Be *agile* from top

# THE MODERN ORGANISATION

to bottom

Sandro Quaranta

# The modern (IT) organisation

## Table of Contents

- Strategic areas
  - Strategic and Operational objectives
- Business Strategy Development
  - Analysing the company and its environment to set strategic direction
  - Analysing the products portfolio and markets to discover opportunities and develop new business
- Strategy implementation
  - Models for managing strategy implementation
  - Innovation / Product Growth Strategic Objectives
  - New Products Design
- Product / Application Lifecycle Management
  - ALM versus SDLC
    - SDLC
  - ICT Product & Service Life Cycle Management
  - Traditional Demand management
- Objectives from Risk perspective
  - Corporate Risk taxonomy
  - Strategic objectives into tactical / value stream / tribe objectives

4. Enterprise Release Management
   - ERM 4X-diagram
     - The Permanent Organisation
     - Changing the permanent organisation
     - ERM Business stream
       - New business
     - ERM Operations stream
     - ERM Application stream
     - ERM 4X-diagram Infrastructure stream
     - ERM compliance stream
     - ERM Application Quality Control Gates
     - ERM 4X-diagram
     - Value Streams and the ERM 4X-diagram
       - Value Streams vs. traditional Demand & Portfolio Management
   - *agile* Enterprise Release Management
     - Integrated Enterprise Release Planning

- Traditional Release & Change Management
- Scaled Agile, DEVOPS and Quality Assurance

5. The traditional and modern Enterprise Portfolio
   - The traditional Enterprise Portfolio
   - The modern Enterprise Portfolio
   - OPS & non-functional requirements
     - Service Level Agreement (SLA)
   - Enterprise Quality Management
     - Quality Management
     - Nowadays ICT QA is about risk profiles
       - The need for risk management
     - QA Standards and frameworks
     - Risk Assessment
     - Risk Mitigation according to NIST SP 800-30
     - Linking Quality Management and Risk Management
     - ISO 27005, ISO 9001, NIST SP 800-30
     - Quality Management in DevOps/Agile from policing to health checks
6. Business and Value Stream objectives - 2nd Lean-Agile level
   - Identify Value Streams
   - Operational Value Streams in the Real-Life Automotive Case
     - Enterprise Change Value stream objective definition
     - New business and Value Streams
7. Epics and Features - 3th & 4th Lean-Agile levels
   - Epics and Features
   - Epics in the real-life automotive case
   - Features and the agile eSDLC
     - Requirements Prioritization
8. The *agile* enterprise SDLC - 5th Lean-Agile level
   - Traditional Agile “SDLC”
   - Continuous delivery
   - eSDLC also covers the product management phase
   - SPECS cycle
     - Requirements engineering
   - DEV cycle
     - Quality Controls
     - Real-life case

- Feature, PBI and work management with TFS Scrum
- Feature Branch Workflow with Gitflow
  - Branching
  - Software Configuration management
- Fully automated Quality Controls
- Fully automated Release Pipeline
- Software delivery, how do the BIG guys do it?
  - Microsoft BING
  - Google
  - Facebook
  - Amazon
  - Spotify
  - Unbounce
- Continuous delivery

9. Outsourcing IT Functions
   - Why choose for outsourcing?
   - Software development outsourcing models
     - From total outsourcing to selective outsourcing
   - Application Delivery, maintenance, support & operations models
     - Contractual options for the different outsourcing models
   - IT Systems multi-criteria outsourcing decision model
   - IT Systems multi-criteria outsourcing decision model
     - Go/No-Go layer: Strategic systems and capability
       - Core systems
       - Outsource other disciplines / services
       - Business capability
       - Functional capability
         - Requirements engineering / Architecture / design
         - Functional testing
       - Technical capability
         - Technical Testing
       - OPS
     - Differentiating layer: factors for outsourcing
       - System components selection criteria for outsourcing

