CORE®

Anti Aging for IT

Maintaining flexibility - managing renewal

Tatsiana Bychkouskaya
Christian Böhning
Dr. Carsten Wedekind
Karsten Trostmann

April 2020
Blogpost

Key facts

- IT systems are aging faster due to the increasing pace of technical development in recent years.
- The implementation of new requirements is causing IT systems to deviate more and more from their original design. Without appropriate countermeasures, systems lose their conceptual integrity.
- Loss of integrity leads to more complexity in the systems, reducing adaptability and leading to higher maintenance costs.
- The approach of evolutionary architecture tries to slow down and limit the effects of the aging process of IT systems.
- An important component of evolutionary architecture is to record and manage the technical debt in IT systems.

Report

Our service portfolio as strategic IT management consulting firm includes the analysis and evaluation of the technical condition of IT systems. These architecture reviews are usually triggered by poor adaptability of existing IT systems to new requirements, or inadequate performance and quality of the implementation of requirements and the resulting dissatisfaction of system users. As a result of the analysis and evaluation conducted in such a review, we often have to break the bad news that the IT systems under review are either outdated or in a very bad technical condition. The sustainable solution would be a comprehensive restructuring or complete replacement of the systems. Both options require time and resource intensive projects, that tie up available capacities and make it difficult to further develop other systems.

The IT managers of the affected systems rightly ask themselves, how they have come to find themselves in such a situation. In one specific case, those in charge were unable to explain why their relatively young IT system, not even 20 years old, modelled clean from scratch and implemented predominantly in Java, was constantly suffering from quality issues, and why making changes was a complex and lengthy process. The main reasons for this were:

- Since the system went live in 2000, extensions and adaptations increasingly led to a deviation from the original concept and design of the IT system. The original clean and stringent design had become a mix of different concepts and frameworks.
- The system had undergone a great deal of technical development during its lifetime. This also included adding features to perform tasks, which it was not originally designed to perform, leading to a vast amount of workarounds, additional interfaces to other IT systems - and ultimately more dependencies.

This pattern can be observed in many architecture reviews and so it becomes apparent that IT systems undergo a process not unlike the aging process of humans. Both become more sluggish with age and lose the ability to adapt quickly to changing external conditions. Their efficacy and reliability decreases, while the expenditure for care and maintenance increases.

The ageing process of IT systems is not a newly discovered phenomenon. However, its importance has increased even further in recent years. Technological innovation cycles are becoming shorter and shorter, and IT systems are playing a much more important role in value added chain as the digitalisation of processes in companies progresses. In order to limit both the effects and the speed of the ageing of IT systems, the causes must be understood and suitable countermeasures must be introduced in good time.

Why do IT systems become obsolete?

Computer science considered the process of aging of IT systems and software as early as the 90s. None other than David L. Parnas - to whom we owe, among other things, the encapsulation and secrecy principle, which is fundamentally important for any non-trivial system - explained the two main causes in his essay "Software Aging".

He cites as first cause, that systems are designed according to design paradigms that are aligned with the technical, economic and organisational conditions of their epoch. However, these framework conditions are subject to permanent change, which in turn is largely determined by

technological development. Starting from the individual technical innovations themselves, methods and procedures adapted to them are also subject to change.

This can be clearly understood by considering a design paradigm of IT systems derived from prevailing framework parameters. In the 80s and 90s computing power and storage space was expensive. The design paradigms derived from this were the concept of the central computer, the maximum reuse of code and paucity as the supreme principle of data modelling. There paradigms have given rise to highly problematic designs under today's framework conditions. The late effects of which many companies are suffering today. What was good and right as a solution pattern then is an anti-pattern under today's requirements for flexibility. Current methods and procedures of agile software development cannot be adapted without significant limitations on the old technology platforms and system structures of the 90s. The possibilities for efficiency and quality improvement created by the IT development of the last 20 years cannot be fully utilised. In combination with the increased contribution of IT systems to value creation in companies, this ultimately leads to competitive disadvantages.

