

20.04.2018 | Forschungsdesign 4.0, Dresden

Programming Historians

UNVERZICHTBARE KOMPETENZ, METHODISCHES
PLUS ODER NICHT ZWINGENDE EIGENSCHAFT
ZUKÜNFTIGER HISTORIKERINNEN?

Torsten Schrade, Akademie der Wissenschaften und der Literatur & Hochschule Mainz
 @digicademy, @_mainzed |  digicademy | CC-BY 4.0

GLIEDERUNG

1. Einführung und Überblick

L'histoire et l'ordinateur après 50 ans

2. "Weniger schlecht programmieren"

Vom einfachen Skripting zu professioneller Forschungssoftware

3. Naming things

Domain Driven Design als Modellierungsmethode in der
geschichtswissenschaftlichen Softwareentwicklung

4. Fazit

Geschichtswissenschaften im Spannungsfeld zwischen Software
Engineering und digitaler Quellenkritik

FRAGESTELLUNGEN

- ▶ Mit den Auswirkungen der **digitalen Transformation** in allen Bereichen unserer Gesellschaft unterliegen auch die Geschichtswissenschaften einem starken Wandel. Traditionelle **Ausbildungswege verändern sich**, klassische Forschungsfragen werden unter **Einbeziehung informatischer Methoden** neu gestellt.
- ▶ Der **Aufbau digitaler Kompetenzen** von Lernenden, Lehrenden und Forschenden rückt zunehmend in den Fokus.
- ▶ Müssen zukünftige **HistorikerInnen** für ihre Forschungsarbeit nun zwingend **programmieren lernen**? Oder wird uns diese Wahrnehmung durch die digitale Transformation "einprogrammiert"?

- ▶ Wo liegen die **Potentiale**, wo die **Grenzen** eines Kompetenzaufbaus für HistorikerInnen im Bereich der Softwareentwicklung und Programmierung?

01

EINFÜHRUNG UND ÜBERBLICK

L'historien et l'ordinateur après 50 ans

THE PROGRAMMING HISTORIAN

Kollaborative Lernplattform für HistorikerInnen (2008 - 2018)

"In 2008, William J. Turkel and Alan MacEachern introduced the project to the world by writing a series of tutorials **aimed at everyday working historians looking to build their technical skills.** [...] A decade on, and the project has attracted more than **1.3-million visitors**, has included a **team of 24 different scholars in 6 countries**. Contributions have come from **63 authors and 97 different peer reviewers** who have published **102 tutorials in 2 languages**. Each of those tutorials is about the length of a journal article, making the project a **substantial piece of collective writing**. [Adam Crymble, A Decade of Programming Historians, 23.03.2018](#)

THE PROGRAMMING HISTORIAN

Kollaborative Lernplattform für HistorikerInnen (2008 - 2018)

The Programming Historian

ABOUT ▾ CONTRIBUTE ▾ LESSONS BLOG [español]



The Programming Historian

We publish novice-friendly, peer-reviewed tutorials that help humanists learn a wide range of digital tools, techniques, and workflows to facilitate research and teaching. We are committed to fostering a diverse and inclusive community of editors, writers, and readers.

The Programming Historian ...

- ▶ ... is committed to open source values.
- ▶ ... is committed to diversity
- ▶ ... is a volunteer-driven project.
- ▶ All submissions [...] are published under a Creative Commons 'BY' license.

"We do not solicit reviews to judge whether a tutorial is 'good enough' to be

published. Rather, we consider the review process an integral component of a collaborative, productive, and sustainable effort for scholars to teach and learn from each other."