## List of Figures

2.2. The “agilized” Traditional Strategic Planning Pyramid
2.3. Lean-Agile Enterprise portfolio dashboard [https://enterpriseportfolio.visualstudio.com/]
3.1. Sample Case - Automotive Retail Chain: Mission, vision, goals and objectives
3.2. Lean-Agile Enterprise portfolio dashboard - Strategic Objectives view for Business management
3.3. Video - ALM versus SDLC
3.4. Corporate Risk Taxonomy
4.1. Video – ERM 4X-diagram tutorial
4.2. Business in the center of attention
4.3. The permanent organisation
4.4. ICT Operations providing ICT Services
4.5. ERM 4X-diagram Planning & Preparation and Transition phases
4.6. ERM 4X-diagram Business stream
4.7. ERM 4X-diagram Operations stream
4.8. ERM 4X-diagram Application stream
4.9. ERM Infrastructure stream
4.10. ERM 4X-diagram Compliance stream
4.11. ERM 4X-diagram Quality Control framework non application streams
4.12. ERM 4X-diagram Quality Control framework with blue-green deployment environments
4.13. The ERM 4X-diagram
4.14. Traditional Demand Management process versus Scaled Agile Value Stream
4.15. DevOps and Scaling Agile @ Spotify©
4.16. Different data sources providing the ERM information © Plutora
4.17. Consolidated ERM-planning in © Plutora
4.18. ERM-metrics in © Plutora
5.1. ERM 4X-diagram with the Corporate Business Product Portfolio
5.2. ERM 4X-diagram with the ICT OPS Portfolio
5.3. ERM 4X-diagram with the Corporate Change Portfolio
5.4. Business Operational Value Stream and the Enterprise

Change Value Stream in 4X-diagram
5.5. Production based on Five-Year plans in the former Soviet Union:“These five year will be very fruitful”
5.6. The 4X diagram
5.7. Business Operations requirements
5.8. ICT Service Level & Operations requirements
5.9. Functional & Non-Functional application requirements
5.10. Non-Functional infrastructure requirements
5.11. Requirements hierarchy
5.12. ISO 25010 – ICT Solution high-level Quality Requirements categories
5.13. RCMP Corporate Risk Profile Map
5.14. Risk Profile delivery initiative
6.1. “Spaghetti Organisation” structure
6.2. SBU’s and Value Streams in the Automotive Retail Chain
6.3. Value Stream objectives from the real-life automotive case in Azure DevOps Services
6.4. Value Stream Objectives from the real-life automotive case
7.1. Epics from the real-life automotive case in Azure DevOps Services
7.2. Epics in the real-life automotive case
7.3. Features from the real-life automotive case in Azure DevOps Services
8.1. The Application Change Delivery stream covered by the agile SDLC’s
8.2. Traditional Agile SDLC
8.3. Sprints and Release Trains not used anymore by leading tech companies
8.4. The micro Service Development Life Cycle (mSDLC)
8.5. The software development sprint cycle in the mSDLC
8.6. Product Management Cycle (SPECS): product design, specification and prioritisation
8.7. Lean-Agile Enterprise portfolio dashboard - Features view for PO and SPECS team
8.8. PBI’s and task for the feature 4. Advanced Search.
8.9. The agile enterprise SDLC (agile eSDLC)
8.10. The agile enterprise SDLC (agile eSDLC)

8.11. The agile enterprise SDLC (agile eSDLC)
8.12. The agile enterprise SDLC (agile eSDLC)
8.13. The agile enterprise SDLC (agile eSDLC)
8.14. The agile enterprise SDLC (agile eSDLC)
8.15. The agile enterprise SDLC (agile eSDLC)
8.16. Lean-Agile Enterprise portfolio dashboard - Features view for PO and SPECS team
8.17. MS TFS Scrum Processes
8.18. The GIT workflow
8.19. The NFRs / OPS requirements matrix
8.20. The NFRs / OPS requirements matrix
8.21. Todays DTAP street blue green [VIDEO]
8.22. The Container Integration Stage of the Release pipeline
8.23. image
8.24. image
9.1. The ERM matrix Transition Deliverables
9.2. The ERM matrix Transition Deliverables
B.1. The ERM matrix Transition Deliverables
B.2. The ERM matrix Permanent Organisation Deliverables
B.3. The ERM matrix Permanent Organisation Responsibilities
D.1. The NFRs / OPS requirements matrix
E.1. The Enterprise Release Management landscape (4X-diagram)
E.2. The Enterprise Release Management landscape (4X-diagram)
E.3. The Enterprise Release Management landscape (4X-diagram)
E.4. The Enterprise Release Management landscape (4X-diagram)
E.5. The Enterprise Release Management landscape (4X-diagram)
E.6. The Enterprise Release Management landscape (4X-diagram)
E.7. The Enterprise Release Management landscape (4X-diagram)
E.8. The Enterprise Release Management landscape (4X-diagram)

