Role Model »Digital Design«

Successful digitalisation and digital transformation require a rethink in software development

Publisher

Bitkom e. V.
German Association for Information Technology, Telecommunications and New Media
Albrechtstraße 10 | 10117 Berlin

Contact

Dr. Frank Termer | Head of Software
P +49 30 27576-23 | f.termer@bitkom.org

Responsible Bitkom Committee

Software Designer

Overall Coordination & Editing

Dr. Kim Lauenroth | adesso AG/IREB e.V.

Graphics & Layout

Katrin Krause | Bitkom

Role Model »Digital Design«

Successful digitalisation and digital transformation require a rethink in software development

Table of contents

Acknowledgements

Software is everywhere: It is being used in cars, planes, mobile phones, household appliances, movies and games. Thus, software is an essential part of the digital transformation. The development of high-quality and safe software is central to all economic areas, e.g. Industry 4.0, healthcare, energy supply, mobility, service industry as well as the public sector. Accordingly, the ability to develop high-quality software is a core competency that determines the future viability of Germany as a business location.

Software development has accelerated dramatically in recent years. Innovation and development take place in ever shorter cycles. The software is no longer developed using only classical procedural models but also with the help of more agile approaches and methods to accommodate for this increased speed with appropriate quality as well as safety. Software development, therefore, needs to combine agility as well as engineering. As a result, software is increasingly being developed by joint cross-functional teams. The existing fragmentation of software development into individual sub-disciplines and thus roles, which is further driven by the current structure of IT training, is becoming an increasingly difficult obstacle to overcome in software development.

The task force »Software Designers« has used this situation as an opportunity as well as a starting point to reflect on new ways and necessities that the software development needs to adopt in the course of the digital transformation. The present document describes the outcome of these considerations, which have emerged from the discourse with numerous companies, academic and research institutions as well as the interested public. We wish to thank the following people for their collaboration in developing the present document. Their dedication and strong personal commitment have made its creation possible:

- Dr. Kim Lauenroth (adesso AG/IREB e.V.)
- Prof. Dr. Karsten Lehn (University of Applied Sciences Hamm-Lippstadt)
- Ulf Schubert (Datev eG)
- Dr. Marcus Trapp (Fraunhofer Institute for Experimental Software Engineering IESE)

Furthermore, we would like to thank the following people who actively contributed to the work of the task force and provided valuable information during the creation of the document:

- Kai Bünseler (City of Dortmund Economic Development)
- Ralf Dehnhof (iSAQB e.V.)
- Thomas Geis (ProContext Consulting GmbH/UXQB e.V.)
- David Gilbert (DB Systel GmbH)
- Dr. Ronald Hartwig (untrouble GmbH)
- Michael Mahlberg (iSAQB e.V.)
- Dr. Sabine Radomski (University of Applied Sciences Leipzig (HfTL))
- Dr. Frank Simon (Zurich Versicherung/GTB e.V.)

1 Introduction and Motivation

1 Introduction and Motivation

1.1. Digital transformation and software development

Keywords like »digitalisation« and »digital transformation« describe the change in society and the economy associated with the gradual introduction of digital technologies – such as the Internet, computers, smartphones, etc. Examples of this change range from Internet shopping, e.g. Amazon, eBay, to the digitalisation of media – such as music, films, books, newspapers – but also the integration of social networks, e.g. Facebook, Twitter, XING, in our daily lives right up to entirely new business models on a digital basis, e.g. Airbnb, Uber, Spotify. Around the world, major conferences are being held to address the subject. In 2017, led by Germany's initiative, the G20 even held a conference on the subject of digitalisation.

If software development is considered as a discipline, then it plays a crucial role in digitalisation: Without software, digitalisation cannot take place. It is astonishing, however, that the topic of »digitalisation« or »digital transformation« hardly seems to play any role in the software industry and appears to be seen as nothing more than hype. Relevant software conferences – industry and science – are dominated by topics such as »cloud«, »big data«, »microservices«, »internet of things« or »agile development«. To put it bluntly, one takes care of oneself; namely, the focus is on new technologies, tools or development approaches. The unanimous opinion from a wide range of experts seems to be that the subject of »digitalisation« has always been a part of the software industry since software has always been in development. This perspective is strongly technology-centred. Above all, there appears to be a lack of discussion about the effects of digitalisation on people, companies and society. Admittedly, this perspective is hugely curtailed and simplified, yet the picture emerges that in recent years there has been a widening gap between the world of those who use digitalisation and the world of those who implement digitalisation.