**The Programming Historian (ISSN: 2397-2068),
URL: <https://programminghistorian.org/>, CC-BY 4.0**

THE PROGRAMMING HISTORIAN

Überblick und Analyse des Tutorienangebotes

The screenshot shows the homepage of 'The Programming Historian'. At the top, there's a dark header bar with the title 'The Programming Historian' and a menu icon. Below it is a large white section titled 'Lesson Index'. A text block explains that lessons are organized by research phases and general topics, with a link to contact if something is missing. Below this are several buttons for filtering lessons: 'ACQUIRE (10)', 'TRANSFORM (29)', 'ANALYZE (10)', 'PRESENT (18)', and 'SUSTAIN (3)'. Further down are more buttons for specific topics: 'APIS (3)', 'PYTHON (18)', 'DATA MANAGEMENT (7)', 'DATA MANIPULATION (16)', 'DISTANT READING (8)', 'SET UP (7)', 'LINKED OPEN DATA (2)', 'MAPPING (9)', 'NETWORK ANALYSIS (4)', 'WEB SCRAPING (6)', and 'DIGITAL PUBLISHING (8)'. At the bottom is a button labeled 'RESET TO SEE ALL LESSONS (70)'.

- ▶ Ein Schwerpunkt liegt (immer noch) in der Datengenerierung, -verarbeitung und -transformation mit Python
- ▶ Mehrere neuere Bereiche der Digital Humanities, die auch von besonderem Interesse für die Geschichtswissenschaften sind, werden abgedeckt (bspw. Netzwerkanalyse, Linked Data), andere fehlen noch (bspw. Machine Learning)
- ▶ Kein direkter Fokus auf Software Engineering, eher auf Skripting

THE PROGRAMMING HISTORIAN

Einschätzungen aus der Community

"The Programming Historian is currently **the best one-stop-shop for technical skill-building in the digital humanities**. But this comes with a caveat: it is heavily slanted towards a particular kind of technical skill-building, one bound up within a tradition of humanities computing and tech culture. [...] [It] **can feel daunting for people who are not already familiar with humanities computing or tech culture**. [...] Lesson titles such as 'Transliterating non-ASCII Characters with Python', 'Supervised Classification with a Naive Bayesian', or even the tutorial I followed for this review, 'Sustainable Authorship in Plain Text Using Pandoc and Markdown', **assume an existing familiarity with what all those words mean and how they might help you**. ASCII, Python, Bayesian, Pandoc, Markdown: these are part of an insider vocabulary." [Cameron Blevins, Review of The Programming Historian, 2015](#)

»DER HISTORIKER VON MORGEN WIRD PROGRAMMIERER SEIN ODER ES WIRD IHN NICHT MEHR GEBEN«

Zitat aus dem [Blogpost von Mareike König, 03.04.2015](#) zur Kontextualisierung des Beitrags „L'historien et l'ordinateur“ des französischen Mediävisten [Emmanuel Le Roy Ladurie](#) von 1968

Aus den Kommentaren

*"Was insgesamt nicht schaden könnte, statt generell über Historiker und Programmieren zu diskutieren, ist vielleicht eine **sinnvolle Kenntnis von Methodik zu entwickeln**. Sinn macht Programmieren ja erst, wenn man entweder eine eindeutig datengetriebene Frage hat, oder eine Frage so formulieren kann, dass man sie datengetrieben bearbeiten kann."* [Kommentar von Daniel Stange, 06.04.2015](#)

"Die Frage ist für mich, was verloren geht, wenn man das Curriculum von Historikern mit Statistik, Programmierkursen etc. vollpackt." [Kommentar von Eckhart Arnold, 07.04.2015](#)

*"Ohne diese '**code literacy**' wird man nicht Tools nicht ausreichend verstehen und methodisch kritisch hinterfragen können, um sie anschließend anwenden zu können"* [Kommentar von Mareike König, 10.04.2015](#)

PROGRAMMING HISTORIANS

Transfer und Effizienzsteigerung

*"One of the ways historians can learn more about the digital environment is through programming. [...] learning how to code is fun, as it helps us develop the ability to think outside of the box and to build logical connections. **I wouldn't ask everyone to become a programmer, but rather to foster computational thinking.** [...] Historians could read sources in a foreign language using a translator, but need to know the specifics of the relevant syntax to fully understand the content, right? It is the same logic for coding. **If we understand the structure of the internet, the nature of the programs we use,** the values behind a software and the various ways to verify the information on the internet, **we could handle the provided resources more effectively.**"*