E.9. The Enterprise Release Management landscape (4X-diagram)
E.10. The Enterprise Release Management landscape (4X-diagram)
E.11. The Enterprise Release Management landscape (4X-diagram)
E.12. The Enterprise Release Management landscape (4X-diagram)
E.13. The Enterprise Release Management landscape (4X-diagram)
F.1. IT asset lifecycle management
F.2. IT asset stages

# The modern (IT) organisation

**Sandro Quaranta**

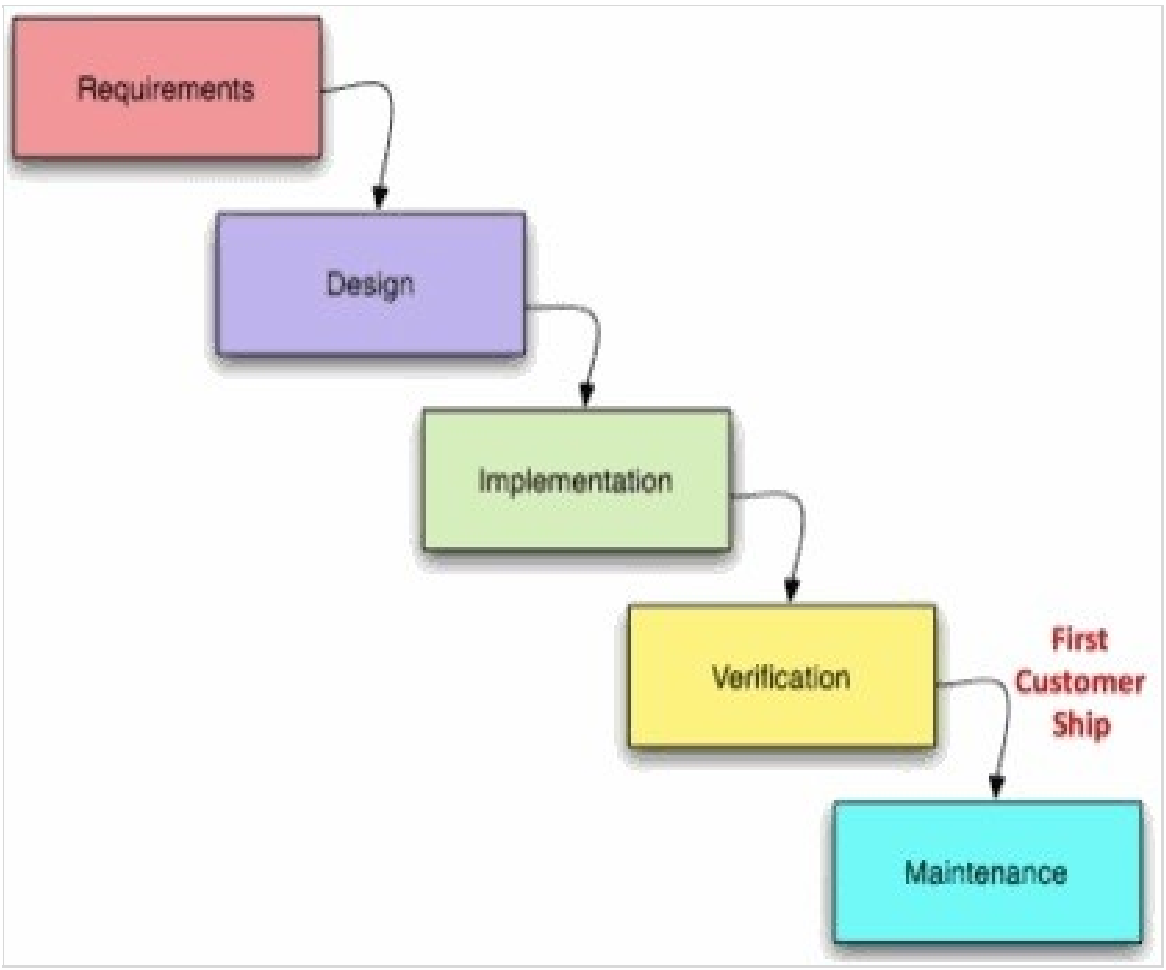
| **Revision History** | | |
|---|---|---|
| Revision 0.8 | 2018-10-02 | SQ |

