Longevity as an Information Systems Design Concern

Diogo Proenca[1], Goncalo Antunes[1], Jose Borbinha[1], Artur Caetano[1], Stefan Biffl[2], Dietmar Winkler[2], Christoph Becker[2]

[1] IST/INESC-ID - Information Systems Group, Lisbon, Portugal
{diogo.proenca,goncalo.antunes,jlb,artur.caetano}@ist.utl.pt
[2] Vienna University of Technology, Vienna, Austria
{stefan.biffl,dietmar.winkler,christoph.becker}@tuwien.ac.at

Abstract. Digital longevity has emerged as a key challenge for information systems in many domains. In this article we explore the hypothesis that longevity is a non-functional quality attribute of information and software artifacts, driven by organizational capabilities and socio-technical change processes. While software evolution and maintenance have produced a rich body of knowledge on keeping software systems alive after creation, little emphasis has been placed on the underlying factors contributing to the longevity of information systems up-front. We attempt to define, motivate and outline the facets of longevity and pose a series of research questions that lead to a research roadmap and open up an interdisciplinary research path on Information Systems Longevity Engineering.

Keywords: Information Systems Engineering, Information Longevity, Systems Longevity, Digital Preservation

1 Introduction

> *longevity.* 1a: long duration of individual life. b: length of life. 2: long continuance : permanence, durability[3]

This article argues that a new perspective on the concerns of longevity in information systems design can improve the discipline's success in ensuring controlled achievement of longevity in our ultimate core product: information systems. We outline such a perspective and outline an initial research agenda by posing a series of focused research questions. Essentially, *longevity* can be seen as a non-functional quality attribute of an artifact that describes the degree to which the artifact continues to fulfil its purpose for a certain timespan or as long as a defined set of conditions holds. We need to distinguish between the

[3] http://www.merriam-webster.com/dictionary/longevity

longevity of information artifacts, the longevity of the system and software components, and the longevity of the information systems managing these artifacts, while acknowledging these are intricately linked.

Users have understood the business side of the problem, being faced with data loss, software cost overruns, lack of business continuity, and violations of compliance regulations requiring access to data and traceability of business processes. However, in IS design and engineering, there is a focus on ex-post preservation, i.e. maintenance and evolution. This contributes to cost overruns over the system lifecycles and ever-increasing concerns about 'aging software systems' [12] and the huge costs involved in software systems maintenance and evolution [4].

While both the preservation of information and the governance, maintenance and evolution of software and information systems as disciplines are each making advancements, it is crucial to be aware that these are but two sides of one coin: digital data is meaningless without interpretative systems, and systems become pointless without the data they are expected to process.

At the same time, the longevity of the information processed within software systems poses similarly complex challenges. These challenges have evolved into an active area of research most commonly called digital preservation. In this article, we attempt to bring these two perspectives together and outline a research agenda for improving our ability to control the longevity of information systems.

This document is structured as follows. Section 2 discusses digital longevity from the perspective of keeping digital content alive and discusses how notions of longevity are increasingly becoming urgent matters of attention for information systems and software systems themselves, and argues why a more holistic perspective is needed. Section 3 introduces a conceptual perspective for addressing longevity, and poses a number of focused research questions to be tackled. Section 4 summarizes the discussion and outlines next steps.

2 Longevity

To understand the facets of longevity as it applies to information systems, it is useful to introduce a discipline entirely focused on digital longevity: The field of Digital Preservation (DP) is concerned with keeping digital information authentic, understandable, and usable, through time and across changing sociotechnological environments [14].

DP is often seen as a case of interoperability through time. Consider a Shannon communication channel [15] as a metaphor, as shown in Figure 1. The core problem is that transmission is asynchronous and may last an indefinite time. At the time of receiving the message, the original message may not exist anymore, the recipient may not possess an appropriate decoder, the sender may not exist anymore, there may be no encoder to check against, and the recipient may not be the initialy intended addressee. The communication channel thus will often need to convert the message so that the original message intention is preserved. Theoretically, any converter function carrying out such a transcoding should respect the type of the original message [17]. However, the complexity and change

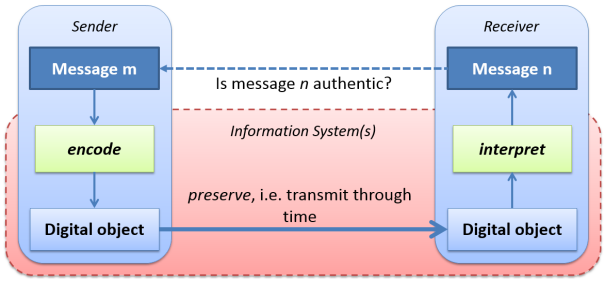

Fig. 1. Information Longevity is communication over time

rate of common environments and information representations today have the effect that this is rarely the case.

