse, the lack of objectivity often personalises dialogues – a decision over the shape of a company boils down to one person's view against another's, thus creating conflict.

If the design of organisations can be handed over to professionals or outsourced to specialists then companies can focus on advancing their business without the paralysing distraction of individuals fighting for their destiny. Separating design from execution would be a welcome deathblow to corporate politicking and good design could release unimaginable human potential.

1.4 IS THERE A LANGUAGE WE CAN USE TO DESCRIBE ORGANISATIONS?

When we restructure our firms, the lack of a vocabulary to describe different structures and their qualities inhibits the process.

Management innovation is the most wealth-generating form of entrepreneurship, yet our ability to imagine is constrained by the words and concepts that we are familiar with: matrix, hierarchy, policy and so on. What if there are management patterns and tools out there that are better than those with which we are familiar?

By definition there are because the future hasn't been invented yet and because it is inconceivable that the future will be realised with hierarchies, silos and dotted lines. How do we discover these new patterns – patterns that will release value just by being more enabling and efficient than the structures we currently have?

1.5 MAKING 'MANAGEMENT 2.0' A REALITY

Management 2.0 teaches us servant leadership, ownership, empowerment, peer accountability, freedom of information, democracy. But while everybody knows it when they see it – Google, Gore, Whole Foods – we struggle to discern a set of actions we can apply to our own organisations.

What if we could obtain a service pack to upgrade our management, an upgrade that addresses the question at the heart of the matter: how do you get the right balance of power between the leaders and the newly empowered followers without releasing chaos? A large part of the answer lies in how you design your organisation.

1.6 WHAT PREVENTS CONTINUAL EVOLUTION?

We long for pain-free renewal of our organisations, structuring them so that they continually regenerate themselves without needing crises-invoked rethinks. Is there something in our organisations that prevents natural and healthy mutation? If so, it must in part relate to the way in which they are currently structured. What do we need to change to make progressive evolution a sustainable reality?

‘ Competition has its place in the marketplace, or against last year’s performance – perhaps even against another office or individual where there is no particular interdependence, no need to cooperate. But cooperation in the workplace is as important to free enterprise as competition in the marketplace. The spirit of win/win cannot survive in an environment of competition and contests ’

STEPHEN COVEY
THE SEVEN HABITS OF HIGHLY EFFECTIVE PEOPLE

1.7 AN EVERYDAY EXAMPLE – VERTICAL VERSUS HORIZONTAL

To maximise a firm’s efficiency we want to have common horizontal layers offering shared services. But to maximise a firm’s adaptability we want to have highly focused client-aligned products and channels – vertical silos. This structure destroys accountability: the verticals blame their missed opportunities on the inadequacy of the horizontals, while the horizontals blame competing demands from different verticals.

The vertical channels control revenue and have the power that comes with this, so can threaten to externalise the horizontal functions thus undermining their value and purpose. This matrix structure inherently creates conflict. How do you decide what should be ‘horizontal’ and what should be ‘vertical’? Who is impartial and empowered to make this decision? (Only the C-suite and this is well below their radar).

Perhaps this grid-like design is flawed and the competing objectives of focus and efficiency can be met with other designs that make them compatible objectives, and perhaps neutral observers can perform the design process?

1.8 STRUCTURE INFLUENCES CULTURE

We know the physical form of our environment influences our community culture and persona. We see that open plan offices with communal areas and towns with shared amenities encourage interaction, and this changes our social comfort zone. Openness promotes community – we are more ready to collaborate with others. Might changes to our functional landscape also influence our group psyche in a positive way? If we create a more constructive working environment might we enjoy work more and might this improve our creativity and productivity?

What if there is a perfect way to design an organisation? What if this can be decoupled from the actual running of the business? What if this allows Management 2.0 to be realised in practice, releasing human capabilities from the bondage of inhumane structures and power politics to be closer to our natural human selves?

OBJECT-ORIENTED ORGANISATIONAL DESIGN

2.1 ORIGINS OF THE OBJECTED-ORIENTED MOVEMENT

The problems in our corporations bear many resemblances to those that faced the software industry a generation ago. Monolithic systems with millions of actors… a learning curve measured in years… one small change breaking the entire machine… unmanageable change projects… top-down requirements-driven implementation… This was the norm prior to the invention of Object-Oriented systems development – ‘OO’ in software parlance, pronounced ‘oh, oh’.