As a result of this gap, a part of software development – especially in larger companies that are not traditionally associated with technology – has become steadily more accustomed to its role as a passive implementer, with individuals on both sides increasingly identifying with this role model – customers, clients, users or development. This somewhat harsh diagnosis is presumably related to the history of software development. Professional software development has been around since the 1950s/1960s. The business world was undoubtedly a key driver for the software industry. Banks and insurance companies recognised computers and software as instruments with the potential to make their existing business processes more efficient. More specifically, established processes – that were previously realised with the help of paper and humans – were transferred to the computer. One could even describe this as an early form of digitalisation.

In this situation, software development was faced with the challenge of first understanding the existing processes and issues and then translating them appropriately into high-quality software. Two challenges dominated software development for a long time (Boehm, 2006):

1. The small but growing performance of existing hardware: limited storage capacity, processing speed, network transmission capacity and availability of high-performance end-user devices, amongst other things.
2. The development of high-quality software, namely software that is as error-free as possible and meets the customer´s specifications.

The first challenge was not directly related to software development but had a massive impact on the applicability and distribution of software. Without any acceptable hardware requirements, even the best software could not generate any real usefulness or attract new users. However, the technological progress has seen this situation improve continuously. Following the second challenge, software development was increasingly perceived as an engineering discipline, and an attempt was made to achieve better quality through engineering. This development coined the term software engineering. The often misunderstood waterfall model emerged as early as the 1970s; this model formed the basis for many approaches that structure development work in the same way as an implementation process (Royce, 1970). Since the mid-1990s, so-called agile development and human-centred design have become increasingly important and established themselves as alternative models for the development of software.

1.2. Understanding software development as realisation discipline

It can be concluded that software development has grown in a world, in which established processes and procedures have been translated to software, and where the technology has always lagged behind the expectations of the customers or users. This development culminated in the so-called dot-com bubble: the expectations that were placed on IT and software soared to unprecedented heights. The bubble did burst as it became clear to everyone involved that these expectations could not be met for the foreseeable future.

The situation since the burst of the dot-com bubble has changed, almost imperceptibly. The technological performance of hardware and software has reached a level that is more than sufficient for most applications. Storage capacity and computing power are available in abundance, as are inexpensive devices for all customer groups. This is accompanied by the establishment of the Internet as a powerful communication medium for the masses that is used on both stationary and especially portable end-user devices. As a result, this change has seen technical capabilities significantly outgrow expectations towards software in recent years. It could even be argued that software today can do much more than the customer expects. Nonetheless, in many areas, software development is still a long way from catching up with technological possibilities. This is aggravated by the fact that companies still struggle to conceive and formulate new processes, services or business models that would not be possible without software. However, this is indispensable today, because it is no longer just a matter of digitising existing processes. This phenomenon is also commonly summarised under the heading digital transformation.

This transition fundamentally changes the range of tasks for software development. The implicit assumption that the customer side of software development determines what it has to develop no longer applies to many undertakings or does so only in part. Instead, the customer can formulate only vaguely, at best, what users want to do with the system. Unfortunately, the description of one's ideas falls back on analogies too often – »It has to be like Google«. This change in circumstances is met by a kind of software development that is entirely focused on obtaining as accurate descriptions as possible of what is to be created.

This has inevitably led to problems in software development and thus created suboptimal results and products. The dissemination of the previously mentioned agile development as a new procedural model was probably made possible or at least strongly favoured by this situation. Current standards (Bourque and Fairley, 2014) describe agile development as one of the central methods for developing software.

The following core ideas of agile development are relevant for the previous considerations: intensive regular communication between the development and the customer side, a central point of contact – depending on the model, for example as »onsite customer« or »product owner« – and fast feedback cycles based on actual software. These ideas allowed management and customers of software development to maintain their existing worldview – »someone else defines what needs to be done« – without the need for an accurate understanding of the software to be implemented on the customer side. Intensive communication and regular feedback may now enable the development of a more concrete understanding of the emerging software on both sides. This form of development also reaches its limits as soon as the customer side lacks a precise understanding of the use of the planned software. In such situations, the iterative approach cannot converge towards an accepted solution.