From its origin in the areas of cultural heritage and eScience, DP has emerged as a key challenge for information systems in almost any domain from eCommerce and eGovernment to manufacturing, finance, health, and private lifestyle [5, 2, 13]. As the field matures, increasing attention is turned towards the technology stacks and systems from whose rapid pace of change the 'problem' of longevity seems to be originating. Here, the concern of information longevity intersects with software and systems engineering and maintenance. Increasingly, it is understood that we cannot separate the longevity of information entirely from the longevity of the systems that create, manage and dispose of information. The concerns of longevity are being tackled from two sides: The digital preservation community started to build *digital preservation systems* to preserve information, while the software and systems community spent decades on software maintenance and evolution.

We define *information longevity* as the objective that is met if information artifacts remain fulfilling their intended purpose across time for as long as needed. On the other hand, *systems longevity* for an information system can be defined as the objective that is met if it is possible to manage the system over time so that it remains fulfilling its intended purpose for as long as needed.

For *information systems longevity*, that means that

1. the information managed by the system needs to fulfill the objective of longevity,
2. it needs to be possible to sustain the information system, across an unstable organizational and technological context, for as long as a defined set of conditions holds, and,
3. it can be shown that one effectively can, for that system, move the information base and the defined valid states changes (the systems' behavioural schema, or business rules) to another instance of another information system.

What arises from this view is that *information longevity* as an organisational capability relates to the ability to govern information independently of and across systems. Existing approaches to IT Governance, in comparison, are considering information primarily as how it is managed *within* a system.

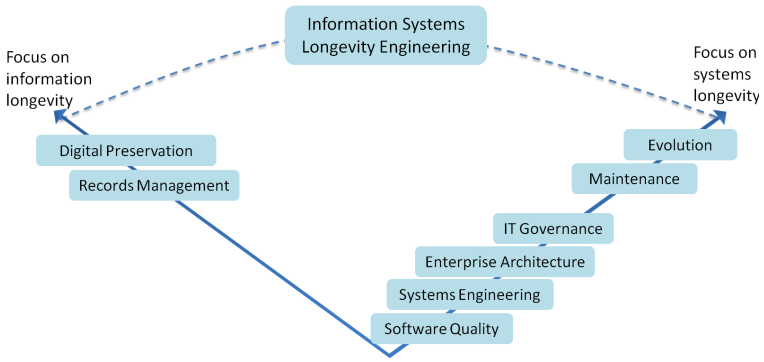

Fig. 2. Information Longevity and Systems Longevity need to converge

The concerns of systems longevity have in different expressions and with narrower focus already stirred interest in the software engineering (SE) and IS communities. However, longevity concerns are generally considered late in the software lifecycle and on isolated levels such as technical portability across platforms, modifiability of source code, interoperability of components, or resilience against disruptions [4]. Uncontrollable lifecycle costs of software and information systems have been a topic of research for decades. However, not much has changed since Parnas lamented about 'software aging' in the 1990s [12]. In a recent article, Neumann reconfirms the lack of long-term thinking as a critical shortcoming. [11]

To counter software aging, software evolution and maintenance refers to the 'modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.' [1] Evolution continues to contribute a large part of the lifecycle costs of software systems. Adaptive and perfective maintenance in many cases contribute three quarters of the evolution costs [10, 4]. These correspond to changing software environments and changing organization environments, which are primary inhibitors to information longevity and hence primary drivers of digital preservation [3]. While there is a rich body of knowledge in software maintenance and evolution on keeping software systems useful over a long time span, this knowledge is often focused on an ex-post view of life support. We need to combine this knowledge with up-front design perspectives on goals and concerns and a better understanding of systems lifecycle management processes.

Figure 2 shows the two key dimensions of the challenge: Information Longevity, until now, is still treated largely as a problem isolated from Systems Longevity and treated as an ex-post phenomenon. Similarly, information systems longevity is generally considered after the fact. While longevity is a key concern in systems engineering, and a set of problem areas are well-acknowledged and increasingly sophisticated in *contributing to* the ability of information systems to achieve longevity, succesfully establishing *Information Systems Longevity Engineering* requires an interdisciplinary effort, shifting from ex-post treatment to a-priori

design. The potential benefits are increased sustainability, improved cost effectiveness, and better control for system users over their information, irrespective of customer-supplier relationships and contextual change. In particular in times of cloud computing, these challenges are emerging as crucial requirements for information systems design. How can we accomodate these concerns systematically?