OO provides a framework for designing large-scale systems that makes the implementation process more manageable, in part because it makes a clear separation between design and realisation.

There are methodologies and guidelines around how you design and there is a definitive optimal design – a 'best' design for a given purpose. As an OO programmer you can navigate your way around the innards more readily, making it an easier and more productive environment in which to work. You can make changes to small parts of it with a clear understanding of which other parts are affected, enabling continual evolution of the system.

OO changes neither what the software does nor how it does it: it changes the way its contributors work. Counter-intuitively it is not a technology solution but is a management tool for organising labour. It provides a framework and a vocabulary for software designers, developers and managers to interact and communicate.

2.2 RE-PURPOSING OBJECTED-ORIENTED SYSTEMS DESIGN TO STRUCTURING ORGANISATIONS

At one level systems and companies are much the same – they take inputs, they apply processes, they require energy and capital and produce outputs. For a system, the inputs and the outputs are binary data – at some point this data interacts with the real world in a meaningful way by driving an interface – a network, hard disk, mouse, robotic arm, monitor etc.

Likewise companies take in raw materials, they combine multiple inputs and they process them. They apply their expertise, energy and capital to produce value-added goods and services and distribute them through various channels. As software code is to data and resources, so people and processes are to information and physical resources.

Since the practice of good software design is widely understood and well developed then perhaps it can teach us about organisational design – let's take a look.

2.3 OBJECTIVE-ORIENTATION – THE CLASS AS THE BASIC ELEMENT OF DESIGN

The basic unit of OO design is a class. Once a class is designed it then gets implemented by a programmer and when the program is run, one or more objects can be created in memory that take the form of that class – they contain the same data structures and offer the same functionality.

'So often the problem is in the system, not in the people. If you put good people in bad systems, you get bad results'

STEPHEN COVEY
THE SEVEN HABITS OF HIGHLY EFFECTIVE PEOPLE

As I type on my word processor it doubtless has a class inside it called 'document' that was designed and implemented by the vendor, and as I am currently working on eight documents there will be eight document objects in the memory of my computer. While each of them is different in size and content they all offer the same functions. If I choose 'save' from the menu then my word processor application tells this particular document object, this paper, to 'save' which in turn calls classes within the operating system, which write the contents to disk.

In organisational terms a 'class' can be thought of as the set of responsibilities that a group of people provide to the rest of the organisation – the services they offer and functions they perform. When a class becomes a reality, for example, an airline service desk class becomes the JFK Customer Service Desk or the LHR Customer Service Desk, then this is an 'object'.

The first key difference that OO introduces is a clear separation of design from implementation – as this example shows design occurs prior to implementation, and implementation (programming / designing an operating model) occurs prior to usage or rollout. Before OO, design and implementation were integrated; OO decoupled them. This teaches us that design is a separate activity to implementation: you design your operating model and then you roll it out to teams and people ('business process re-engineering', training). This separation depoliticises the debate; those people affected inform and influence the design but they are not in control of it. Design is performed in the context of the whole system rather than fitting a local perspective.

The second key difference that OO introduced was that the concept of a 'class' combined both data and functions – they are inseparable.

In earlier programming languages the data was held independently of the code; there were no restrictions on changing data used by other parts of a system. This necessitated that all code behaved consistently.

Let's imagine a human organisation constructed in the style of a traditional program. If people have access to shared resources, there is inefficiency if the resources are tightly controlled or large-scale chaos if not. Changes can only be tested in situ, making it difficult to predict second- and third-order impacts elsewhere in the organisation. We can still create good organisations this way, in the same way that even today we can still create good non-OO systems, but they are harder to manage and success is less assured. This complexity is a limiting factor on the art of the possible. These symptoms reflect our real-world experiences.

By collating functions into a class and giving it its own data, OO enacted and made tangible the concept of ownership – which in OO parlance is known as containment. OO gives us a broadly applicable set of guidelines as to what ownership / containment means in practice.