1.3. The lack of design skills in software development

Our considerations started with digitalisation/digital transformation and the statement that software development as a discipline does not appear to play an active role in this topic, even though the software is the central element of digitalisation. Instead, there appears to be a deep gap between the two worlds. It is presumed that the reason for this is related to the history of the discipline. In summary, this gap creates a lack of design skills and causes relevant decisions in software design to be made somewhat randomly.

The term »shaping« or »design« is defined quite differently in the literature (Erlhoff and Marshall, 2008). In this document, the terms »shaping« and »design« are used interchangeably and understood as follows: design describes all aspects of the system, software products and services that an end user may experience. Experience thereby refers to the form, e.g. the interface or device used, the function, i.e. the capabilities or purpose that the software serves in context, emotional characteristics – such as aesthetics –, and qualitative characteristics, e.g. responsiveness. Of course, these aspects are primarily determined by the technological capabilities of software as technology provides these capabilities. Designing itself refers to the creative process in which a thing – referring to a material object, structure, process, situation or thought – is changed by the work of the creator. It is thus created, modified or developed, thereby receiving or adopting a particular form or appearance (Davis, 2003).

In the context of digitalisation, a lack of structured design skills entails that software development cannot realise its real strength – namely the knowledge of how the efficiency, possibilities and limits of software may improve existing businesses or create new businesses – sufficiently or at a far too late stage of the project. Thus, many projects might not reach their potential or fail altogether.

In light of this development, the task force »Software Designers« has formed to examine – within Bitkom and in cooperation with the working groups focused on software – whether the previously outlined lack of design skills in software engineering is still up-to-date and how software engineering should position itself in the future with reference to design skills. Basic positions can be: become more active as a software designer or remain somewhat passive as a software implementer.

1.4. Structure of the present report

This report summarises the findings of the task force »Software Designers« and is structured as follows. The first step involves a consideration of the state of practice in software development (SD) and the current training situation of design in software development. This is followed by a report on the results of an exchange with other design disciplines with a focus on design in software development. The conclusion of these results entails that the »digital designer« is derived, defined and described as a new cross-functional and idealised role model. This idealised role model serves as a bridge between existing disciplines of software engineering, software management and the customer, as well as a perspective on the future development. Finally, an outlook on further activities and next steps on the subject of »digital designer« will be provided. Throughout the text, you will also learn how digital design is related to pi.

2 Current state of design in software engineering – perceived starting situation

2 Current state of design in software engineering – perceived starting situation

Other established disciplines have developed independent role models or even entire fields that are dedicated to design. Prominent examples here are the architect from the construction industry and the industrial designer from product development. Central features of these two roles are that they form the link between the customer/user and manufacturing/realisation and bear the main responsibility for the final result – a building or a product. These role models emerged because the activities in the respective field – construction/product development – had reached a level of complexity that was difficult to manage by a single person.

Parallels can also be observed in software development. The complexity of software that is regarded as »matter« led to the development of various specialised fields. For example, the current IEEE Software Engineering Body of Knowledge lists 15 areas of expertise for software development. Particularly noteworthy is the section on »software design« in SWEBOK V3.0 (Bourque and Fairley, 2014, p. 2-2):

Software design is generally considered a two-step process:

- Architectural design – also referred to as high-level design and top-level design – describes how software is organized into components.
- Detailed design describes the desired behaviour of these components.

This definition is directed primarily inward, i.e. at the software to be developed and its structure. Software architecture and software architects are often mentioned in this context. At this stage, the software architect cannot be compared to the role of the architect in the construction industry.

The SWEBOK V3.0 guide also considers the design of the user interface – user interface design – a part of software engineering. Other disciplines have evolved with a focus on designing the user interface as a sub-aspect of software, e.g. user experience, usability engineering, service design, information architecture and interaction design. These disciplines can be regarded as established, but are not as ubiquitous or leading in software development as architects within the building trade or industrial designers in product development. Presumably, there was no need to develop such a holistic and responsible design role for software so far since this task was taken over by the customer side (Chapter 1).