3 Longevity as a design concern

Goals and Challenges. Certain non-functional quality attributes of systems, such as maintainability and portability, aim at extending the system lifetime on the technological development level. The standard ISO SQUARE [7] quality model elements related to longevity, such as modifiability and adaptability, are horizontal attributes generally considered in the development stage. Quality attributes of software commonly considered include isolated aspects of change (such as changeability, testability and adaptability), but these concerns fall short of addressing the compound issues brought about by technological disruptions, innovation and changes in the systems' environment and context in relation to internal changes of organizational goals and capabilities as well as the system's components and its structure.

Longevity as a design concern, on the other hand, must be considered from the conception phase on, on all levels from the system context to the source code. It has implications for all levels in all successive phases, including retirement. Longevity is not meant to replace existing concerns, but connect them. Correspondingly, an engineering process that integrates longevity will need to bridge existing processes and concerns on all abstraction levels throughout the entire system lifecycle. In fact, longevity can be seen as a crucial enabler for business/IT alignment. Software, as an intermediate logical layer that connects the organization to the physical technological infrastructure and supports necessary and innovative business capabilities, should have the capability to keep up with the organization's goals in the short- and long-term. Not accommodating the concern in software systems engineering causes severe business disruptions since systems do not keep pace with technological evolution and with abrupt changes in the business environment.

Research Strategies. To achieve the specification of such a process, we need a solid conceptual model of the relationship between design concerns, non-functional requirements and quality attributes of systems, engineering processes, artifacts, and process metrics. We thus propose to create a systematic link between longevity processes, capabilities, and systems quality attributes. We believe that it is possible to systematically analyze primary enablers and inhibitors on particular software engineering processes that contribute to longevity of capabilities, software systems and components. The required perspective is provided by Enterprise Architecture [9] and Capability Engineering, hence integrating system lifecycle processes, SE techniques and methods, and concerns such as non-functional requirements.

A primary goal for enterprise architecture is to maintain the alignment between business and IT in organisations [16], which makes it a key tool for infusing longevity capabilities into current SE practices. Based on such a high-level perspective, it becomes possible to analyse the relation of SE processes to longevity-related quality attributes and measurable attributes of software artifacts that can be collected over time.

Based on a systematic analysis of (1) the high-level perspective of Enterprise Architecture, (2) the specific systems engineering methods and processes, process areas, artifacts and metrics for the purpose of engineering capabilities, and (3) empirical software engineering data, we plan to build a conceptual process framework for addressing longevity as a fundamental concern from the conception phase of an Information System's lifecycle. In the following, we outline some of the initial key research questions that we believe can lead the way. The proposed research not only aims at developing systems with the longevity concern accounted for, but it also aims at assessing and improving existing systems with respect to their longevity qualities.

Research Questions. If our goal is to improve the achievement of controlled levels of longevity, where do we have to start? The following section outlines a number of research questions, starting at the ultimate goal to improve the achievement of controlled levels of longevity.

Define, estimate, and measure. How can we define and measure the needs, values and costs of longevity? How can we estimate the desired life spans of software systems and their components? How can we position the factors causing obsolescence and those contributing to maintainability with respect to the desired longevity of the system and its components?

Costs, benefits, and risks. What are the cost-benefit-risk relations of longevity? What is the value added of increasing the expected longevity of a given artifact by a certain time span? What are the marginal costs of increasing the expected longevity of a given artifact by a certain time span? How early in the development lifecycle can we provide serious estimates?

Longevity as a design concern. How can we integrate longevity as a necessary design concern into the IS lifecycle from Conception to Retirement? How can longevity become a first-class citizen in Information Systems Engineering?

Non-functional quality. How does longevity relate to existing quality attributes of software systems? How do elements of existing quality models influence longevity? What are the relationships between certain intrinsic qualities of software such as modularity, extrinsic properties such as portability, and longevity as a contextual quality in time? How can we decompose and separate these qualities to make them manageable? Which aspects have been neglected so far?

Diagnosis over time. Which artifacts and processes do we need to diagnose, and which measures should we collect to diagnose them? How do phenomena such as code decay [6] inhibit longevity? How can we empirically validate hypotheses about correlations between engineering processes and longevity with

the purpose of improving decision quality in SE? Which long-term empirical data can be used to evaluate, on a statistical basis, such correlations and causal relationships?

