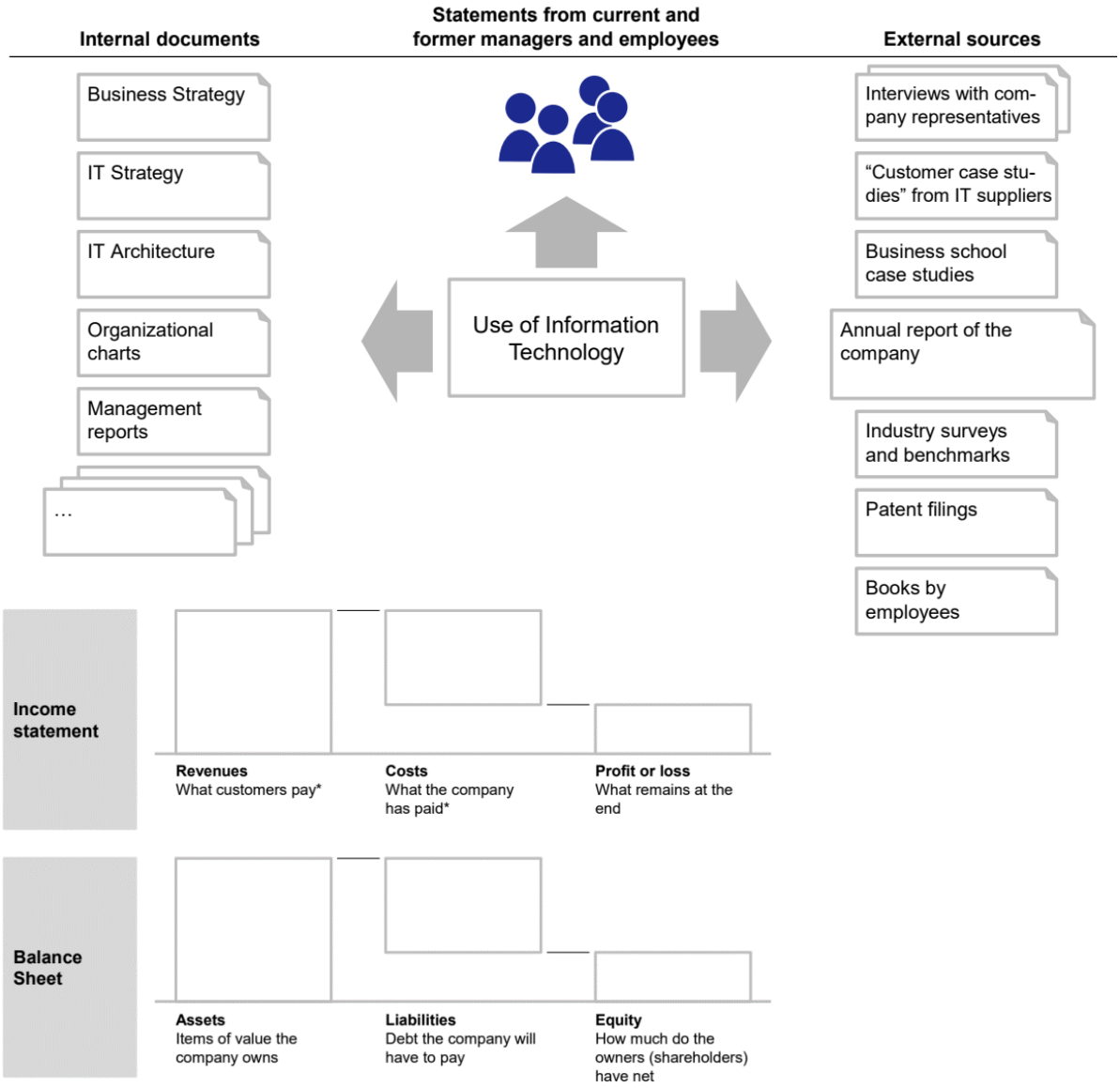
01 Use of IT in companies



- Companies use IT purely as a utility, as a strategic differentiator for their business, or IT is their core business; concrete cases are often not clear cut and are somewhere in between those points; they can also vary by business unit/division/department within a company, and they can change over time
- There are a number of reasons why one would want to understand how a company uses IT, be it as a potential employer, as a potential supplier, customer or business partner, or as a competitor
- There are many sources for information about how a company uses IT; one of the most easily accessible and most reliable source are annual reports of publicly listed companies
- Annual reports may not reveal all information, but what they reveal can be considered to be truthful; they may contain information about the use of IT in various sections, most notably in the description of the business, the strategy section, the risk section, and in the financial statements
- Decisions in companies and businesses, including on IT topics, are driven by how they impact costs and income; it is thus important to understand the financial view of a company, and how IT fits into the picture

02 IT Economics

Investor/ Business Owner
- How much value will I get from investing CHF 1 in this company vs. into other opportunities?
- How much risk is there that I will loose?

Managers/ Board of Directors
- How can we fulfill the expectations of the investors, maximizing the value (equity and RoE*)?
- How can we avoid "blowing up the business" (large risks)?

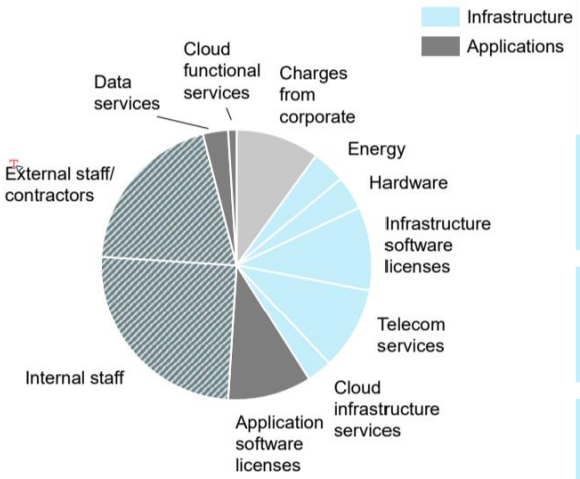
Decisions based on
- Benefits
- Costs
- Risks
- Strategic flexibility

Entrepreneur
- How can I make enough money to pay my bills, incl. the salaries of my staff?
- How can I keep investing to stay ahead of the competition?