Janine Noack, Why historians should learn how to code (at least a bit), 2015

PROGRAMMING HISTORIANS

Erweiterung des Methodenrepertoires, Unabhängigkeit

*"Historians are very thorough when it comes to the integrity of their sources, so it seems logical that they should apply the same rigorous approach to an algorithm that decides what sources are the most visible, or appear at all. [...] **there probably isn't a program out there that does exactly what you want it to do**, this is where programming comes in. **Programming allows you to create the program** or add-on that will enable you to manipulate the data in the way you want to. [...] The biggest reason why we do not see programming used more widely is that it is difficult to learn, and can be very **time consuming**. Also of concern is **which language is worth spending the time** and effort to learn."*

[*mal1661, How can Programming be Useful for Historians?, 27.04.2015*](#)

PROGRAMMING HISTORIANS

Programmieren als Kulturtechnik

"[...] beide Kompetenzen, **die klassische Quellenkritik ebenso wie eine adäquate Medienkompetenz** [sind] im Kurrikulum der Geschichtswissenschaften existentiell bedroht bzw. **hochgradig unterrepräsentiert**. [...] von Programmcode ganz zu schweigen, der in manchen Teilen der Wissenschaft - auch jenseits seiner funktionellen Eigenschaft als **source code - längst zur historischen Quelle geworden ist**. [...] Denn so wie die Kodikologie, die Diplomatik, die Numismatik und die Sphragistik nie ohne den Sachverständ der Kunstgeschichte, Mediävistik, Altertumskunde, Physik oder Chemie auskommen konnten, so wenig wird die Geschichtswissenschaft in Zukunft ohne den **intensiven Austausch** [...] – zwischen Informatik, den Philologien, der Medientheorie und der Philosophie und ihrer digitalen Expertise bestehen."

Forum: Markus Krajewski: Programmieren als Kulturtechnik, in: H-Soz-Kult, 30.11.2015,
[<www.hsozkult.de/debate/id/diskussionen-2901>](http://www.hsozkult.de/debate/id/diskussionen-2901)

02

"WENIGER SCHLECHT PROGRAMMIEREN"