Architectures and Tradeoffs. In which way does the introduction of longevity as a concern restrict the design space of ÌS? At which point in the development lifecycle is longevity so important that it should take center stage? What is the impact of longevity as a concern on enterprise architectures? How can this be assessed and in which kind of viewpoint? How can we create a set of viewpoints to analyse this impact?

Given a desired set of features derived from longevity as a concern, how can we prioritize and select these and assess the impact of not implementing subsets of them? Whose concern should longevity be at which point of the lifecycle? Given that only a limited number of quality concerns can be feasibly incorporated into a design at any given point, which other qualities might get displaced at which point? What are typical trade-off relations? How can we incorporate these concerns into systematic trade-off analysis methods such as ATAM [8]?

Who should have the responsibility to incorporate longevity as a concern into information systems engineering processes? What are useful patterns for addressing longevity?

Baseline analysis. Which techniques that contribute to information systems longevity by addressing non-functional quality concerns exist in which fields? Considering axes such as the Information Systems lifecycle phases and enterprise abstraction layers shown in Figure 2: Which are the actual dimensions we need to consider to position longevity and the related concerns? Which variables do we need to meausre to assess the current state of art of these techniques towards achieving longevity?

It becomes clear that in order to develop fundamental concepts for integrating longevity as a necessary design concern into the IS lifecycle from Conception to Retirement, an interdisciplinary approach is required that bridges Enterprise Governance of IT, Software Engineering, Requirements Engineering, Digital Preservation, Empirical Software Engineering, Systems Engineering, Information Systems Design, and other disciplines.

4 Conclusion and Outlook

Our hypothesis is that longevity is a non-functional quality attribute of software artifacts, driven by organizational capabilities and socio-technical change processes. By including the fundamental concerns of longevity into the design and engineering of systems on the capability level, longevity as a vertical integrator should be able to relate these development concerns with change on the organizational level.

The definition of a longevity engineering approach as a bridge between isolated perspectives of existing processes will contribute to an increased independence and independent evolution of capabilities versus their constituent parts.

This article set out first elements of a research roadmap towards a fundamental understanding of the factors contributing to software longevity. It is clear that this is a challenging effort requiring focused, longer-term research.

Acknowledgments

Part of this work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT12-046 (BenchmarkDP).

References

1. *IEEE Std. 1219: Standard for Software Maintenance.* IEEE Computer Society Press, 1993.
2. Data's shameful neglect. *Nature*, 461(145), 2009.
3. Christoph Becker, Kresimir Duretec, Petar Petrov, Luis Faria, Miguel Ferreira, and Jose Carlos Ramalho. Preservation watch: What to monitor and how. In *Proc. IPRES*, 2012.
4. Keith H. Bennett and Václav T. Rajlich. Software maintenance and evolution: a roadmap. In *Proc. ICSE*, ICSE '00, pages 73–87, New York, NY, USA, 2000. ACM.
5. Orit Edelstein, Michael Factor, Ross King, Thomas Risse, Eliot Salant, and Philip Taylor. Evolving domains, problems and solutions for long term digital preservation. In *Proc. of iPRES 2011*, 2011.
6. Stephen G. Eick, Todd L, Graves, Alan F. Karr, J. S. Marron, and Audris Mockus. Does code decay? assessing the evidence from change management data. *IEEE TSE*, 27(1), 2001.
7. ISO/IEC. *Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models (ISO/IEC 25010).* International Standards Organisation, 2011.
8. Rick Kazman, Mark Klein, and Paul Clements. *ATAM: Method for Architecture Evaluation.* CMU/SEI-2000-TR-004, 2000.
9. M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication, and Analysis.* Springer, 2005.
10. B. P. Lientz and E. B. Swanson. *Software Maintenance Management.* Addison Wesley, Reading, MA, 1980.
11. Peter G. Neumann. The foresight saga, redux. *Commun. ACM*, 55(10):26–29, October 2012.
12. David Lorge Parnas. Software aging. In *Proc. ICSE*, ICSE '94, pages 279–287, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.
13. Eugenie Samuel Reich. Tevatron's legacy set to disappear. *Nature*, 474:16–17, 2011.
14. J. Rothenberg. Ensuring the longevity of digital documents. *Scientific American*, 272, 1995.
15. C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423 and and 623–656, July and October 1948.
16. Wim Van Grembergen and Steven De Haes. *Enterprise Governance of Information Technology: Achieving Strategic Alignment and Value.* Springer Publishing Company, Incorporated, 1st edition, 2009.
17. Jeannette M. Wing and John Ockerbloom. Respectful type converters. *IEEE TSE*, 26(7):579–593, 2000.