Examples

Improved products/ services	• Easier to use, more adaptive products • More efficient, resource consumption reducing products • New fields of product use due to improved products (e.g. as mobile devices, or with less skilled user(s))
Improved processes	• Faster turn-around times • More flexibility to adapt production to market demand • Improved decision making
New business models	• Shift from selling products to providing services • Highly configurable platforms challenging specialized products

- Concrete benefits are highly specific to industry, field of application, and technology
- Sources of inspiration include
  - Vendor presentations
  - Reports from other companies, incl. from other industries
  - Presentations from researchers



Infrastructure Provisioning	• Installing and configuring hardware, network, and infrastructure software (operating system, database systems, etc.)
Software Deployment	• Installing and configuring application software on the infrastructure • Distributing software to distributed devices (e.g. PCs, mobile phones, remote industrial devices)
Systems Manage-ment/Monitoring	• Monitoring running application and infrastructure software • Identifying issues and outages • Taking corrective actions For 24/7 monitoring we need ≥ 4 people. ⇒ Cost ≥ 4 · 250'000 = 1 Mio.
User Support/ Helpdesk	• Handling user questions and issue reports • Taking corrective actions for issues • Administering user access rights • Informing users about planned or unplanned unavailabilities

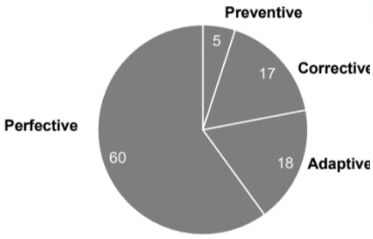
Changes to application software to improve maintainability or to reduce the likelihood of future problems

Changes to application software due to new requirements (functional or non-functional)

	Correction	Enhancement
Proactive	Preventive	Perfective
Reactive	Corrective	Adaptive

Removal of defects in application software

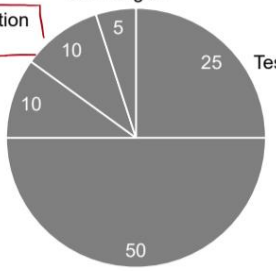
Adaptation of the application software to changes in infrastructure it is relying on or in applications it is connected to

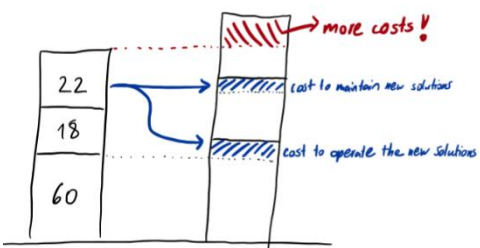


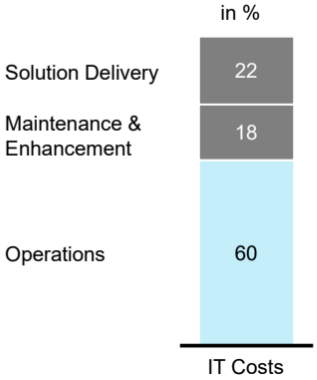
Only 10% of the time we really do changes!

Implementation of changes

Planning of changes







- Operations is the largest cost block
- Typically, only half of software development capacity can be used for (new) solution delivery

COROLLARY FOR IT INVESTMENTS

- The cost structure in IT is largely a reflection of past IT investment decisions, and very hard to change as long as you continue using respective applications
- If you decide to implement an application today, it is a given that you will have operations and maintenance costs when you use the application in the future, which are largely fixed
- A solid rule-of-thumb for estimating expected maintenance costs is 10-15% of original investment costs annually
- Client Example: Effective costs are underestimated by a factor 2.7!

BUSINESS CASE

- Enables the evaluation of whether a project or investment will overall be beneficial for the company
- Enables the comparison and prioritization of different projects
- Clearly describes benefits, costs, and risks of a project, both for the implementation phase and the subsequent ongoing use of the end product (application, system, business process, etc.)

➔ An evaluation of 500 projects of different types revealed that half (53%) of the projects are considered failed (31% partially successful and 16% successful).

➔ Cost overrun of average 89% and time overrun on average 122%.

Benefits

- All benefits obtained when using the product resulting from the project or investment
- Quantified and separated for every year in which it is planned to be used
- Note that "cost reductions" are benefits

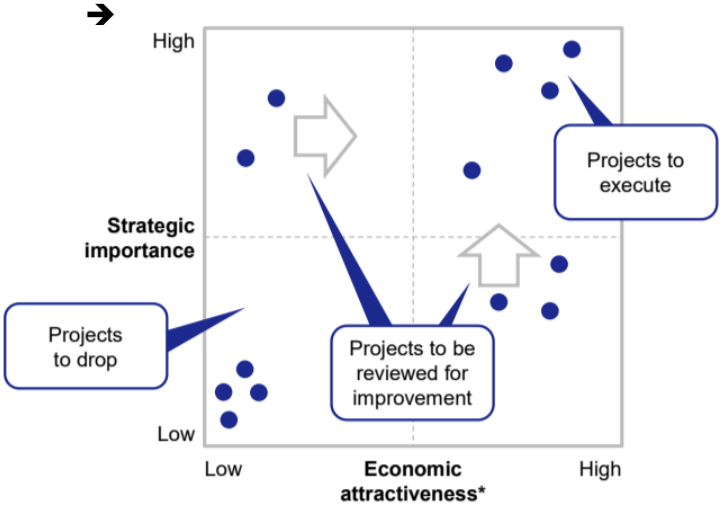
Costs

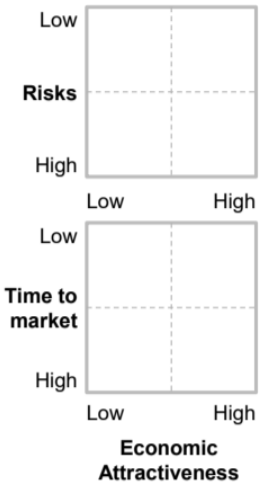
- **Project costs in business:** All costs incurred in the business for the project, e.g., project management, business analysis, user acceptance testing, training
- **Project costs in IT:** All costs incurred to realize the project in IT
- **Ongoing costs in business:** Some projects lead to additional business costs once in use, e.g., for staff using the product, or for advertising the new Services enabled by the project. Since all benefits are counted, also all costs need to be counted
- **Ongoing costs in IT:** Costs to maintain and to operate the resulting application, once it is life