2.4 PRINCIPLE 1 – CONTAINMENT

The concept of a 'class' is generalist – there are rules but they are few. Ownership as enacted by containment means:

RESPONSIBILITY – every class has a name that suggests its purpose and declares the functions that it offers to the rest of the system, so every class has visibility of the purpose and role of every other class. There is no hiding – the whole system sees the whole system. The set of functions a class offers is called its interface.

EMPOWERMENT – a class combines its functions with the resources and data it needs to enact them – it has authority commensurate with its responsibilities such that it is self-sufficient. Any imbalance is unhealthy – hogging more resources than required will impair another class that needs them and means the squandering class is at best inefficient or, worst, unaccountable. Vice-versa, a class with inadequate resources to service its functions means it has responsibility without the means to achieve it and will fail to deliver; people are over-committed and under-resourced. The right balance is where every function has all the resources it needs and where every resource is useful and necessary – there are no redundant resources. This is a basic test of good organisation design: an OO design that does not comply with this cannot be built.

If one class invokes a second class and does not receive a response, then that second class has failed its commitment to the system – it dropped the ball. If one class is waiting for another class then that period of waiting affects the behaviour of the whole – the behaviour of one determines the behaviour of the whole

COMMITMENT – most classes depend on other classes and the system as a whole only works if every class performs properly – classes commit to the rest of the system. If one class invokes a second class and does not receive a response, then that second class has failed its commitment to the system – it dropped the ball. If one class is waiting for another class then that period of waiting affects the behaviour of the whole – the behaviour of one determines the behaviour of the whole.

If an exception (error) occurs in one class, provided that the exception is built in at the design stage it is the responsibility of the receiving class to handle that exception – it is a planned for and legitimate scenario. Otherwise, if an unplanned error occurs it is the fault of the class where it occurred. OO teaches us how individual behaviour influences corporate behaviour and how to diagnose culprits – how to debug the system.

CITIZENSHIP – each class is responsible for creating objects, for acquiring the resources they need and for tidying up once they have served their purpose. If you initiate something you are responsible for completing it and clearing up any waste – you are looking after the longer-term interests of the whole.

Every class has a constructor and a destructor – mandatory functions that are responsible for the preservation of a healthy ecosystem. Classes can only access resources via the class that embodies the resource – in an OO system it is not possible to bypass this. If you want to borrow a book from the library you have to ask the 'librarian class' and they will service you and manage their inventory; you cannot just take the book. Good citizenship is structurally embodied in OO.

RECOGNITION – those parts of the system that are used a lot and those that are redundant are readily apparent within an OO framework. There are no unknowns, nothing is unused or superfluous – all contribute to the whole. For humans to be properly motivated we need to know why we do what we do, we need to know that we are valued, we need to own what we do from inception through to completion and be credited by our peers.

If your organisation design is compliant with OO design principles then the above interpretation of ownership will be realised as a consequence. Ownership is enabled and enforced via structure.

While 'containment' is the primary rule for OO systems, on its own it is insufficient and requires two further rules to complete the rubric of ownership.

2.5 PRINCIPLE 2 – INFORMATION SHARING AND HIDING

Information is power and is often subverted for the abuse of power. Controlling access to valuable resources and data is a complex subject, but classes offer a simple and versatily solution by binding functions to the data they access. There are three levels of 'information visibility' within an OO system:

- Private data can be accessed only by the class incorporating it – all data is 'private' by default.
- Friend data can be accessed by other classes that are identified by the incorporating class as 'friends'.
- Public data can be accessed by any part of the system.

Within an organisation, any person / team / function needs control over its resources but there are times when they may want to share these resources with other trusted parts of the organisation.

OO makes trust formal and explicit. For example, senior levels of government will want to classify the sensitivity of data – secret data will be visible to their 'friends' but invisible to other parts of the organisation. Detailed P&L data may be shared between finance and the people to whom it pertains – these groups are friends.

OO provides a simple model for protecting and sharing data, one that realises the concept of ownership and provides for all situations, building trust throughout the company. Everybody knows the framework for managing privacy, but participants only see the content that the owners see fit to share with them.