# Epilogue

Software development methodologies evolution from Waterfall down the Punch Card to DevOps in the Cloud

In the late 1950s the second generation computers, transistor-based and replacing the enormous punch band fed vacuum-tube machines, started the commercial usage of computers. These Mainframes had to be fed with punch cards that took days or weeks to “program” and which had to be entered into the computer in a small available timeslot. CPU usage was extremely expensive, hence off-line programming and extensive preparation of the program via the Waterfall method. The only available bullet had to be on target.

**The Waterfall method**




It was termed "waterfall" because teams complete one stage, fully, before moving on to the next. Requirements must be complete before moving on to functional design, functional design complete before detailed design, and so on through the sequence. And like water not flowing uphill, there are rarely provisions to return to an earlier stage of the process. Once you were finished with a stage, that stage was frozen in time. The Application Crisis Time between a validated business need and an actual application in production was about three years

**Read more** Agile development

The introduction of magnetic storage in the 50’s and terminals in the late 60’s speeded up the software development process, but this was still organised based on the old days punch card practices with long development cycles in

the waterfall model. Solutions for unspecified or missing features (aka bugs) and change requests had to come through the complete Waterfall cycle again. Development of large systems took years and many projects ran over budget and schedule. ICT departments and consultants started a strained search for technology and practices to better control the delivery.

The Waterfall model originates in the manufacturing and construction industries in which after-the–fact changes are prohibitively costly, if not impossible. The structure and design of a delivered office tower or bridge cannot be modified anymore. For punch cards it was more or less the same, only small modifications after creation were possible, hence the Waterfall approach. Electronically stored data however can be copied and modified instantly. On top of that CPU processing power became less and less scarce so deviation from the old practices was technically possible since the 60's.

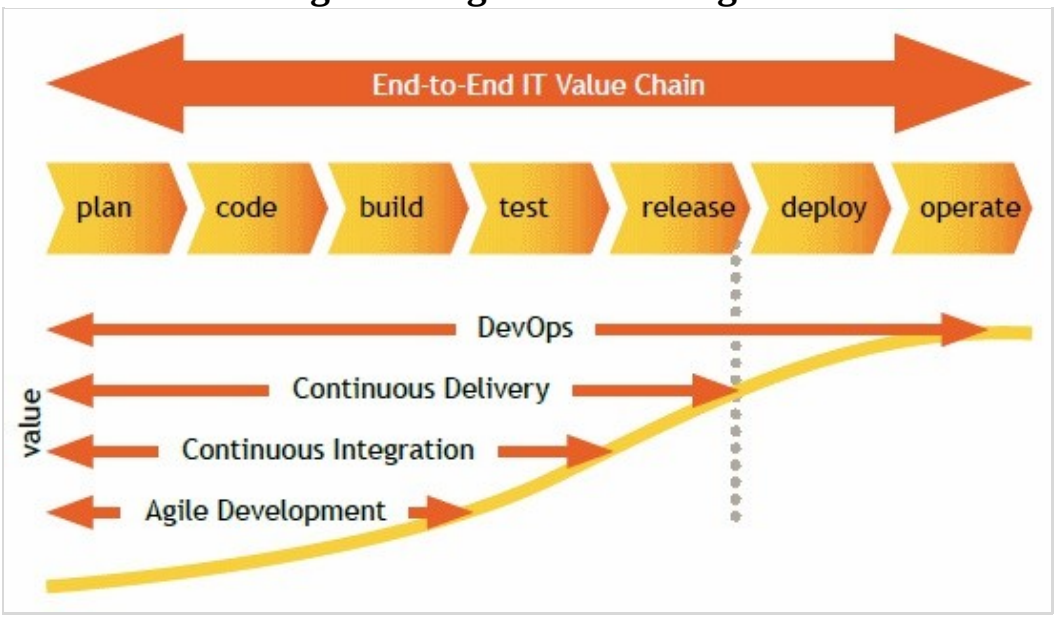
**Figure 1. The Computer evolution**


After the Mainframe Era PC's (with their killer apps), client/server and the internet were introduced, but still Waterfall remained and in many cases still is the preferred way of developing software. Although in theory already introduced before, it was in the 90's during the application development crisis, when rapid development tools together with affordable PC processing power, made agile development practically possible. Agile was the natural