Risks

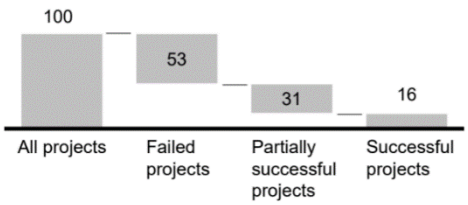
- **Project risks:** Risks that will endanger the project, incl. assessment of how it will impact benefits and costs described above
- **Operational risks:** Specific additional risks in the business once the resulting application is life, incl. assessment how it will impact benefits and costs

➔





Evaluation of 500 projects of different types, in %



Reasons for project failure

Incomplete requirements	13%
Insufficient user involvement	12%
Inadequate resources	11%
Unrealistic user expectations	10%
Insufficient management support	9%
Requirement changes	9%
Insufficient planning	8%

➔ **Cost overrun on average 89%, Time overrun on average 122%, Most reasons are at the interface business/IT**

MEASURING ECONOMIC ATTRACTIVENESS

		Advantages	Disadvantages
• **Benefit/cost-ratio:**	$\frac{\sum \text{Benefits}}{\sum \text{Costs}}$	• Easy to calculate and to understand	• Does not distinguish between projects of different size and time to market
The higher the better.			
• **Payback-time:**	$\frac{\text{Project Costs}}{\text{Annual Net Benefits*}}$	• Easy to understand for simple projects	• Difficult to define for multi-year projects with varying benefits over time
The lower the better.			
• **Net Present Value (NPV):** The difference between the present value of cash inflows and the present value of cash outflows over a period of time.	$\sum_{y = \text{years}} \frac{1}{(1+i)^y} \times \text{Net Benefit}_y$ (i: return, e.g. 10%)	• Captures many aspects of projects	• Relatively complex to calculate and understand • Used as such, favors large projects over small ones
The higher the better			
• **Internal Rate of Return (IRR):**	i solved for NPV = 0	• Similar to NPV • Result relative to project size avoiding bias of NPV	• Complex to calculate and understand
The higher the better.			

* Ongoing annual benefits – ongoing annual costs

If the owner gives our company capital, he/she expects a certain return, measured as "Return-on-Equity" (RoE), for typical large business, 15% or 20%.

RISKS

IT Project Risks
- Project failure/cancellation
- Project budget and time overrun
- Results not fulfilling project objectives

It is very difficult to obtain complete requirements (people cannot imagine everything they will need in a new system). → This difficulty often leads to excessive "flexibility" requirements.

Business Operational Risks
- Business loss, reputational damage, or financial damage due to outages
- Damage due to system malfunction

Cyber Risks
- Intrusion to cause damage
- Theft of intellectual property, business or customer data
- Device abuse (e.g. as part of a botnet)
- Ransom
- Collateral damage of unfocused attack

Attacks and incidents will happen, you can only prepare for them
Think like a hacker

Intangible damages account for over 40% of the total impact
→ Reputation Loss
→ Customer Loss

Stealing valuable information

Demanding ransom

Benefitting indirectly

Mastering an intellectual challenge

Generally, the immediate and visible costs represent only a small part of the incident – the biggest loss for the company comes in the long-term and is hard to evaluate!

STRATEGIC FLEXIBILITY

Key inhibitors related to technology	Good practices
Vendor dependency	• Understand the strategic relevance of what you buy • Understand the strategic intent of suppliers of key components • Aim for either partnerships or vendor independence in fields of strategic importance in which you cannot build on your own • Make smart make/buy decisions
IT complexity*	• Proactively manage the technology landscape • Avoid business requirements of marginal value – Thoroughly review requirements before sign-off, discarding requirements with little/no business value – Conduct a "Project Value Analysis" to force prioritization ("how to do the project with x% less budget in y% less time") • Regularly invest into "architecture cleanup" projects (often called "reducing technical debt")
Lack of skills	• Encourage and support staff to build skills in areas of strategic importance • Hire talent in relevant fields and – let them learn about company and industry through "career starter" programs, stages, conferences – give them room to stay up-to-date on developments in their field – let them explore innovation opportunities, coaching them to make it relevant

MAKE ? BUY

- Capability is a source of competitive advantage
- System is tightly integrated with other internal systems
  - Integration efforts
  - Influences further make/buy decisions
- There is no strong solution in the market

- Capability is a commodity for the business
- There is one or multiple strong solutions in the market
- The needed integration with other systems is relatively low

You can be sure, that the new system is completely integrated in your existing IT landscape.

(Generic) Strategy of Some Suppliers

Leverage market position to grow into other businesses

- **Forward-integration** into customers'/partners' business
- **Disintermediation** of partners'/customers' businesses

Forward-integration

Vertical integration along an industry supply-chain in the direction towards customers/partners with intents such as

- Optimize processes and reduce slack
- Increase market control and power
- Increase profits

Disintermediation

Directly interact with customers'/partners' customers with intents such as

- Eliminating the middleman from the supply chain
- Optimize processes and reduce slack
- Increase market control and power
- Increase profits

CASE STUDY: Building direct customer relationships: **Cisco Connection Online (CCO)** lets customers design networking solutions directly, cutting process time to 2-3 weeks and improving efficiency, while keeping resellers involved and using only Cisco products. This streamlines the process, reduces errors, and enhances customer satisfaction.

ANTITRUST LAW: ESSENTIAL FACILITIES DOCTRINE

➔ You are allowed to build a monopoly and maintain it, but you are not allowed to use it hindering competition in connected markets (**essential facilities doctrine)**

➔ **"Almost monopolists" are often happy to have a small competitor and even support it, so they can claim there is no "bottleneck".** E.g., Microsoft supporting Apple in the 1990's, Google supporting Mozilla (Firefox).