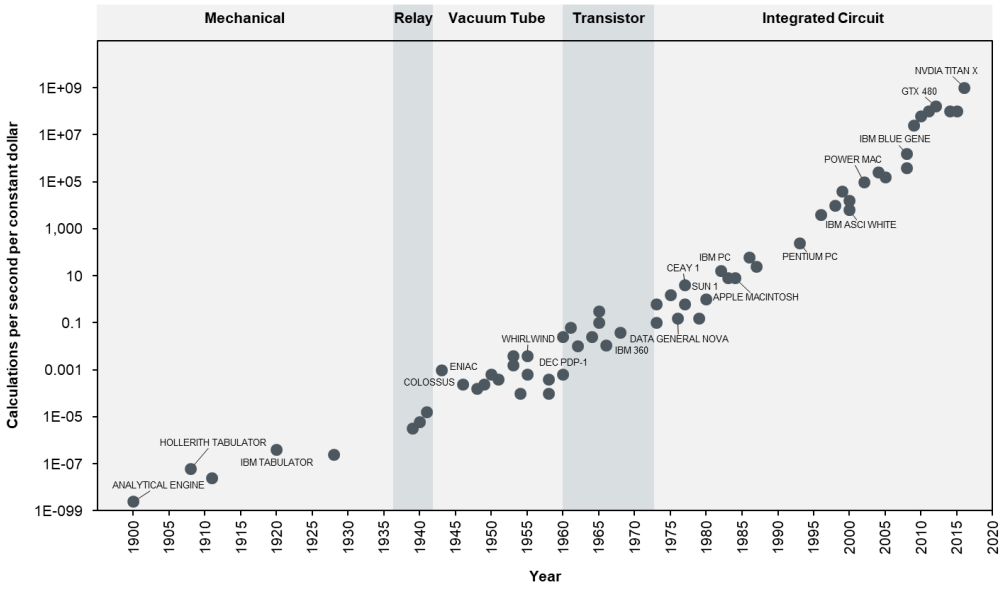

Figure 1: Moor's Law 1900 to 2020

Taking a look at the development of the technology environment, using the example of the historical development of computing power, we see that technological development has an evolutionary behaviour. Likewise as in all evolutionary processes, this development is not linear, but exponential. In the current situation of spring 2020, we are all very much aware of what exponential growth means. The technical innovation cycles are becoming shorter and shorter and to the same extent the framework conditions and design paradigms are changing. IT systems are

thus becoming obsolete at an even faster rate due to the constantly accelerating pace of technological development.

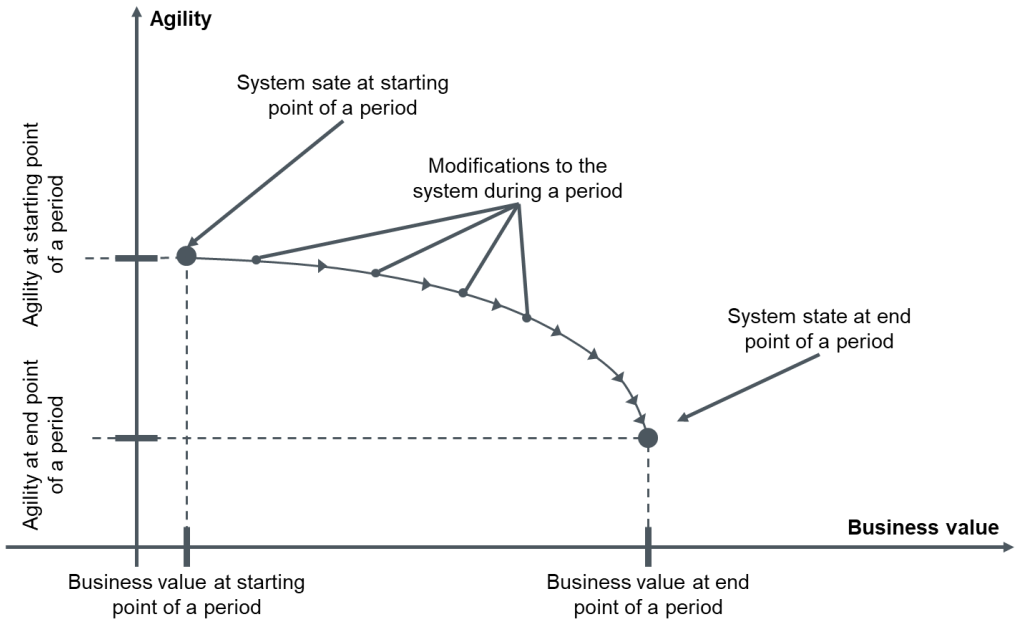

Figure 2: IT system agility and business benefits

The second reason cited for the obsolescence of IT systems is the change and adaptation of IT systems themselves. In the majority of IT systems used in companies, comprehensive concepts have been and are being created that follow a specific business objective and technical paradigms. At the time of their commissioning, the systems does also comply with this concept. They thus exhibit a high level of conceptual integrity. However, the incipient change process usually takes a natural, more or less entropic course: As the number of changes and functions implemented increases, the system deviates more and more from the original concept. This conceptual erosion accelerates, as the system becomes older and leads to increasingly poor adaptability and reduced system quality.