way to go in the mid 90's for Web Applications. Business/users did not know too much about the possibilities provided by the internet technology - many of them had never seen a webpage in their live. The traditional waterfall method was not fit for developing Web Applications. Users cannot describe something they do not even know it exists. The only way was to show the technical possibilities and in narrow cooperation iteratively develop the first Web Applications. Agile software development methodologies like Scrum, DSDM and Xp were introduced.

**Agile methodologies**

**Figure 2. Agile methodologies**




This new Agile way of developing applications revealed another problem: IT development and IT operations are miles apart. A quick and dirty "developed" mock-up running on a laptop sets expectations to users, but is far from PRD ready - "I like the App, can we start using it?" On top of that bringing a system that is development ready into PRD will also require the necessary time and effort. It will not be accepted that easily by OPS to deploy, run and manage. OPS has some valid reasons. They will be the ones called out of their bed for problems on xMas night and will take the first hit from unsatisfied users. In this area of tension developers came up with DEVOPS to bring their results of work with the users faster into production. {DevOps link} But also with the arrival of DevOps we are not there yet.

Isolated agile practices applied by DEV only, do in many cases still not bring the delivery speed needed by the business.

# Introduction

### Enterprise Agility fulfils the Lean-Agile promise

In most larger organisations even the latest Lean-Agile system development and DevOps practices do not bring the total agility and fast results expected by the business. Many existing blocking factors slow down and neutralize the benefits of agile IT system delivery.

The key to establish Enterprise Agility is in transforming the traditional work style of the total Enterprise Change Delivery Organisation, including Strategic Management, into a lean scaled-agile way of working. Agility should not be limited to the Application Development Teams only, but the Lean-Agile spirit must flow through all the veins of the organisation. Without adapting the environment and processes around the Lean-Agile development layers, organisations cannot benefit from true agile business value delivery. The total organisation has to *be Agile from top to bottom*.

### "you or your employers got a SAFe certification, and now?"

The Scaled-Agile Enterprise Frameworks SAFe, DA[D], LeSS and LeadingAgile provide several useful points of departure, but many extra miles need to be made to reach the level of agility that the leading Tech Companies and major "disruptors" have. These frameworks are in general quite theoretical and established by commercial parties with the intention to sell consultancy and services; quite abstract and without providing practical solutions on how to tackle the different aspects of the real Lean-Agile way of working the IT transition should result into.

### This book: The modern (IT) organisation

In this book the modern (IT) organisation is described, agile from top to bottom, using the latest Lean-Agile Enterprise Change Delivery practices successfully applied by the leading Scaled-Agile organisations.

[*(IT) between brackets, the characteristic of modern organisations is the integration of non-IT and IT, both agile and equally important in reaching the business goals.]

Theory and practices will be described to facilitate the IT transition in overcoming several fundamental blocking obstacles for benefiting from the Agile way of working, like traditional cost accounting, governance (financial, progress, ROI) and people management. Techniques will be provided for designing and implementing Value Streams to scale up the agile way-of-working of the systems development teams to the other levels in the organisation.

The vision on the ideal Agile organisation sketches the complete Enterprise Change Process and the whole IT landscape in the context of Lean-Agile and provides concrete methods, tooling and real life examples on how the modern IT Organisation functions and is *agile from top to bottom*.

The description of Total Enterprise Agility starts at the top of the organisational hierarchy where the Corporate Strategy is translated into business and value stream objectives. In the Value Streams product (lifecycle) management results into product & service goals and objectives which are delivered with help of Enterprise Release Management and the latest agile development practices from the agile enterprise SDLC: continuous delivery with scrum workflows and gitflows used by the DevOps teams.

**Audience**

The Scaled-agile methods, tools and practices in this book are useful for any organisation that manages its own IT products and wants speed up enterprise change delivery. All roles involved from business and IT management to software development Also highly regulated large enterprises (HRLE’s) can benefit from scaled-agile practices, compliance mainly affects the lower execution and implementation levels.