KEY TAKEAWAYS

- The decision makers in a company take decisions based on (expected) benefits, costs, risks, and strategic flexibility
- When generating beneficial ideas, aim to understand the pain of potential users/buyers, not necessarily their stated requirements, come up with a solution that eliminates or reduces the pain
- Business cases for an investment (e.g. an IT project) puts benefits, costs, and risks into relation; the actual decision on or prioritization of a project differs by company, but it is generally driven by some combination of economic attractiveness (e.g. payback or NPV) and other attributes important to the company, such as fit with strategy, low risk, or quick time to market (i.e. short realization time)
- Business cases are usually put together as spreadsheets, listing all benefits and costs, and calculating easily comparable indicators such as payback time, NPV, or IRR
- In more advanced analysis, risk factors can be modelled as different scenarios and respective ranges, e.g. failure of a provider to deliver a component in time can be modelled as a delay, shifting benefits and costs into the future from the moment the risk materializes, and adding additional costs during that time and to address the failure
- Avoid "jumping to conclusions", and do the analysis thoroughly: Lower cost is not always increasing the "economic value" of a project (namely if it leads to later delivery of benefits in highly attractive projects), as an example
- Business case analysis and analysis of project portfolios needs to consider practical aspects: Not all benefits can be expressed by numbers (and thus be analyzed) easily, and delivery time cannot arbitrarily be shortened to offer early benefits
- Companies are exposed to many different types of risks; related to IT, IT Project risks, Business Operational risks, and Cyber Risks are particularly important
- IT Project risks are very large, with many large projects failing or not delivering what they are supposed to deliver. This is one of the reasons why many companies prefer to buy a solution or service rather to build one, even if they believe they could do better than what is available in the market
- Many IT project risks are related to the interface between business and IT, and to the ability to collect business requirements
- Losses due to operational risks can be huge, and being able to continue operations while others cannot can be a huge advantage
- Cyber-attacks and respective breaches will happen in every company; the impact is much bigger than having to restore systems
- Strategic flexibility is obtaining by avoiding inhibitors for change, namely vendor dependency, IT complexity, and lack of skills
- The business strategies of some IT suppliers, namely forward-integration into and disintermediation of customers can be a large threat to existing businesses
- After about 20 years of relatively weak antitrust enforcement, there is renewed activity in the field, especially related to very large technology corporations; decisions will have large impact on the IT industry, potentially limiting what large corporations can do, protecting other market participants (customers, partners, competitors) from strong-armed tactics, and opening up opportunities for others
- The financial analysis of project portfolios provide insights into how changes to projects (e.g. cutting or increasing budgets, or delaying or cancelling projects) impact project value for the company, laying the ground work fact-based decisions on such changes

03 Technology Selection

Considerations: Innovation pace, trade-offs (breadth vs. depth), capabilities overlap, stability/maturity, integration risk, skill availability, ...

One technology solution covering all/most requirements

- ⊕ Only one technology and one supplier
- ⊖ More complex technology
  - Less fit with individual set of requirements
  - Stronger dependence on supplier

Sets of requirements from different business stakeholders

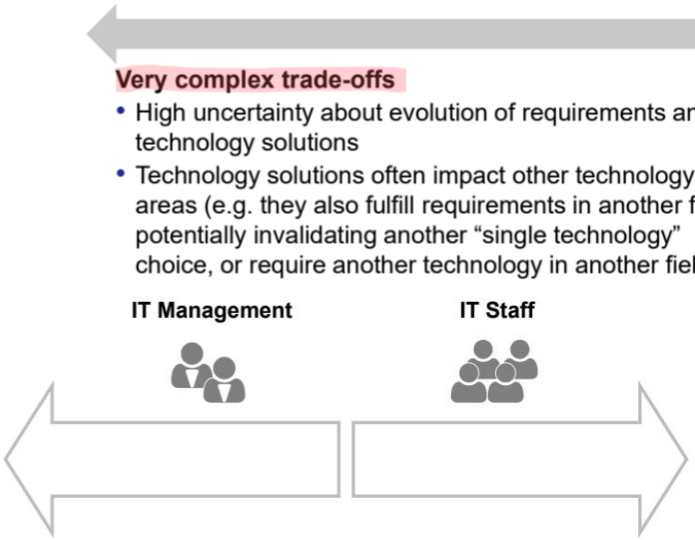
Different, specific technology solutions for each set of requirements

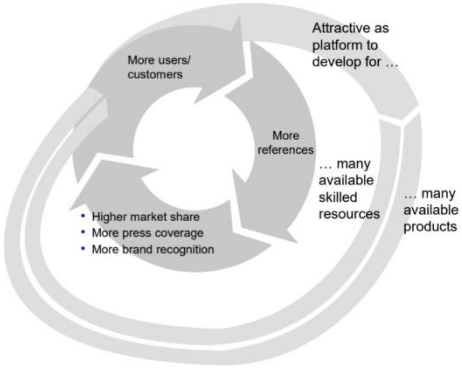
- ⊕ Close fit with each individual set of requirements possible
  - Each technology solution likely simpler and more flexible
  - Less dependence on single supplier
- ⊖ Multiple technologies and multiple suppliers
  - New internal decision for each project which technology solution to use

Very complex trade-offs

- High uncertainty about evolution of requirements and technology solutions
- Technology solutions often impact other technology areas (e.g. they also fulfill requirements in another field, potentially invalidating another "single technology" choice, or require another technology in another field)

IT Management **IT Staff**





One technology solution

- ⊕ Less solutions and suppliers to manage
  - Simpler decision making internally and enforcement once established
- ⊖ More difficult and controversial to choose initially
  - Stronger dependence on single supplier

Different, specific technology solutions

- ⊕ Solution more tailored to problem at hand
  - Less to learn
  - Less to coordinate with others
- ⊖ Only relevant in this field, little use if I move elsewhere