Vom einfachen Skripting zu professioneller Forschungssoftware

RESEARCH SOFTWARE ENGINEERING

Von Programmierkompetenzen zu Forschungssoftware

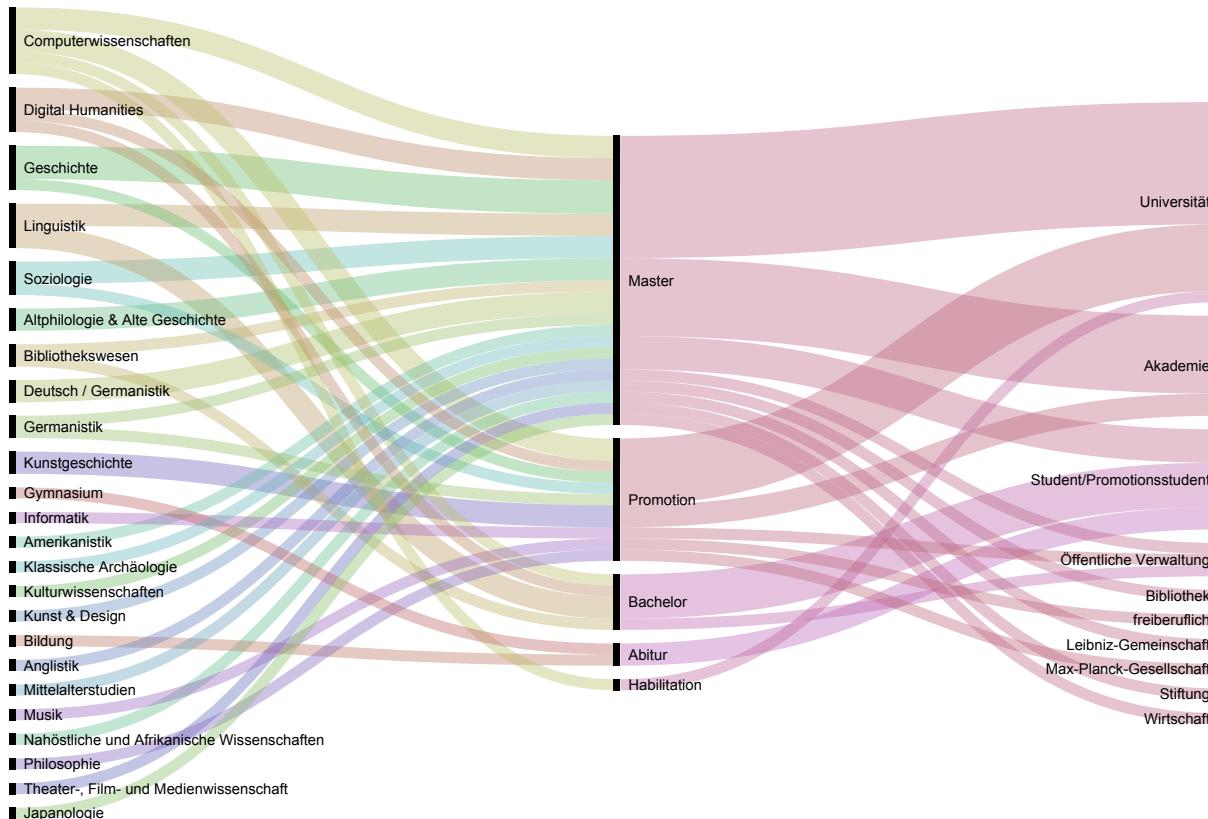


Quelle: Ulrike Henny-Krahmer, Patrick Sahle, Einreichungen zur DHd 2018, URL: <http://dhd-blog.org/?p=9001>

Quelle: Stephan Janosch, Keynote zum DH RSE Workshop auf der DHd 2018, URL: [Figshare](#)