# Chapter 1. Total Lean-Agility in traditional organisations

## The Problem of Traditional Project Cost Accounting and Product Management

In modern organisations Lean-Agile development accelerates the delivery of business value. In many traditional organisations the drive for business agility through Lean-Agile development conflicts with current methods of budgeting, project cost management, accounting and product management. Invalidating the business benefits that Lean-Agile development should bring and reducing its contribution to incremental work management only.

At the beginning of this century Lean-Agile methodologies under the umbrella of The Agile Manifesto (Scrum, DSDM, XP etc.) emerged. Especially in start-ups and smaller lean organisations, where small teams maintain a single web site, App or tool, these agile practices function very well. Although the IT world in larger organisations is more complex, development teams happily took over the habits of their unrestricted colleagues from sexy start-ups. The business, expecting equal fast results as the Lean-Agile competition got, further encouraged this way of working, without adapting the environment and processes around the Lean-Agile development practices to benefit from true agile business value delivery.

# Problems achieving benefits of Lean-Agile

Traditional project cost accounting and product management practices don't coexist very well with Lean-Agile development. The major Agile Enterprise Frameworks like Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS), Disciplined Agile [Delivery] (Da[D]) and LeadingAgile identified the common problems in achieving the true benefits of Lean-Agile:

- Demand management practices drive teams to make fine-grained decisions far too early in the 'cone of uncertainty.' If they can't define the functionality and identify all the tasks needed to deliver, how can they reasonably estimate how many people are needed, and for how long? Their hand is forced. [SAFe]
- Fixed detailed project and product scope is anchored in yearly work plans, no room for agile adaptation to changing (market) situations and needs.[4X]
- Line management determines what needs to be done, not the product owner and Scrum delivery teams. [LeSS]
- The project budget process is slow and complicated. It requires many individual cost centre budgets and authorisation layers to fund the fixed scoped projects. [SAFe]
- Today's agile SDLC's are mainly invented from the perspective of software developers. Product management, (requirements) specification and (architecture) design are heavily neglected. [4X]
- Many organisations struggle with 18 month delivery cycles in their Water-Scrum-Fall [LeanAgile]
- The accounting models prevent individuals and teams from working together for longer than the duration of a single project. This hinders learning, team performance, employee engagement and DevOps cooperation practices. [DAD]
- The mind-set in traditional organisations is still project and release based, predefined bundles of functionality are specified and delivered in long cycles. [4X]
- The model drives cost center managers to make sure everyone is fully allocated to one or more projects. However, running product development at 100 percent utilization is an economic disaster [2],

resulting in high variability between forecasted and actual, time and costs. [SAFe]

# Agile Enterprise Frameworks a nice start

The solutions for total “enterprise agility” provided by the (partially obsolete) Agile Enterprise Frameworks are a step in the right direction, but do not take into account the latest IT developments in continuous delivery, micro services, DevOps, containers, automated Quality Frameworks, outsourcing etc.. Another flaw of many Frameworks (except DA[D]) is that these are confined to application delivery and not look at IT service delivery from a broader perspective as will be encountered in larger organisations.

The major approaches for bringing total agility into large enterprises, SAFe, DA[D], LeSS and LeadingAgile, all (mainly) focus on the non-continuous application delivery aspects of IT Solutions & Service delivery. IT Service design and implementation are not part of these frameworks, “IT service” is not even mentioned in SAFe or LeSS. On top of that these methodologies are still in the traditional Release Trains age, not making use of the latest practices and IT technology like continuous delivery via a microservice architecture and containers. Only DA(D) states the importance of other disciplines (hence Disciplined Agile) like Service delivery and Enterprise Release Management. Therefore in most organisations the suitability of these frameworks is limited to the maintenance phases of the ALM, based on non-continuous delivery practices executed by fixed maintenance teams and not in project mode.

SAFe is build around the ARTs (Agile Release Trains), a practice that converges on todays continuous delivery practices. The primary disadvantage is that development teams are constrained to a common release schedule, making it difficult to support lean or continuous delivery strategies.

The Big Six and successful Tech Startups tackle the total agility problems without sailing blind on predefined Agility Frameworks. Today’s leading Tech companies go further where the Agile Frameworks stop, making use of the latest practices and IT technology. Their common denominators in successfully completing the quest for total Lean-Agility will be the guidance for the total agile Enterprise Change Delivery practices and agile Enterprise SDLC described here.