3 Analysis of the current training situation in software engineering with a view to design skills

3 Analysis of the current training situation in software engineering with a view to design skills

A statistic on degree programs in computer science with a significant share of teachings covering design disciplines was compiled based on the publicly accessible database »Hochschulkompass« (↗https://www.hochschulkompass.de/en/study-in-germany.html). The aim was to answer the question which degree programs already provide extensive design skills[1] in connection with training in software development. This database was chosen as a good foundation from a selection of other portals with information on degree programs, as it is part of the official presence of the Hochschulrektorenkonferenz (HRK) [Higher Education Conference for Principles], which is composed of 268 public and officially recognised universities from all over Germany. In total, about 94 percent of students enrolled in Germany are represented by these institutions (Hochschulrektorenkonferenz, 2017a).

Studies in computer science	Number	Share
..., all, undergraduate	672	100%
... with design, undergraduate	42	6.3%
..., all, post-graduate	504	100%
... with design, post-graduate	21	4.2%
..., all, undergraduate and post-graduate	1176	100%
... with design, undergraduate and post-graduate	63	5.4%
Data based on ↗www.hochschulkompass.de, status: 22.12.2016		

Table 1: Degree Programs in computer science and degree programs in computer science with a significant share of design disciplines in Germany

Table 1 shows that the proportion of programs with a significant share of design disciplines is 5.4%. Courses of study (undergraduate) with a Bachelor or Diploma (UAS) account for a share of 6.3% whereas courses that lead to a Master's degree (post-graduate) account for a share of 4.2%. These absolute numbers appear to be low and thus provide a first impression of the current situation.

1 In the context of this survey, »design« and »shaping« along with »design discipline« and »shaping discipline« were used interchangeably (Erlhoff and Marshall, 2008, p. 176 et seq.)

To be able to make a quantitative statement as to which extent there is a need for improvement, other factors have to be considered, and a reference or target value has to be determined. Other factors include the number of graduates per academic year and degree program as well as the number of graduates that start working in the software industry.

A reference value can be determined from surveys in companies.

Degree programs in computer science with and without design disciplines in Germany

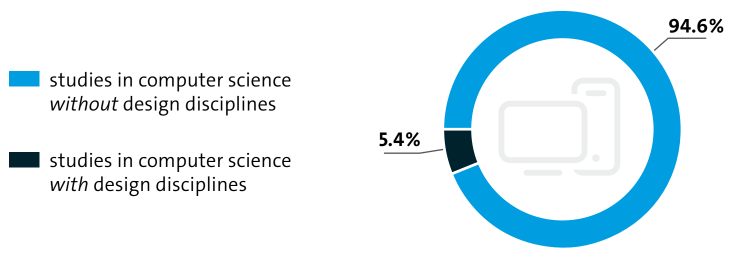


Figure 1: Degree programs in computer science with and without design disciplines in Germany (based on ↗www.hochschulkompass.de, status 22.12.2016)

Figure 1 compares the proportion of degree programs in computer science in Germany with and without a significant proportion of design. Degree programs in computer science that teach design skills make up a share of 5.4% of all computer science courses in Germany. This consideration not only includes degree programs in software engineering but also general courses of study in computer science since graduates of general computer science courses may also work in software development. The statement of the result is thus more general; however, the proportion of design in the smaller group of software engineering courses could be higher.

A course of study was only attributed an »important creative element« if the keyword search of the used database showed that a computer science program referred to a design discipline. In this case, it can be assumed that the course will cover the relevant design discipline both fundamentally and in great detail. It does not include courses of study that teach a small proportion of design disciplines so that this share was not significant enough to be included in the list of keywords for the course description and classification.

4 Design meets IT – a short review of the interplay between IT and design

4 Design meets IT – a short review of the interplay between IT and design

To initiate the exchange between design and IT roles, the Bitkom event ↗»Design meets IT« (only available in German) took place on 4 May 2017, at the Dortmunder U. The event focused on the question »What can the IT industry learn from designers«. A total of 219 participants from the fields of product/software development, UX and design discussed possible answers based on several keynote speeches together with the speakers Andrea Augsten, Andreas Enslin, Marc Hassenzahl, Frank Jacob, Uwe Kemker and Bernd Wiesenauer.