We want to present some of the background to this erosion from our point of view. In many cases, the loss of conceptual integrity is caused by change management that focuses exclusively on functional enhancements. Changes to the IT system should be implemented opportunistically with minimal cost and time. The conceptual integrity of the system does not represent a value for the user, is not perceived as a target dimension and is therefore not managed. Major changes, especially in monolithic systems with large functional areas and changing service providers, then lead to a real impetus of erosion.

IT systems are usually initially created in larger projects, often by external service providers. With the going live of the IT system the software is usually handed over from a project team to a maintenance team. The members of the maintenance team are usually only involved in late project phases or are not part of the project team at all. They then have to familiarise themselves with the system on the basis of documentation, which is often inadequate and mostly coined by technical and architectural concepts, and they have to familiarise themselves with an enormous code base. It has become evident that even with the most careful documentation, transition losses cannot be avoided and an "emotional attachment" to the code is (still) missing. Changes are made

pragmatically in the fastest and most direct way, born out of lack of knowledge or ignorance of the original concept.

One further yet decisive point is that the decreasing conceptual integrity is accompanied by a loss of value in the IT system, which is not recorded anywhere, but which burdens on the system like an invisible and constantly growing mortgage. This loss of value is only in very few cases registered and known to the companies. The problem is therefore not at all or only incompletely known.

Anti-Aging for IT systems

Any form of control is based on concrete facts and figures. This is the only way to identify corresponding focal points for action, plan measures and implement them. In software technology, the concept of technical debt has established itself as a measure for the degeneration of a software system.

The technical debt here comprises the sum of all technical deficits that have crept into the IT system over time. These are no concrete technical errors in the software, but various aspects of insufficient quality, which can be classified into the following categories:

- Design or architectural weaknesses (structure smells)
- Deficiencies in the implementation (code Smells)
- Insufficient test coverage or insufficient automation (test debts)
- Insufficient documentation

The technical debt expresses the effort of the measures necessary to bring the IT system back to the desired quality level. Implementation and test debts can be recorded quite easily with the help of tools. For documentation purposes, the introduction of a formal track recording of architectural decisions (architectural decision records) is helpful. Decisions and concepts can thus be traced later. They also serve to document the reduction of structural deficits.

The method of the evolutionary architecture approach is suitable for detecting structural weaknesses. The core of these methods are so-called fitness functions, which are formulated as architecture scenarios and validate the structural integrity of the IT systems in regular reviews (every 3-6 months).

If technical debts are known and recorded, appropriate measures can be taken. The idea of evolutionary architecture management follows the approach of allowing technical debts within a predefined corridor. The interpretation of technical debts as "wrong" architecture decisions that need to be corrected is not appropriate. The technical debt can best be compared to a loan. It can and does make sense to borrow technical debt specifically to exploit competitive advantages or market opportunities. However, they must be recorded, and their effects limited. This is the only way to ensure the long-term sustainability of IT systems.

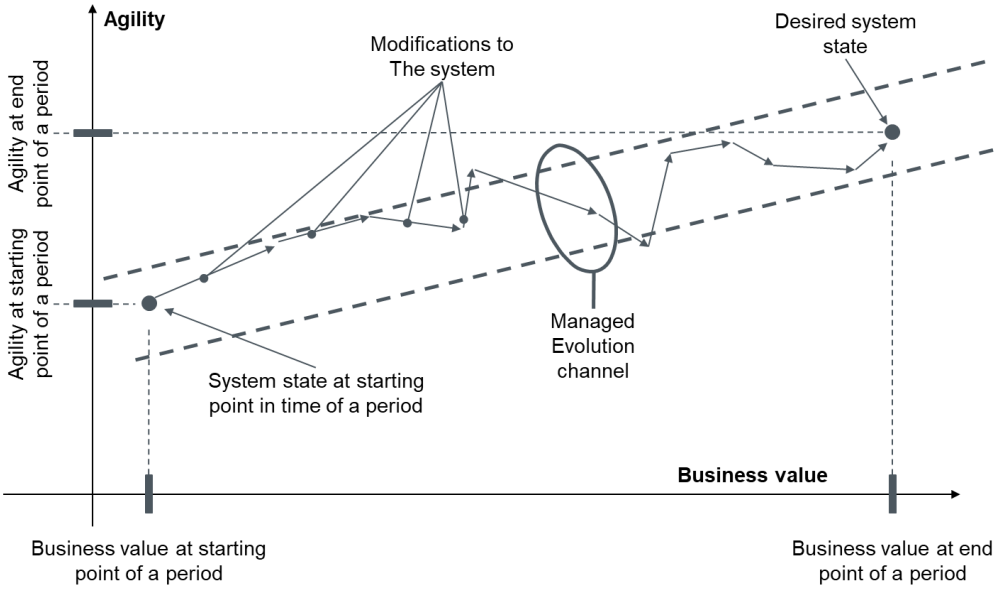

Figure 1: Evolutionary trajectory of managed IT systems

Two principles can be applied to counter the loss of adaptability due to ever shorter technology cycles:

- Disposable architectures: a concrete end-of-life for the system is already defined during the construction of the system and the new implementation is included in the budget as a technical debt. This approach is particularly suitable for IT systems and components that are subject to fast paced change cycles. These include, for example, customer front-ends and APIs.
- Design-to-Change: the architecture of the system is designed for maximum changeability. The principle of a hexagonal architecture is combined with a modularisation exclusively oriented on the technical function by means of Domain Driven Design. The strong functional modularisation and technical decoupling makes these approaches more complex to implement, but all the more robust against changes.

Conclusion

Active management of change is also an inherent imperative with regards to IT systems. This change approaches IT systems from two sides. The power of technical progress as a driver of change can be seen, at a glance, from the exponential increase in the number of possible arithmetic operations - as postulated in Moore's Law. Another driver of change rarely or never appears to outsiders. The range of programming languages and frameworks is constantly increasing and thus requires an expansion of the available architectures, which in turn are subject to their own trends and additional conceptual developments. The technical change offers one thing above all: opportunities. The third driver of change is the constantly changing or increasing business requirements. IT systems are the facilitators of the core business, so implementing changed business requirements is its central function and, in context, has to be regarded an opportunity and less as an isolated nuisance.

The focus of change on business-driven adaptations without accompanying technological changes corrupts the initial conceptual integrity of the IT system. Failing efforts to restore this

integrity leads to further degeneration of the IT systems, which is reflected above all in the decreasing agility to meet further requirements. Unless there is a renewed investment in conceptual integrity, the concept of technical debt, which has become established in this context, also leads to the unavoidable: bankruptcy, the smashing of the bench and thus the loss of the foundation of business.

The evolutionary architecture work can be seen as the technical counterpart to financial planning. In this concept, the compromise of conceptual integrity to meet business requirements in the higher layers of the system architecture can be accepted over time but should be restored downstream by adjustments in the layers below.

Sources

David L. Parnas. (Mai 1994). Software aging. *Proceedings of the 16th international conference on Software engineering* (S. 279-287). Sorrento, Italien: IEEE Computer Society PressWashingtonDCUnited States.

Murer, S. a. (2010). *Managed Evolution: A Strategy for Very Large Information Systems.* Springer.

Tatsiana Bychkouskaya is a Transformation Associate at CORE. Her focus is on strategy consulting for tech start-ups, agile project management, design and implementation of complex projects in the area of nationwide IT infrastructure. Tatsiana's experience includes implementing a nationwide project and strategy consulting for start-ups in the tech industry.

Mail: tatsiana.bychkouskaya@core.se

Christian Böhning is Managing Director at CORE. He has many years of experience in the implementation of technology-driven transformations in the financial industry. His work focuses on IT architecture modernisation programs, implementation of compliance initiatives and the realignment of IT organizations.

Mail:christian.boehning@core.se

As Expert Director at CORE, **Dr. Carsten Wedekind** focuses on the strategic alignment of IT architectures. He is also co-chair of the Lending Working Group in the Banking Industry Architecture Network (BIAN). Dr. Wedekind, who holds a doctorate in physics, has many years of experience in the implementation of IT projects with agile and classic methods.

Mail: carsten.wedekind@core.se

Karsten Trostmann is Expert Director at CORE. The computer scientist covers the following main topics: IT strategy, evolutionary IT architecture, domain driven design, agile software development and cloud infrastructures. Karsten's experience includes evaluating the platform strategy for a digital insurance company, developing cloud operations for an identity provider, architecture reviews in various projects.

Mail: karsten.trostmann@core.se

CORE SE
Am Sandwerder 21-23
14109 Berlin | Germany
https://core.se/
Phone: +49 30 263 440 20
office@core.se

COREtransform GmbH
Am Sandwerder 21-23
14109 Berlin | Germany
https://core.se/
Phone: +49 30 263 440 20
office@core.se

COREtransform GmbH
Limmatquai 1
8001 Zürich | Helvetia
https://core.se/
Phone: +41 44 261 0143
office@core.se

COREtransform Ltd.
Canary Wharf, One Canada Square
London E14 5DY | Great Britain
https://core.se/
Phone: +44 20 328 563 61
office@core.se

COREtransform Consulting MEA Ltd.
DIFC – 105, Currency
House, Tower 1
P.O. Box 506656
Dubai I UAE Emirates
https://core.se/
Phone: +97 14 323 0633
office@core.se