2.6 PRINCIPLE 3 – INHERITANCE

Software and organisations are expensive to create so re-usability is key to achieving economies of scale. The concept of inheritance provides for this in two ways:

- Often you have classes that are outwardly identical but internally different – if you are designing an organisation, you are likely to want to have the same functions in each geographical location so that you have the same customer experience everywhere, supporting the same range of products and services. Although the outward representation of each class of your organisation is the same, the realisation is location-specific. If there are functions in the parent that are common (say, a CRM workflow) they can be inherited (replicated) without having to be recreated from scratch. The standard functionality of the base class can be overloaded with non-standard functionality in a derived class. Technologists call this polymorphism.

- A new class can be created from multiple parent classes to provide a broader range of functionality – this is called multiple-inheritance. If you normally keep your sales and billing services separate but for your high-end clients you wish to offer an integrated service, then a 'premier' class is easy to create by just inheriting the existing sales and billing classes. But this technique needs to be used with discretion since overuse rapidly leads to complexity.

Design reuse in an organisation – repeating operating models and workflows and being able to manage variations – reduces the magnitude of the challenge. It is easy to use OO designs to compare different organisations and identify opportunities for rationalisation but inheritance goes beyond purely defensive strategies: it gives you a tool set to explore potential synergies and to mix and match existing competencies and capabilities into new offerings – it facilitates management innovation.

Together with techniques in the design phase (see page 8) it provides guidance around whether a centralised or distributed model is to be preferred, and a vocabulary and rubric for socialising this.

While re-usability of designs may seem unimportant (designs are easy to copy), operational efficiency has a significant impact on the bottom line on an enterprise-wide scale. Reusable standardised operating models are powerful antidotes to unnecessary, undetectable and unchangeable costs.

2.7 OBJECTED-ORIENTED PRINCIPLES IN PRACTICE

The class as a concept is different to the organisational design elements with which we are familiar, precisely because, in and of itself, it has no structure. Instead, the concept of a class offers a set of rules that all organisational structures conform to.

It is these rules that enact the principles of Management 2.0. These are a simple set of concise rules – light but sufficiently robust to ensure good governance within a large-scale system. By re-purposing the OO tools and techniques to organisational design we can reasonably expect to get the same benefits:

- A focus on business rather than organisational issues.
- Better execution since clear rules around purpose, ownership, responsibility and authority are built in at design time.
- Management innovation, since novel combinations can be identified at a design level.
- Better efficiency, owing to better design and reuse of design.

These are the benefits derived from the OO concepts and principles in design; there are more to come when we consider its realisation, as we shall see.

2.8 THE OBJECTIVELY-ORIENTED ORGANISATION PROPOSITION

We have identified similarities in the challenges affecting large-scale systems and organisations, and in the patterns and designs of their structures. We have noted that many of the advantages of OO design, particularly in the use of classes, have their parallels in many of the recommendations of prevailing management thinking.

So the proposition is this: (i) that we separate the design of organisations from the implementation, and (ii) that we think of them in terms of classes that expose an interface to their organisation (their responsibilities and functions) and have commensurate authority over the resources required to service their interface, and that we stop thinking in terms of existing patterns – hierarchies, matrices, silos.

This subtle but profound shift was difficult for software developers to adapt to. For persons steeped in traditional procedural thinking the OO concepts and their justification were hard to grasp and any benefits seemed unnecessary. Systems on the whole worked well. After all, OO is a management technique; strong performers didn't need it, weaker ones didn't get it. At one level OO is just a simple way to reorganise the internals of a system. But the payoff was real –

a new way of working enabled a new way of thinking. Technologists today cannot imagine a world without OO.

2.9 THE PRACTICE AND METHODOLOGY OF OBJECTED-ORIENTED DESIGN

How then, can we make this real for our organisations? How do we a change an expansive, rigid, complex structure into something more adaptable, configurable and fluid, something more human? How can we reduce entropy to make firms simpler to navigate, learn and govern?

We continue to learn from the OO movement. The methodologies developed to migrate systems from the traditional mould to the OO fold are instructive. You scope your problem – the whole system / organisation, or a part that you have responsibility for, you decompose this into the foundation processes to eradicate any preconceptions you may have, you aggregate these back together such that processes using common resources (data) are pulled together in classes and then you implement these classes. Implement means 'program' if we're talking systems, or 'define responsibilities' if we're talking humans. Once done you can test and run your program or business process re-engineer your organisation.