4.1. Keynote presentations

The keynote presentations were intended to offer an insight into designers' working world and their way of thinking. Furthermore, they also demonstrated how designers create value in companies:

- Andrea Augsten (Founder of design:transfer) emphasised the human-centred design of products, services and organisations in her presentation. She provided a critical and exploratory account of current phenomena and presented impressions from practice.
- Andreas Enslin (Head of the Designcenter, Miele & Cie. KG) focused on good product design using examples from his daily practice. He described the interdisciplinary cooperation between design, business and technology as a critical success factor to making useful, relevant and successful products.
- Marc Hassenzahl (Professor of »Experience & Interaction Design« at the University of Siegen) used his presentation on well-being and experience to appeal for the reconciliation of »having an experience« and »using technology« in the design process. The aim is to not only automate things or use techniques because it is feasible but always to question which needs are to be satisfied and to ensure that the experience is not too shallow and thus meaningless.
- Frank Jacob (Professor of Interface Design at the Muthesius Academy of Fine Arts in Kiel and owner of HID Human Interface Design GmbH) devoted his presentation to the role and importance of designing in the daily work of designers. From his point of view, IT and design may spur innovation if they work together in a tool chain and build on the respective work results instead of communicating these through mutual documentation and explanations.
- Uwe Kemker (Head of Industrial Design, Vorwerk & Co. KG) introduced the design process at Vorwerk and addressed the challenges that designers face when they want to achieve customer value through good product design.
- Bernd Wiesenauer (Senior Manager User Experience, Robert Bosch GmbH) used his lecture to report on the experiences he has had with Bosch during the introduction of user experience and design thinking in the context of the digital transformation.

4.2. Panel discussion

The ensuing discussion with the panel and the audience made it evident that the demarcations of the disciplines within IT and design need to be blurred further to achieve better and more successful outcomes in the fast-paced IT world. A provocative comment by an audience member summarised the status quo as follows:

»To be honest, the solutions that emerge when design and development do not communicate with one another are quite frankly shit.«

Although the specialisations of the IT and design disciplines are necessary to ensure that the relevant expertise is available for the individual projects, all representatives of their field must have a shared understanding of the other disciplines involved. The creation of common knowledge is most challenging for creators of the IT curriculum. A comment by an audience member summarised this as follows:

»Talk to your IT people. They do not bite; they just want to play.«

On the one hand, the presentations and discussions illustrated that efforts that seek to bridge this gap are currently underway and that they have already achieved first successes. On the other hand, they also emphasised clear deficits in the training of software developers. One possible way to address this would be to focus more strongly on degree programs in the form of projects. Furthermore, more Master's degree programs in design should be open to students from other disciplines.

The industry must also contribute to bringing the disciplines closer together by decreasing their search for highly specialised experts in IT and design. If the advertisement of profiles with »blurred« margins that seek competency in both disciplines and act as a hybrid between design and IT were to increase, the training would self-adjust.

A detailed report about the event and the keynote presentations can be read on the User Experience Blog by Ulf Schubert (Datev eG) (↗http://www.user-experience-blog.de/tag/dmit17/) (only available in German). Excerpts of the presentation slides can be found at ↗https://www.bitkom.org/designmeetsit/ (only available in German).

5 Conclusion – The »Digital Designer« as design role

5 Conclusion – The »Digital Designer« as design role

The output of the task force illustrates that the lack of design, or rather, the fragmentation of the design skills in software development constitute a severe issue and that the training situation in Germany is underdeveloped for design in software development, i.e. keyword 5%. Considering the importance of software for contemporary society and economy – keyword »digital transformation« –, the formation of a holistic and well-integrated design role in software development –analogous to that of an architect or industrial designer – seems to be the next logical step in this evolution. The »digital designer« is supposed to be this next step forward.

5.1. Derivation of the term »Digital Designer«

The term »digital designer« might be disconcerting for individuals that come from a technical background. The word digital refers to the use of binary values to represent information. However, the term »digital designer« seems more appropriate once you look at the word »digitalisation« and its changed meaning in society (Chapter 1). The term »software designer« could have been a conceivable alternative, but it falls short because in the context of digitalisation it is not only software that is created but also the environment, e.g. business processes.

An example of successful digital design is the company Airbnb along with their website. Airbnb is a platform that allows private individuals to let or rent accommodation. In doing so, Airbnb serves as an intermediary and handles the payment process. This has created an entirely new economic system for the provision of private housing as an alternative to the classic hotel business. In this sense, Airbnb not only designed a software but a new digital business model.

The precise definition is based on the definition of the industrial designer: »Digital Designers« design and optimise digital products, systems and services. They consider the interplay between user requests and requirements, the economic framework conditions as well as the technical possibilities. Digital designers lead development processes through sketches, models, specifications as well as prototypes. Together with management, marketing, development and software operations, they work in multidisciplinary groups.