COROLLARIES OF SELF-REINFORCING CYCLES

- **In every technology field, there is a strong drive towards an oligopoly** (market with only a few large sellers): IT suppliers tend to avoid monopolies to escape anti-trust regulations (Examples: Mobile OS with iOS and Google Android, Web Browsers with chrome, firefox and safari, Desktop OS with Windows maxOS, Linux)
- **Dominating products tend to be technically outdated:** Dominating products have become hard to change (Examples: Internet Protocols, Microsoft Windows)
- Leveraging the company brand for overselling: Large, successful IT companies enjoy a large credibility advantage in the market and can sell immature or weak products with relative ease (Example: AirTag)
- **Suppliers changing their business model from software provider to service providers:** Many software suppliers have announced to offer future produce releases only as service in the cloud (e.g. Microsoft Exchange, Adobe Photoshop)
- Customers being sued by 3 rd party patent holders

CASE STUDY: OPEN-SOURCE SOFTWARE

Four typen of OSS: Hobbyist(e.g. Linux), Academic (e.g. PostgreSQL), Professional (e.g. Google TensorFlow), Sponsored (eg. Mozilla Firefox)

KEY TAKE-AWAYS

- It is (almost) impossible to obtain an overview of potential technology solutions in the market for a given business problem
- One of the most profound trade-offs to make when choosing a technology is whether to have one solution for all of the company, or more specific solutions by use case
- The market dynamics lead to self-reinforcing cycles of adoption, ultimately yielding oligopolies in every technology field, with rather outdated products, that will continue to strive until a major innovation makes them obsolete
- Capturing market share as a new entrant in an existing technology field is very hard; the only likely way to succeed is to create a product that solves major pain points customers have with existing products, sell it to a few customers while not yet being visible for the market leaders, and then rapidly scale up
- Achieving growth and not violating antitrust laws can become very difficult for a market leading company
- When selecting a technology, strategic decisions by the supplier can have large impact on your business, e.g. decisions to only offer software as a service; this makes it important to understand the strategy of the supplier, and to remain capable of switching to other suppliers
- Free Open-Source Software (FOSS) can reduce vendor dependency, but practically comes with its own set of strategic challenges, be it for users of FOSS, for companies offering their products as FOSS, and for other market participants

04 Software Metrics and IT Complexity

The productivity in software development decreases dramatically with the increasing size of a system.

Usage	Benefits
FPA can be used to estimate: • Efforts: how many person months does one require to develop a software system? • Costs: how much are the total development costs? • Schedule: when will it be done? • Deliveries: all of the above for individual deliveries (i.e. releases)	FPA offers: • Technology independence: regardless of language, development method, or hardware platform used, the number of function points for a system remains constant • Variability: FPA can be employed for all kinds of software (Realtime, MIS, ...) • Completeness: FPA measures all activities, not only coding • ... and it allows for statistical analysis

100	1'000	10'000	100'000
• Very small systems • Built by single person or small team within a few months • Cost ~ 100k CHF	• Small systems • Built by team of 5-10 staff within 1-2 years • Cost ~3m CHF	• Typical major systems in large corporations • Built by large teams over 3-5 years, often extended over many years • Cost ~50m CHF	• Largest systems in existence, very few examples • Built by very large teams over many years in several major releases • Cost one billion or more CHF

FUNCTION POINT ANALYSIS:

A metric to assess the quality and productivity of software.

A function point is a unit of measurement to express the amount of business functionality an information system provides to a user. The cost (in dollars or hours) of a single unit can be derived from past projects.

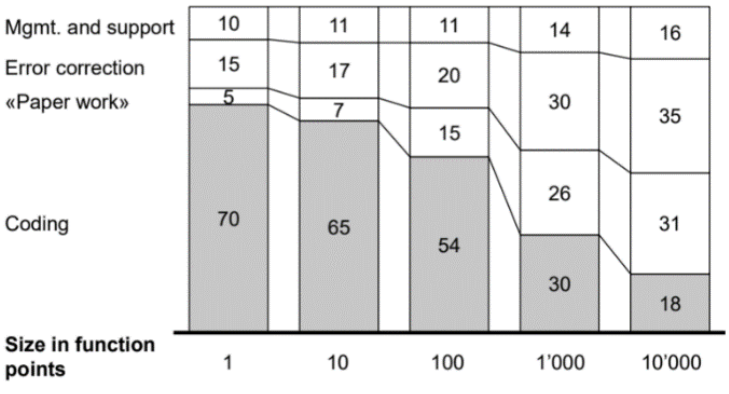
Lines-of-Code (LoC) is a bad measurement!

CAVEATS OF FP-ANALYSIS

In a very strict sense, Function Point Analysis does not measure, but estimate software size and complexity

IMPACT OF PROJECT SIZE ON PROJECT SUCCESS

The larger the project, the higher the proability that the project is cancelled,delayed or of very poor quality!



KEY TAKE-AWAYS

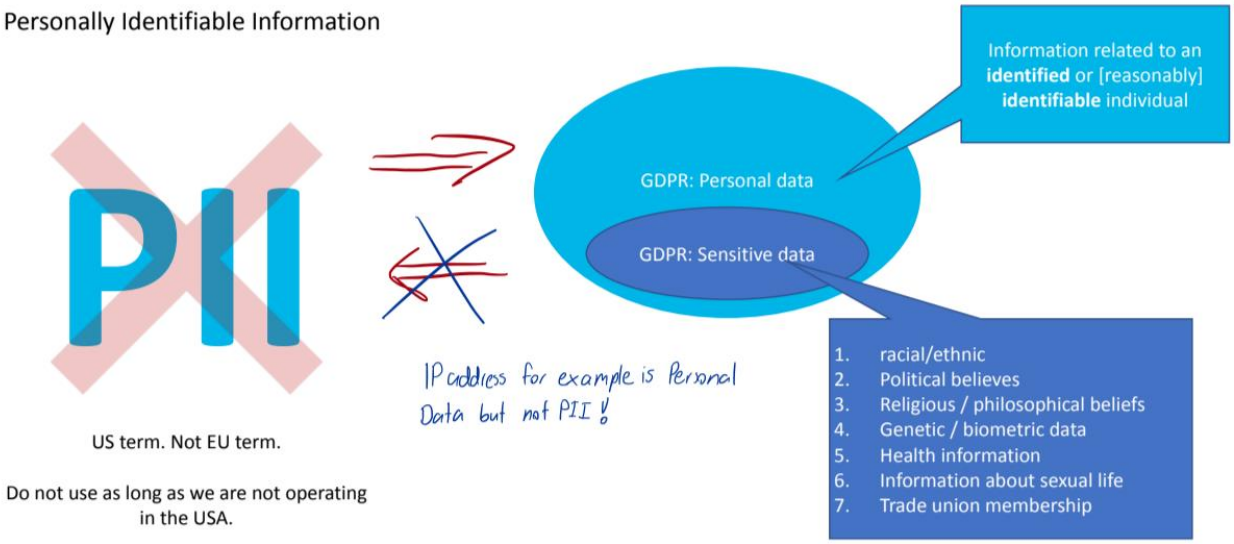
- The productivity in software development decreases dramatically with the increasing size of a system – in particular, the efforts for non-programming related tasks increase overproportionally
- Be careful when combining projects with supposedly similar requirements – synergies can only be realized if the overlap of requirements is high
- Each new piece of functionality developed has to be maintained in the following years and increases the complexity when implementing new requirements
- Rigor when getting rid of marginal requirements pays off: removing them decreases costs above average, decreases risks and increases flexibility
- Even if you buy software, the complexity of the bought solution matters: Complexity will make the vendor less productive in realizing changes. Customers will experience more difficulty to get the vendor implement new functionality, higher cost, and potentially more defects