There are three discrete steps to the OO methodology:

2.9.1 ANALYSE

The purpose of the analysis phase is to determine a conceptual model rather than an actual specific design – this comes later. Processes are collated according to their task or problem domain with the goal of minimising the complexity of the overall system.

The optimal conceptual design is the one that has the minimum number of classes that facilitate the full capability of the organisation and where every resource (data element) is exclusively owned by a class.

This analysis is conceptual in that it considers the function of the organisation in isolation of logistical factors such as environmental constraints or geographic distribution, and behavioural factors such as throughput and response time. In geek-speak we are designing the logical model before we design the physical model.

This is a well-trodden path in systems analysis based upon a process called normalisation – the universe of all known items is compiled, then collated into similar items and then rationalised into the simplest possible structure.

For analysing and normalising an organisation we can use the tools of core competencies and core capabilities to assess key domains. Both of these enable a firm to determine its primary competitive advantages – those aspects of the firm that generate value and that are difficult for competitors to replicate. A core competency depends on knowledge and skills to create a barrier to entry – competitors will struggle or be unable to acquire the expertise and sometimes this can be protected as intellectual property.

A core capability relies upon the level of investment it takes to assemble a differentiating process – this can relate to distribution, human relationships in sales, infrastructure. A core capability is easy to replicate in a laboratory but prohibitively expensive and time-consuming to scale. Core competencies and capabilities are both differentiating and value generating but they are modelled differently.

While a traditional functional decomposition of a company will help explain the incumbent processes, by analytically assessing the core competencies and capabilities of your organisation a different perspective may be obtained that will challenge the existing functional framework and processes. This is not a cosmetic exercise – a proper understanding of these two techniques guides investment strategy and in order to be effective in executing these strategies it helps if the organisation is structured appropriately.

With a conceptual map of your firm developed you have a tool that enables your firm to understand its generic structure and this can then be mapped into the physical boundaries the firm operates within

A core competency will be stronger in resources and information and is likely to have simpler processes whereas a core capability is likely to have more sophisticated processes that work with simpler resources and data.

This is visually represented in class diagrams – a small number of sophisticated classes for a competency and a large number of simple classes for a capability. If you are investing into these then you want to develop the sophistication of individual classes for a competency and the efficiency and scale of the overall structure for a capability.

Core competency and capability analyses will identify the cornerstones of an organisation's structure and these can then be built upon with two systems-analyses techniques:

- Use case – sequences of external events that act upon your organisation are considered with a view to simplifying and improving customer experience. By analysing your firm in this way you determine the conceptual model that gives the best client experience spanning all your products and services.
- System sequence diagrams – for each use case you identify the chains of processes invoked within your organisation as they traverse the classes that enact them. Understanding this improves the efficiency of your organisation by reducing complexity and costs (it's easy to identify redundant elements or overly complex paths) as well as helping you understand and control your customer experience. When used as a communication tool, these diagrams help staff understand their purpose since they can readily see how their role contributes to the whole organisation.

The output from this analysis is a visualisation of your foundation business model – a pipeline showing your inputs, how you add value and how you distribute this value to your customers.

2.9.2 DESIGN

With a conceptual map of your firm developed you have a tool that enables your firm to understand its generic structure and this can then be mapped into the physical boundaries the firm operates within. Not all of your business units will be the same – the level of sophistication will vary, but the process of realising a concrete operating model from your conceptual model helps both designers and implementers understand the rationale for the optimisations and compromises they will make.

We continue to borrow principles from the world of OO. When it comes to design there are two 'dos' and one 'don't':

- Keep your building blocks simple (a.k.a. the composite reuse principle) – you are better off collating objects together to form a new object than designing a new and more complex class with broader functionality. This means you keep your conceptual model simple and address exceptional situations in the design stage. A simple principle and one that has a significant impact if applied consistently on a large-scale system: the common challenge this confronts is that designers can easily use too much inheritance, resulting in a complex and inefficient design. A simple, practical example: if you provide a service for building 'apps', rather than an operating model that brings graphic design, user experience, software development and content together in all circumstances, for each project you create a team out of the right combination of the four competencies. Obvious common sense, yes, but this OO principle defines and mandates elegant organisational design.
- Build process dependencies into your design (a.k.a. dependency injection) – for most steady state businesses dependencies be-tween classes are part of the design.