RESEARCH SOFTWARE ENGINEERING

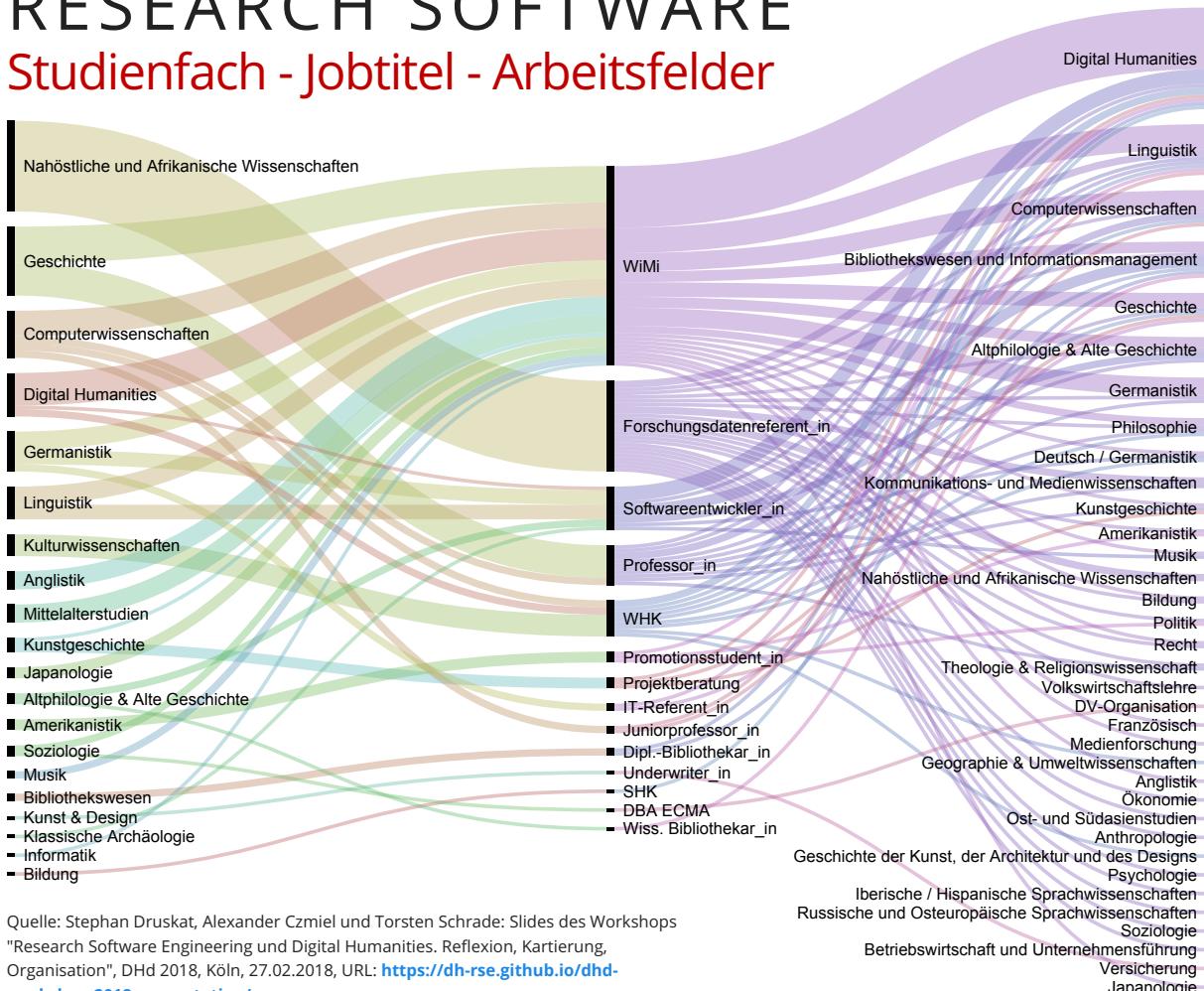
Studienfach - Abschluss - Arbeitsort



Quelle: Stephan Druskat, Alexander Czmiel und Torsten Schrade: Slides des Workshops "Research Software Engineering und Digital Humanities. Reflexion, Kartierung, Organisation", DHd 2018, Köln, 27.02.2018, URL: <https://dh-rse.github.io/dhd-workshop-2018-presentation/>.