05 DATA PRIVACY REGULATIONS

Direct identifiers	Quasi-identifiers	Pseudonymous data	Anonymous data
Directly identifies a person.	**Attributes that can help identify a person.**	**Person <u>can</u> be identified with a <u>reasonable</u> effort.**	**Person <u>cannot</u> be identified with a reasonable tech effort.**
• Email • Name, surname • AHV / SSN • Passport Nr	• Date of birth • ZIP Code • Blood group	• Quasi-identifiers if a link to the person can be reverse-engineered • Aggregated data if can be mapped to persons to reveal any personal information about them.	• Aggregated statistics that cannot reveal any personal attributes. • Datasets with personal information <u>irreversibly</u> removed

(Zip-Code, Gender, Date of Birth) is unique for 87% of US population!

DATA PROTECTION MANTRA: "Say what you do. Do what you say. Don't surprise the user."
Creepiness-test: Might a user pursue what we are doing as being creepy?

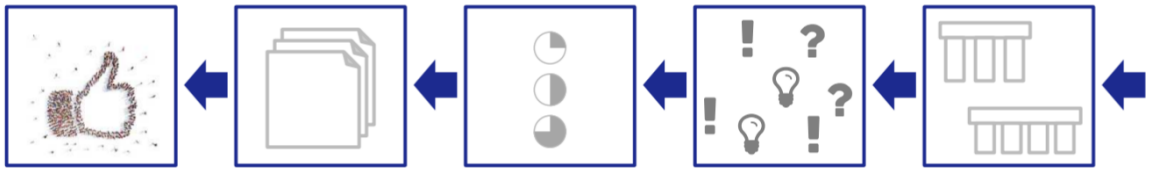


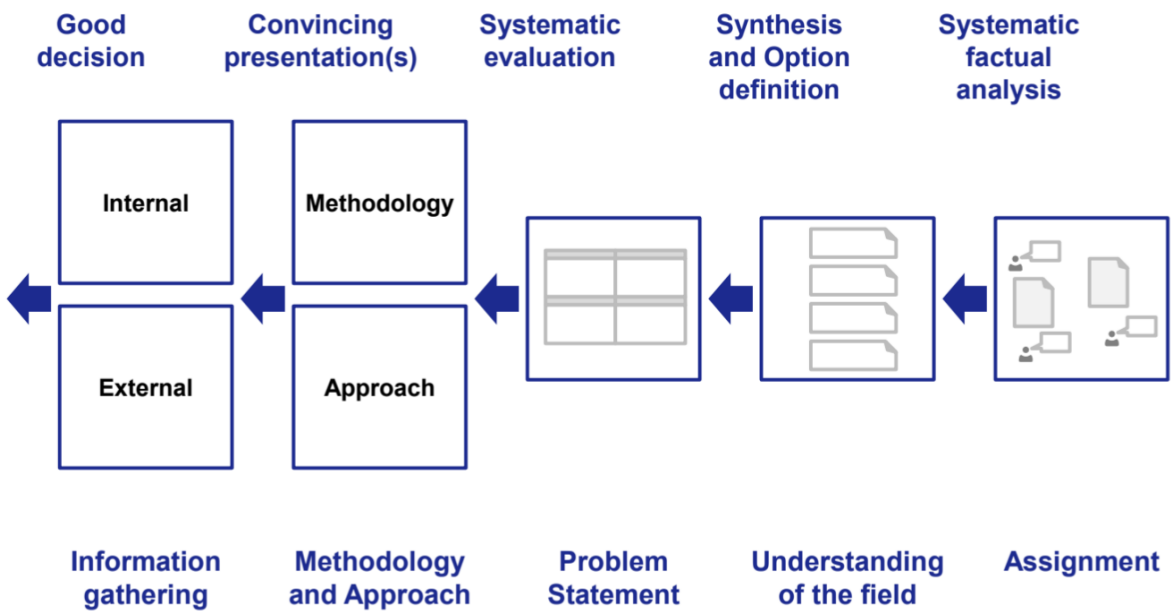
For each piece of data, sit together with people who need it and decide (1) for which purposes (2) for how long (3) to which extent (not all data maybe only few attributes) they need.

KEY TAKE-AWAYS

- Data privacy regulations influence strongly what companies can do with data they have obtained, impacting strategy, business processes, and IT architecture
- Not complying with regulations is not an option
- A security breach does not release a company from its obligations regarding data privacy
- Outsourcing the processing to a third party does not release the company from its compliance duties

06 Business Problem Solving





PROBLEM SOLVING TOOLBOX – SUMMARY:

- Step-wise Approach: Structured problem-solving prevents jumping to conclusions; involves defining the problem, structuring, analysis, and synthesis.
- Problem Definition: Clear understanding aligns with stakeholders and focuses on relevant issues.
- Problem Structuring: Breaks down complex issues into manageable pieces using issue trees and hypothesis trees.
- Analysis: Validates arguments through data collection and stakeholder input.
- Synthesis: Generates solution options, evaluates feasibility, benefits, costs, and risks.
- Recommendation: Chooses the best option and outlines next steps, with iterative refinement if necessary.