But for businesses with rapidly repeated organisation changes, such as professional services and events management, if one object is dependent upon another then the latter is expressed as a precursor to the dependent one at its instantiation. Translated into an organisational context, if you are creating a team that is dependent upon, say, a venue, then whoever creates that team is unable to do so until they have secured the venue.

- And don't build cycles into your design (a.k.a the acyclic dependencies principle) – since classes call other classes, and they in turn depend upon other classes, the designer needs to make sure that there is no possibility of either a deadlock or a feedback loop occurring. A deadlock can manifest itself as an operational impasse, for example, a budget allocation process where a project cannot be authorised without a budget, but a budget cannot be provided without a solid costing for the project. A well-known feedback loop was built into the operating model of the Lloyds insurance market where syndicates were able to re-insure each other repeatedly making it unclear who was wearing the risk and introducing moral hazard when settling claims. Both of these scenarios can be identified and dealt with during the design phase.

Once you have a well-designed OO representation of your firm, one that complies with the above rules, by definition it is the structure that best reflects the natural shape of your firm. It will be a platform that enables you to realise current management thinking and one that reflects the physical realities and practicality of your operating environment. The final stage is to roll it out.

2.9.3 IMPLEMENTATION

For a system the realisation process takes the form of building and testing. For an organisation it is translating the operating model into real people, their roles and responsibilities, training and all other things 'business progress re-engineering'.

You know your existing operating model, the OO analysis and design methodology outlined above will deliver a more efficient and effective target operating model, and an impact study will inform you of your program of change. If your new design is 'good' then the transition will involve a range of changes, some of which bring immediate benefit (efficacy, efficiency) and some of which take time – the human factors resulting from clear mandates and structurally supported ownership – motivation, innovation, renewal.

Humans prefer to discover their role for themselves rather than by instruction. The class they belong to has a name that advertises their role and purpose within the organisation and has a set of responsibilities that they can determine how best to fulfil. By handing over responsibilities of a clearly delineated domain to those executing it they can innovate and improve on their way of realising it without disrupting others. Those responsible for the design can liaise with those responsible for the execution on an on-going basis – this continual review process enables incremental renewal of the overall firm.

2.10 HOW OO DESIGNS DIFFER FROM TRADITIONAL ORGANISATIONS

Common management mistakes – committees that are meant to innovate, managers without the means, competing agendas between separate but similar groups, mismatched expectations between the corporate leaders and the functional leaders, intricately interdependent functions that cannot enact change, matrices that institutionalise conflict, dotted lines that solidify ambiguity – all of these are violations of Object-Oriented design principles. If CAD tools were used to design an organisation a significant number of common structural mistakes are avoided because the design software does not allow them and because human error in the design process is more readily and visually apparent.

Instead of a traditional design that generally has a top and a bottom, an OO design is more cellular – there is likely be a centre rather than a top. It is more likely to be three-dimensional in the sense that different objects will interact on multiple levels and in different contexts. There is no such concept as a 'dotted line' – there is no vagary in a relationship since every class-to-class interaction serves a specific function. The concept of 'reporting to' is binned – everybody is purpose-driven, provides a service to the organisation and is led by the owner of the class they are in.

Shared services, rather than a cat's cradle of ties between horizontal and vertical functions, provide a clear interface to the rest of the organisation and may depend on and interface with many other classes. Profit centres are modelled either as core competencies or core capabilities with a production line of sequential classes that facilitate the creation of value – the value chain is visualised in the design. Client interactions with the organisation can be readily traced through classes of any kind, thus ensuring enterprise-wide efficiency.

Instead of more rigid designs – ones that are expected to last until the next bout of crises-invoked restructuring – the organisational design dialogue can be better socialised, involving many persons in an organisation. Like painting the Forth Bridge, the firm can be continually renewed. Instead of structures being abused to further personal ambitions the debate is depersonalised to enable a dialogue around what is best for the good of the whole.

2.11 THE END RESULT

As practitioners of OO became more confident of this management technology, the scale of problems undertaken by software engineers increased dramatically. Rather than being confined to a single machine, OO systems were created on an enterprise-wide basis. Distributed computing emerged – a technology infrastructure unimaginable and impossible to create using traditional procedural techniques. Each machine offers a range of interfaces – the same concept as a class but rather than being internal to an application they are external system-to-system interfaces, known as services.