Foresight is essential. The digital designer can think beyond the current state of possibilities and develop new concepts as well as applications that did not seem possible previously. At the same time, he is aware of the – current – limitations. The design process is prioritised over planning procedure.

5.2. Digital Designer as the idealised image and bridgehead

The digital designer needs to be understood as an idealised role. Just as there is no industrial designer or architect within the building trade, there will be no precisely defined project role for the digital designer. In practice, the responsibilities will indeed be distributed depending on the context.

Nevertheless, given the current situation in software development, this idealised role is of particular importance. The described lack of design skills, as well as the fragmented design competence, can only be overcome sensibly if interdisciplinary activity is promoted. Stable connections can only form if a concise idealised role is defined, which other individuals, in turn, can use as practical orientation. The idealised role is supposed to polarise and evoke a discourse on design and software development, within but also outside the software industry.

5.3. Differentiation towards other idealised role models

In addition to the digital designer, the software/system engineer and the software manager can both be defined as two further idealised role models in software development. The combination results in a triad with the following clearly defined responsibilities:

- The **digital designer** is responsible for design the perceptible aspects of digital products, systems and services for customers. Amongst others, these are functions, user interfaces, quality aspects, such as speed, but also the consequences for environment and surroundings. Summarised in a few words: The digital designer is responsible for everything that customers/users can experience.
- The **software/system engineer** is responsible for design and realising the technical aspects of digital products, systems and services. The term design is also used for engineers, since digital technology, in particular software and systems, is relatively complex based on its reliance on a multitude of technological possibilities and applications (Glass, 2006). The engineer is responsible for everything that is »under the hood«.
- The **software manager** is responsible for the process design as well as the economical implementation. The term design is also used since the implementation of digital products, systems as well as services tend to come with a more complex realisation process than traditional consumer goods. Therefore, processes need to be designed and managed proactively to be successful. The manager is responsible for processes, time management as well as budget.

The description of all the role models illustrates that digitalisation can only be successful if all three roles work together and take their responsibilities seriously. If a role predominates or a role model is neglected, then an imbalance and a suboptimal result arise. These three role models cannot only be seen as bridgeheads within software development but can also be used in particular in the external presentation.

One example for this is the Bitkom campaign page erlebe-IT for software jobs (↗https://www.erlebe-it.de/software-berufe/) (only available in German). This campaign shows that the IT industry is already utilising a broad range of occupations, requiring various talents as well as skills.