07 Legacy Software

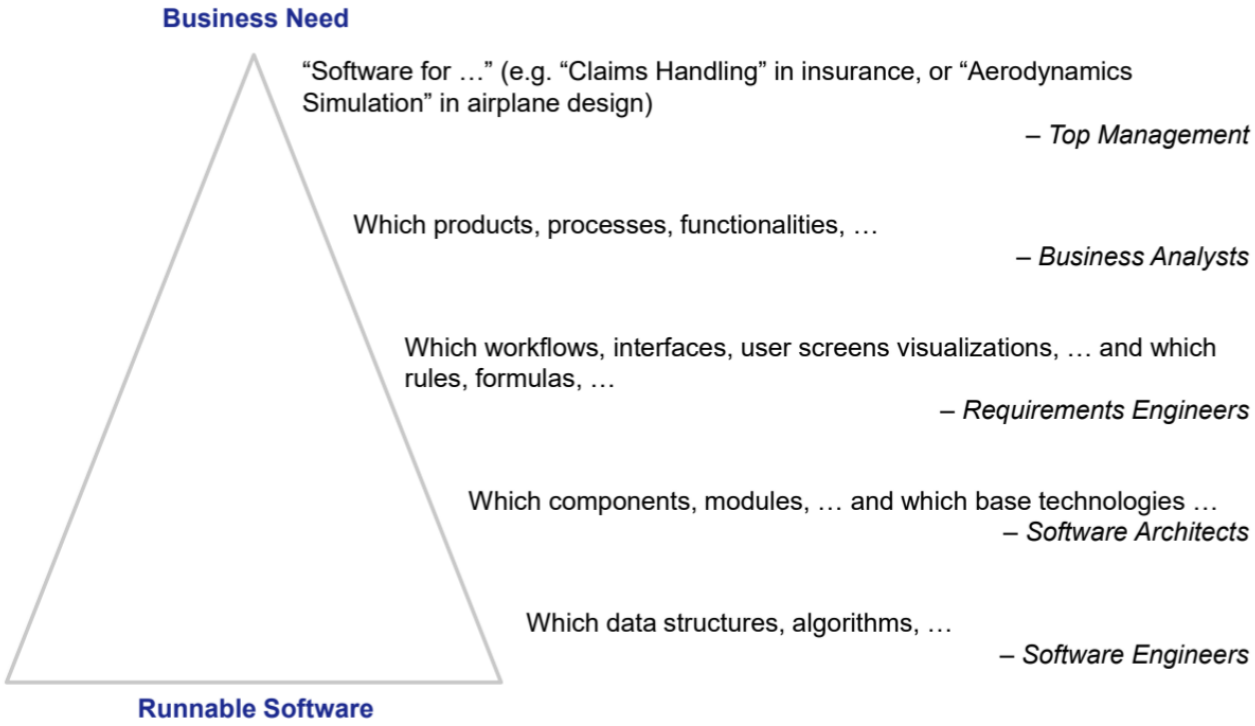
HOW DID COMPANIES GET THERE: IT staff leaves, architectural tweaks, System growth, Changes done without fully understanding the system.

Enterprise software packages (80's/90's/2000's)	Fat client applications (90's)	Microsoft Office	Leading internet companies
• Industry solutions, e.g. in banking, insurance, retail, transportation • Design principles – Real-time processing – Single instance centralized data store – Full transactional consistency at any time • Specific issues today – Monolithic design around central DB – Weak scalability	• User-driven developments on client systems (i.e. PCs) • Inconsistent design principles (or none) • Leveraging libraries and frameworks from external and internal sources • Specific issues today – High degree of duplication – Many dependencies – Very weakly understood	• Office package grown over many years, >30m LoC (2006) • Inconsistent functionalities between applications, e.g. text editing and display	• Employee statement: "Most of our work is maintaining and enhancing what we built 10 years ago." • Large investments in tools to manage code base

POTENTIAL LEVERS TO ACCELERATE DEVELOPMENT OF BUSINESS FUNCTIONALITY:

Technical Migration (migrate gradually onto new infrastructures), Analysis and Reengineering with large teams, Migration to software packages, Use of industry-specific frameworks, use of domain-specific-languages, **use of AI**

LEVELS OF ABSTRACTION FOR BUILDING SOFTWARE



Many hand-over points between different groups of people, with every group having to assume the work of the preceeding groups is complete, accurate, consistent.
The only feedback loop is at the very end when the software is delivered.

Buy a (little configurable) package

- Low effort for software itself
- Generally requires changes to products, processes, …
- Often leads to user-driven add-ons and workarounds (e.g. using Excel)
- Integration challenging

Buy a (highly configurable) package

- Business Analysis and parts of Requirements Engineering still needed, now with constraints of the package
- Requires highly specific skills regarding the package (e.g. SAP)
- Integration challenging

Buy a industry-specific framework

- Similar to buying a package, but providing more flexibility to create own extensions

Use a industry-specific Domain specific Language (DSL)

- Similar to buying a industry-specific framework, but offering more flexibility to create own features
- Low Code/No Code products fall into this category