This approach has become known as a service-oriented architecture – discrete servers performing functions that can be easily configured at an enterprise level and monitored for good behaviour (utility, latency, capacity….).

OO principles transformed the behaviour and feel of software and liberated the productivity of developers. Likewise, the application of OO principles to organisations will enact a cultural transformation from top-down control to community-based, purpose-driven service. In the same way that OO enabled feats of software engineering that were previously inconceivable, so a service-oriented organisation can benefit on an industrial scale – easy to configure, easy to repair, easy to scale.

3 FURTHER CONSIDERATIONS

3.1 LIVING WITH AN OO ORGANISATION

The system sequence diagrams used in the analysis can be built into the workflow of an organisation. Deploying these in a business-as-usual context allows you to monitor actual process flow through your company – technologists refer to this as dynamic profiling. It is an accurate way to determine the costs associated with each product or service, enabling actual P&L for each client / product / service / region or any other factor.

These workflows can also identify and quantify operational risks and can provide the information backbone that will enable a Six Sigma methodology to be integrated into your organisation.

3.2 BEYOND THE BASICS – PATTERNS IN ORGANISATIONAL DESIGN

The analyse > design > implement process described above is foundational to all OO systems. As the practice and experience in working this way evolved, designers noted recurring abstract patterns in differing contexts and this has given rise to the toolkit and vocabulary of 'design patterns'.

A basic one is called a 'singleton', which guarantees that for a particular class there can be one and only one instance of it. This is common in an organisational context – you only want one master balance sheet, you only want one set of values. Another pattern is the 'model-view controller', used for pushing data to a GUI and capable of updating multiple windows concurrently: this is a model for information dissemination.

While the efficiency and re-usability of design patterns is widely debated it is clear that they provide a language for describing complex concepts and solutions that are otherwise hard to articulate. This makes the problem itself easier to understand and enables a dialogue between multiple participants within an organisation.

There is no reason why we cannot develop design templates for organisations of general types. If we have recognised standards that purport to work well then an organisation that is doing something different needs to know why.

Are they missing a trick and simply by reorganising they will release more potential from their company? Or are they ahead of the curve and have institutionalised a structural competitive differentiator that the rest of the market has yet to replicate?

3.3 CONCEPTS OF MANAGEMENT 2.0 AND THEIR REALISATION IN AN OO FRAMEWORK

Prevailing management thinking provides clear principles for running next generation businesses.

Let's explore how OO can realise these ideals.

3.3.1 DECENTRALISE PROFIT CENTRES, GLOBALISE SHARED SERVICES

By using the 'core competency' and 'core capability' analysis, together with the OO tools for analysing and designing your organisation, the conceptual model developed will centralise and standardise the classes that comprise the firm. By definition there is only one of everything. But, as the OO methodology progresses into the design phase, explicit choices need to be made around the optimal physical realisation of the business – the design decisions will specifically address where to decentralise and where to globalise. OO does not enforce this principle, but it makes its consideration explicit.

3.3.2 BUILD COMMUNITY

In an OO world, hierarchy still has a role but it is diminished. It is useful for allocation of resources and navigation rather than for asserting control. Communities of similar roles are grouped together in classes: they may assemble hierarchies where appropriate, for example, to facilitate distribution of labour or increased specialisation. But such hierarchies operate within the common purpose of the parent class so competition is on a level playing field. Hierarchies exist in nature but they are not the only pattern and they need to be used carefully so as not to destroy community.

In a traditional procedural system it is hard to determine what certain parts of a system do, how often, and how this fits into the bigger picture. In an OO world, the utilisation of every member of every class can be observed and their economic contribution can be calculated and monitored

In an OO organisation it is easier to determine the economic contribution made by an object. Every class is responsible for receiving inputs and providing outputs and uses energy and capital to perform its functions – the value added by the process is as visible as the cost of executing it.

In a traditional procedural system it is hard to determine what certain parts of a system do, how often, and how this fits into the bigger picture. In an OO world, the utilisation of every member of every class can be observed and their economic contribution can be calculated and monitored – people are recognised and remunerated for their contribution rather than their position.