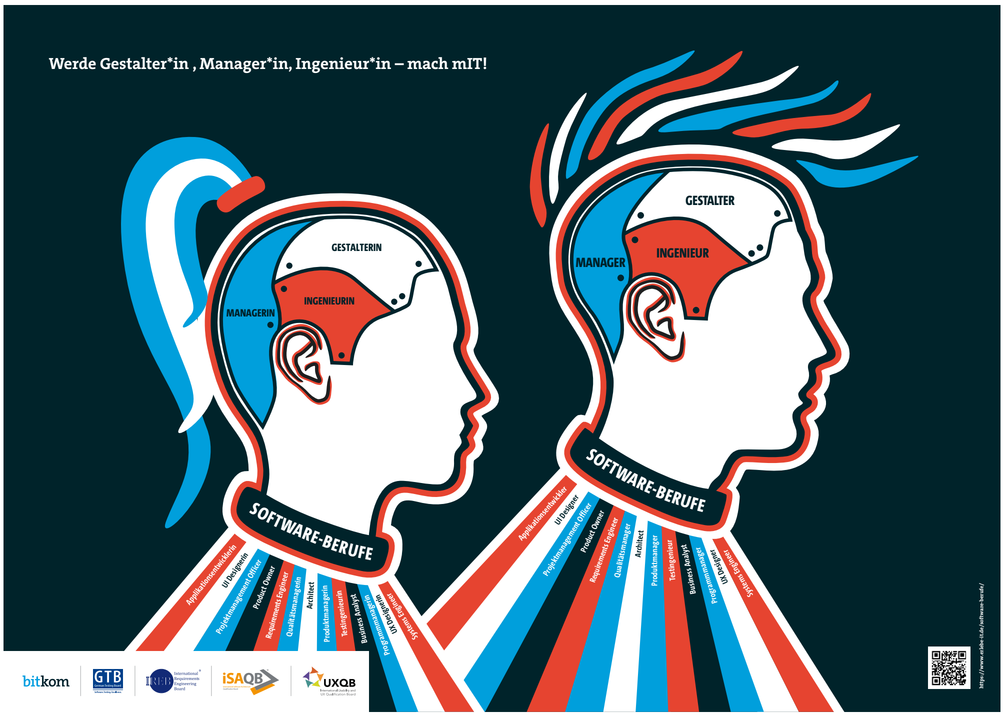
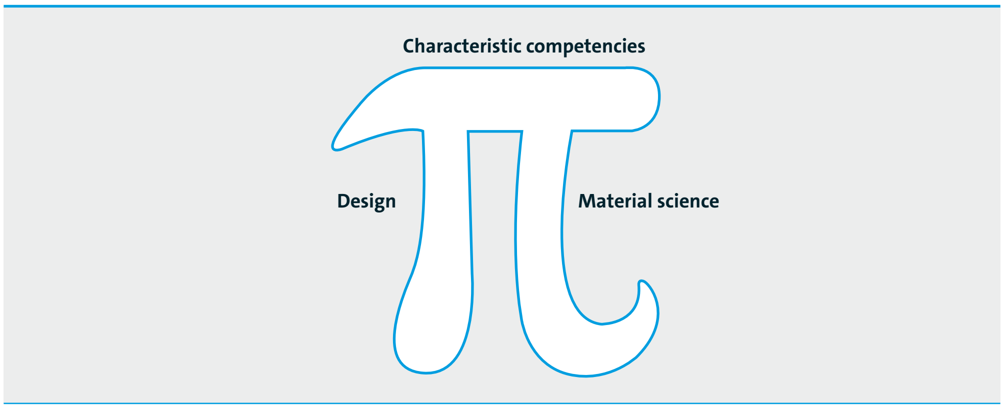


Image 1: Become designer, manager, engineer – participate! (↗https://www.erlebe-it.de/software-berufe/)

5.4. Competency profile »Digital Designer«

The characterisation of the digital designer requires the consideration of the interplay between user requirements, economic feasibility as well as the technical possibilities of one role. The combined perspective of these three aspects places the digital designer at the centre of every development project and allows for mediation between the three aspects to create sustainable, thriving products, systems and services. The definition of the digital designer results in a broad and demanding competence profile as well as a broad characteristic competence field, analogous to the architect within the building trade and the industrial designer. The role of the digital designer extends the role of the »user experience designer«, which is already established in some companies. The competency profile of the digital designer is rooted in two areas, design and material science. In comparison to the »user experience designer«, however, the digital designer faces more technical aspects within the focus area of »materials science«. The digital designer relies more on characteristic competencies in regard to methods as well as procedures for developing software.

These two main focus points along with the characteristic competency result in the so called pi-shaped profile. Thus, the pi symbol on the cover. This is contrasted by the known t-shaped profiles that have one focus and one characteristic competency. As previously discussed in this report, this type of competency pattern is not sufficient for the challenges of digitalisation as well as digital transformation. Two main focus points as pillars as well as one broadly defined characteristic profile are required:



Focus point 1 – Design

- Working with requirements – requirements engineering
- Construction of user interfaces – usability engineering and interaction design
- Essential approaches to design
- Development of new concepts
- Exploration capabilities, ability to conduct person-centred ethnographic field research

Focus point 2 – Material science

- Knowledge of the possibilities as well as limitations of software and hardware
- Knowledge of the possibilities as well as limitations of algorithms
- Types of systems: IT systems, embedded systems
- Types of end-user devices: desktop, notebook, tablet, smartphone and so forth
- Types of interactions: keyboard, mouse, touch, voice, gesture and so forth
- Knowledge of the essential software architecture
- Knowledge of form and colour schemes

Characteristic competence

- Knowledge of methodologies and approaches for the management of development projects:
  - Project management
  - Agile development
  - V-model
  - and so forth
- Knowledge of methodologies and approaches for the development of software:
  - Quality assurance/testing
  - Configuration management
  - Product processes, knowing the entire journey
- Economic aspects concerning the design/development of software
  - Ability to calculate costs
  - Business models for software – pay per use, and so forth
  - Business processes
  - Brand creation
  - Knowledge of companies who create software-based products
- Ability to work on interdisciplinary projects
- Psychological foundations for the realisation of software based on the user and manufacturer side

6 Summary and next steps

6 Summary and next steps

The illustrated and idealised role of the »digital designer« represents the next step in the advancement of software development as a discipline and is thus able to take a central part in the context of digitalisation. Technological competence in software development is essential for successful digitalisation as well as digital transformation but by no means the only skill required. The software industry in Germany, however, concentrates too much on this technological competence and loses its competitive edge by neglecting relevant design skills: the ability to design sustainable, technically excellent and successful products as well as services.