Use generic libraries/frame-works and better development tools and languages

- Increase the productivity of developers to deliver a feature

Use development process automation tools

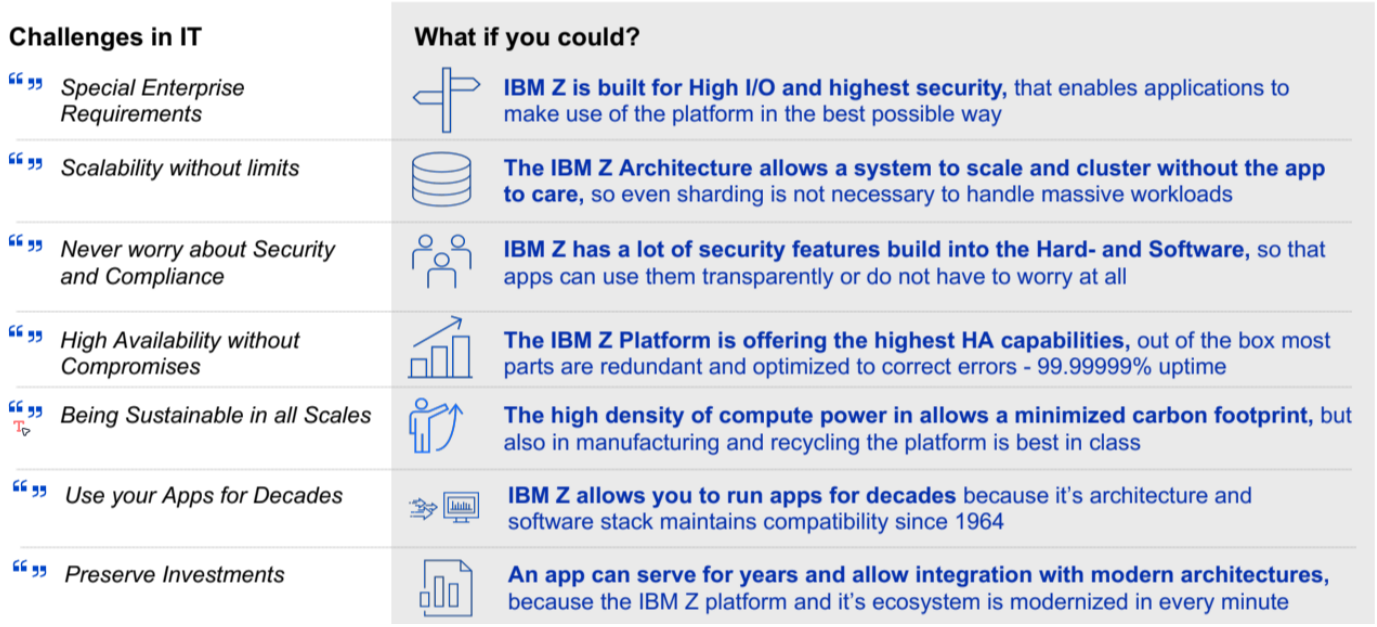
- Simplify/automate communication during development (e.g. requirements management/issue and project tracking such as Jira) and reduce manual activities (e.g. continuous integration/ continuous deployment, DevOps)

KEY TAKE-AWAYS

- Companies did not bother to replace aging software, as they could scale the workload by buying larger systems with more CPU and memory capacity
- Companies later failed to replace such systems (driven by new business demands that were not addressed in the aging application) due to their grown complexity, and due to their lack of detailed understanding
- While the problem of legacy software was first recognized related to mainframe applications, it is widespread today, on all platforms; due to much larger systems being built today and innovation accelerating, software may become legacy more quickly than in the past
- Where legacy software was replaced, it was often by software packages; however, with functionally rich packages generally being quite old by themselves, the packages can be considered legacy as well
- Legacy problems should be expected to become worse, until there are new methods allowing to increase software development productivity by large margins, enabling to build replacement systems at much lower effort, risk, and in much shorter time than the original application
- While there are large advantages in terms of obtaining something to demonstrate and use within a short period of time, agile methods come with their own set of disadvantages
    - The architecture can easily become a patchwork, unless regular architecture clean-up steps are taken
    - It is very difficult to calculate an accurate business case beforehand, with both requirements and respective benefits as well as costs not being fully defined
- Companies aiming to improve productivity (i.e. getting software with less effort, less risk and more quickly) use a number of different approaches, from buying software or frameworks, to using domain-specific languages and better languages and libraries to software development process automation tools

08 Mainframe (IMB Z) Architecture

IBM mainframes run ~70% of all world transactions by value, 77 of the world's top 100 banks use an IBM mainframe



- Despite calls to replace mainframe computers since 30 years, they are still in place
- Key reasons include investment protection for customers (IBM provides backward compatibility to the very first model in 1964), continuous improvement/extension of the architecture, very high scalability and very high availability
- The architecture is also specifically tailored for commercial workloads, with special instructions, tuned memory hierarchies, specialized I/O subsystems, clustering facilities, as well as optimized software
- Being able to scale within a single machine or a tightly-coupled cluster of machines has advantages over distributing the load over a network of smaller machines, be it in terms of manageability or software complexity, depending on the needs of the application • IBM Z systems allow to detect failure of individual

components, restart of the activity on another component (down to instruction level), deactivation of the failed component, and replacement of the component, all while the system remaining in operation
- There is more growth in new applications for IBM Z systems than growth in classical mainframe applications
- Beware of prejudices: Just because something is old does not make it obsolete; highly specific solutions (such as IBM Z) can be great solutions for highly specific needs
- IBM sells its Z (or Linux One) systems mainly under the labels “high scalability” and “high reliability”.
- Both high scalability and high reliability can be achieved on other systems – mainly by “scaling out” over many machines and using distributed algorithms
- However, using an IBM Z system for such needs may reduce time-to-market, cost, or energy consumption, despite its much higher hardware cost
    - Very large number of processing cores in a single machine, all having access to a single shared memory
    - Very large memory capacity
    - Very high I/O bandwidth, and lower I/O bandwidth needs compared to distributed algorithms
    - Clustered environments with shared time and shared locks (“Parallel Sysplex”)

09 Swiss Bank Case Summary

Project NextGen: Initiated to replace the legacy GAIA mainframe system to support complex financial instruments and reduce dependency on hard-to-find mainframe specialists.

Issues: Project suffering from delays, budget overruns, and complexity in integration and requirements gathering.

Key Problems:

- Cost Overruns: Current IT budget significantly increased due to NextGen and GAIA operating simultaneously.
- Resource Challenges: High costs for external developers and difficulty finding internal staff with necessary skills.
- Complex Requirements: Business requirements are unclear and changing, causing delays and scope creep.
- Project Management Issues: Inefficient project start and lack of business support.
- NextGen already has more FP than GAIA while supporting barely any business → a lot of the remaining activity would be about rebuilding functionality that is already in GAIA

In the end, the factors that matter for management decisions are benefits, costs, risks, and strategic flexibility.