Combined with transparency of information, an OO system enables every participant to answer the question 'who contributes the most?' and so the debate over executive pay can be held, correctly, by the body of the firm rather than by the shareholders.

3.3.3 CREATE A LEVEL PLAYING FIELD TO ENABLE DEMOCRACY AND SELF-DETERMINATION

In an OO framework the decisions made around the structuring of the organisation are separated from decisions around the execution of the business. Since the resources and data within a class are visible to all members of the class, everybody has the same information and everybody can support and challenge the decisions made by their leader.

As resources and data are classified within the design as being public, accessible by friends or private, the use of data in supporting decisions is embedded into the organisations structure at design time rather than being withheld or disclosed as events unfold. Data is a strategic asset rather than a tactical toy.

Stripped of the power afforded from controlling and subverting structure, leadership is simplified to inspiring, guiding and mentoring, so the leaders who prevail are those who have the talent to service the purposes of their organisation rather than those who are best at manipulation.

Since everybody within each class is working from a level playing field, peer-review that scores the contribution of everybody within the team can be readily adopted and used for a range of activities that might otherwise be top down – monitoring quality of service, recruitment, investment, selecting leaders, distributing remuneration and discipline. This is taking 360° reviews beyond self-improvement and using the concept as an active management tool.

3.3.4 SERVANT LEADERSHIP AND SUSTAINABLE PROGRESSIVE RENEWAL

Making leaders accountable to their teams is easy to envision in an OO model. The senior leaders perform a similar role to the operating system on your computer – defining scope and purpose, providing stability, instigating change and facilitating the more human aspects of representation and visioning.

The generic leadership class, inherited and practiced by leaders throughout the organisation, exposes a set of responsibilities that define servant leadership – to own the actions of their followers, to interpret external events and set direction, to define and live out corporate values, to monitor the health of their subjects. Everyone in the organisation knows what they should expect from the senior leadership as well as from his or her own leaders.

Leadership in an OO system is not superior to other classes and it need not be responsible for organisational design – it plays a role as per any other class and is as vital to servicing the organisation as any other class. Without elevated control structures organisations flatten; at the enterprise level recognition comes on the back of contribution, not control. This includes executive pay.

Everybody understands their obligations to the rest of the firm as defined by their class. The thought processes involved in determining how to best fulfil these obligations necessitate considerations for efficacy and efficiency. And as they become more specialised in the execution of their class they will be able to make suggestions to the organisation's designers on additional services they can offer the corporation, or request improvements to other parts of the organisation that they could benefit from. Management innovation is institutionalised.

Sustainable non-traumatic organisational restructuring is the norm – renewal is part of the fabric rather than externally imposed invasive surgery.

4 IN CONCLUSION

4.1 A VIEW FROM LOW ORBIT

A market economy is better than a planned economy because it allows greater freedom and decentralisation. Instead of a rigid hierarchical pattern we see a cellular arrangement of organisations that evolve, mutate, assimilate and expire. They conform to a common set of rules (the law) and they all declare their purpose (for-profit or not-for-profit, public or private).

Yet when we zoom into these organisations many of them exhibit the sort of internal structures associated with a planned economy – top-down control, centralisation, rigidity and hierarchy.

Ideally, as we zoom in we would visualise the same metamorphosis as zooming into a fractal – a fluid set of patterns transitioning continuously and seamlessly into new sets of fluid patterns. Zooming in should be a kaleidoscope experience as much as a telescope experience. The substructures within our organisations need to be as fluid as the

economy and likewise conform to a standard set of simple rules with a public purpose and modus operandi. Otherwise, as our markets mutate, the rigid internal structures will strain and break, like ice in a glacier, creating immense pressure and fissures.

This is what the Objectively-Oriented Organisation allows – a structure capable of mutation to support and drive the evolution of the whole within the context of a free market.

Such structures will enable Management 2.0, make firms more human and bring untold social and economic benefits to all. Firms will not be fully capable until they are fully human; OO shows us how to do it.

This is what the Objectively-Oriented Organisation allows – a structure capable of mutation to support and drive the evolution of the whole within the context of a free market. Such structures will enable Management 2.0, make firms more human and bring untold social and economic benefits to all. Firms will not be fully capable until they are fully human; OO shows us how to do it