The output of the task force shows that the establishment of a digital designer is a long-term endeavour:

- There is a need to build bridges within the individual software development disciplines to create a mutual understanding for the relevance of software design but also to establish the idealised role of the digital designer within the software development industry. This effort requires extensive cooperation between appropriate professional bodies and organisations.
- Practice shows that some parts of the ideal image of the digital designer are established already. The successful establishment can already be witnessed in user experience, e.g. user experience design roles. However, the persons involved tend to cover only some of the required competencies. Therefore, training programs, for example in cooperation with relevant training associations, need to be developed to complete the training of such persons.
- The current state of educational training shows that there is also a strong need for change in academic education. University curricula need to be established to enable a sound education as a digital designer. The long-term goal is to establish the occupational profile of the digital designer in the same way as the industrial designer or the architect in the building trade. This goal requires political engagement as well as cooperation with relevant academic associations, e.g. Gesellschaft für Informatik (German Informatics Society).
- Establishing the digital designer in the industry requires a change in the mind-set of corporate governance so that the digital designer can be established as a leading role in companies and projects.

At least as significant as the establishment itself is the change of the external perspective on the digital designer role. Until now, software development has only been considered as realisation. The digital designer has the potential to affect this outside perspective positively and thus make software development as a discipline more complete and close a critical gap in design skills.

The digital designer is undoubtedly an ideal image and must be understood as such. Nevertheless, the definition of such an ideal image is indispensable for a change in software development. The same applies, for example, to architecture, where no structure is built without an architect. Thus, every potential realisation, related to digitalisation or digital transformation, should be done by a qualified digital designer.

7 Literature

7 Literature

Pierre Bourque, Richard E. (Dick) Fairley (Hg.) (2014). Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK). A Project of the IEEE Computer Society. IEEE.

Alan M. Davis (2003). System Phenotypes. IEEE Software, July/August 2003, 54-56.

Barry Boehm (2006). A View of 20th and 21st Century Software Engineering. 28th International conference on Software engineering (ICSE '06), 2006, 12-29.

Michael Erlhoff, Tim Marshall (Hg.) (2008). Design Dictionary: Perspectives on Design Terminology. Basel, Boston, Berlin: Birkhäuser.

Robert L. Glass (2006). Software Creativity 2.0. developer.* Books.

Hochschulrektorenkonferenz (2017). Aufgaben und Struktur. ↗https://www.hrk.de/hrk/aufgaben-und-struktur/. Requested 9.6.2017, 11:00h.

Winston Royce (1970). Managing the development of large software systems. WESCON, August 1970,1-9.

Bitkom represents more than 2,500 companies of the digital economy, including 1,700 direct members. Through IT- and communication services alone, our members generate a domestic annual turnover of 190 billion Euros, including 50 billion Euros in exports. The members of Bitkom employ more than 2 million people in Germany. Among these members are 1,000 small and medium-sized businesses, over 400 start-ups and almost all global players. They offer a wide range of software technologies, IT-services, and telecommunications or internet services, produce hardware and consumer electronics, operate in the digital media sector or are in other ways affiliated with the digital economy. 80 percent of the members' headquarters are located in Germany with an additional 8 percent both in the EU and the USA, as well as 4 percent in other regions of the world. Bitkom promotes the digital transformation of the German economy, as well as of German society at large, enabling citizens to benefit from digitalisation. A strong European digital policy and a fully integrated digital single market are at the heart of Bitkom's concerns, as well as establishing Germany as a key driver of digital change in Europe and globally.

Federal Association for Information Technology, Telecommunications and New Media

Albrechtstraße 10
10117 Berlin
P 030 27576-0
F 030 27576-400
bitkom@bitkom.org
www.bitkom.org