Devolved funding and value streams are interesting concepts bringing total Agility and continuous delivery closer to large organisations. On top of that agile Portfolio & Release management and agile Enterprise SDLC practices should be implemented to reach total Lean-Agility in application change delivery.

IT organisations responsible for more than publishing a handful of Apps or Websites, with (internal) customers that depend on their IT Services, should also take into account the Operational & Support organisation and corresponding processes as part of their delivery process. The IT Service delivery should be aligned with the business process implementation. All streams (infrastructure, application, service and business) in the overall corporate chance delivery orchestrated by the holistic discipline Enterprise Release management.


SAFe only applicable for feature delivery of already running and automatically updateable systems as part of an also already operational IT service. Obsolete release trains

DA[D] also takes into account capabilities like DevOps, enterprise architecture, data management, portfolio management, and IT governance. This framework shows how to be effective at IT in general and how to be effective across the organisation. A disadvantage of Dad is that it still lives in the world of projects Inception where you initiate the project,

Construction where you build/configure the solution, and Transition where you deploy the solution into production or the marketplace. All not very much into the spirit of continuous and disrupting.

**Read more**

1. BCG: Four best practices for Strategic Planning
2. BCG matrix

**Sources**

1. Strategic planning models
2. Strategic planning tools
3. Corporate Business portfolio tool for Business Strategy goals

# Todays Enterprise Agility goes beyond SAFe, DA, DevOps, Scrum etc.

The key to establish Enterprise Agility is in transforming the work style of the Strategic Management and the total Change Delivery Organisation into Lean-Agile. Agility should not be limited to the Application Development Teams only. The Agile Enterprise Frameworks (SAFe, DA[D], LeSS and LeadingAgile ) provide some useful points of departure, but extra miles need to be made to reach the level of agility that the leading Tech Companies have.

Enterprise Agility starts in the top of the Lean-Agile organisational pyramid where the source for the devolved funding streams dynamically wells out of the corporate strategy. Together with dynamic budgeting come decentralized decision making and an empowered organisation. This modus operandi emphasises less on estimating and planning, but more on immediately adjusting, as necessary, priorities in and budgets for the established value streams. Upper management maintains control of the streams by strategically dosing the budget for the streams based on stream performance measurement and external factors. The budget is accompanied by strategic objectives expected to be met by the streams. Inside the streams relevant management levels and agile teams make more fine-grained decisions on result prioritisation in line with their ROI responsibility level using agile Enterprise Change & Release Management practices and the agile Enterprise SDLC’s.

# Lean-Agile Financial IT management with Devolved budget streams

Lean budgeting via devolved funding streams allows fast and empowered decision-making, with appropriate financial control and accountability. Instead of traditionally granting budgets to projects, Value Streams in agile organisations are dynamically funded by Strategic Management. Together with the Lean-Agile budget also come derivative strategic and tactic responsibilities for the different management layers (including DevOps and Scrum teams). Decentralized decision making and an empowered organisation provide these management layers the means to meet the objectives for their value streams. In the modern disrupting world, where organisations are financed with venture capital, these Lean-Agile practices are easy to introduce, the Venture incubators and accelerators will as early stage investors mainly focus on the potential of the starting business and not on strict financial control. An important financial governance concern for (especially listed) traditional companies is how to expense costs properly for the creation of intangible assets. This of course includes the costs of IT, including the costs associated with agile software development teams. Capital Expenses (CapEx) and Operating Expenses (OpEx) represent the basic categories of business expenses. Devolved budgeting conflicts with accounting practices and standards as applied in these organisations.

## Financial IT governance In traditional organisations

Each well-informed CEO/CFO/Controller needs to understand the accounting standards surrounding capitalized software costs for financial compliance and for determining corporate performance. CapEx Influence the book value through the increase of intangible assets ( [Asset management ] ). Lower OpEx will increase the net income. Software development costs can be capitalized, expensed or a combination of both. The accounting standards address these options only at a great length, applying the rules is a matter of subjectivity and opinion. In the EU accounting standards under IFRS (International Financial Reporting Standards) and more specifically IAS 38 for intangible assets are applicable. In the US GAAP (Generally Accepted