RESEARCH SOFTWARE

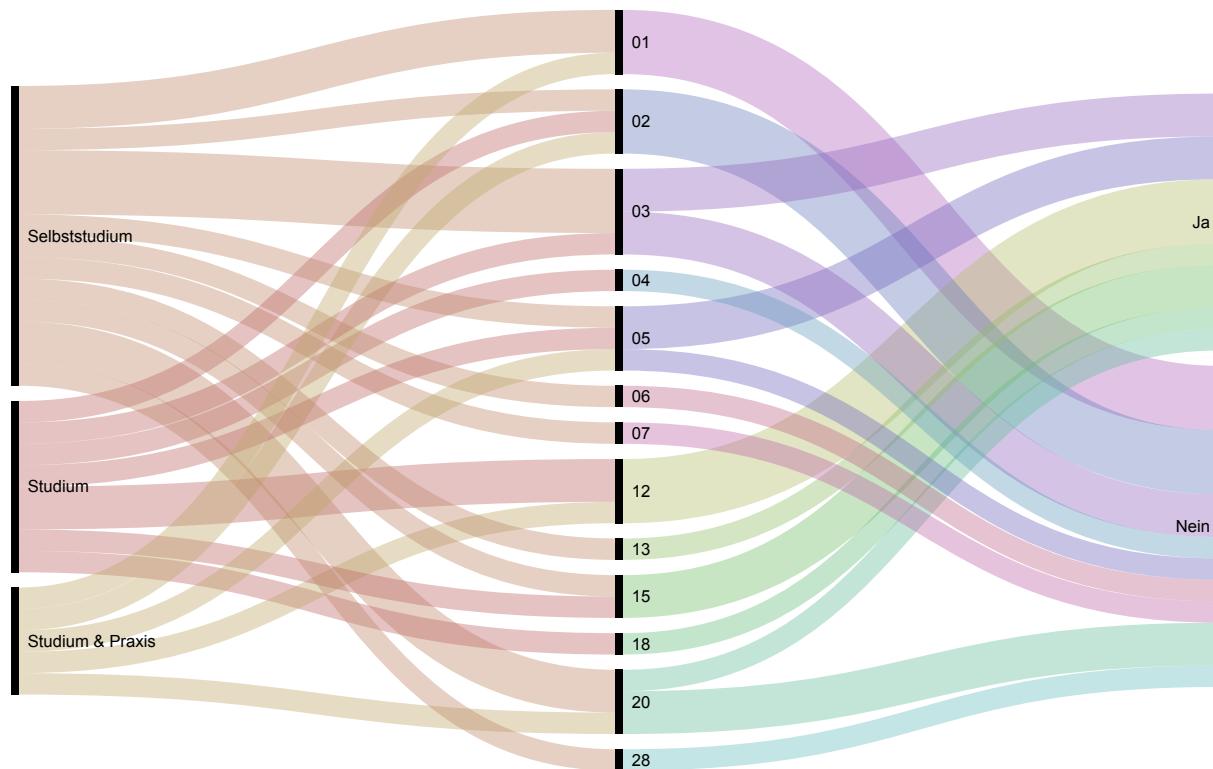
Studienfach - Jobtitel - Arbeitsfelder



Quelle: Stephan Druskat, Alexander Czmiel und Torsten Schrade: Slides des Workshops
 "Research Software Engineering und Digital Humanities. Reflexion, Kartierung,
 Organisation", DH 2018, Köln, 27.02.2018, URL: <https://dh-rse.github.io/dhd-workshop-2018-presentation/>.

RESEARCH SOFTWARE ENGINEERING

Kompetenzerwerb - Erfahrung - Professionalität



Quelle: Stephan Druskat, Alexander Czmiel und Torsten Schrade: Slides des Workshops "Research Software Engineering und Digital Humanities. Reflexion, Kartierung, Organisation", DHd 2018, Köln, 27.02.2018, URL: <https://dh-rse.github.io/dhd-workshop-2018-presentation/>.

RESEARCH SOFTWARE ENGINEERING

Problemstellungen, für die wir ein Grundverständnis benötigen

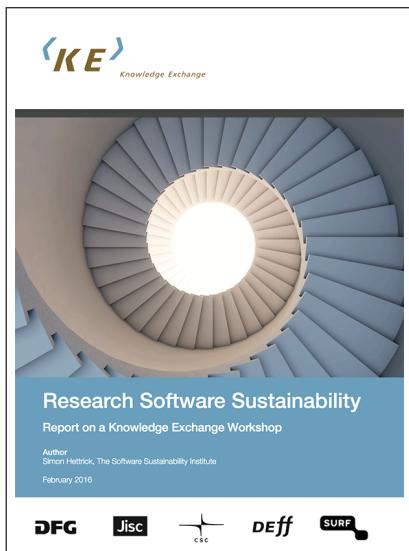
- ▶ **Ausgelaufene Projektfinanzierungen** bedingen, dass der Weiterbetrieb einer Software "irgendwie anders" aufrecht erhalten werden muss
- ▶ EntwicklerInnen stehen einem Projekt nicht mehr zur Verfügung, waren aber die ausschließlichen **Wissensträger** (sog. Busfaktor...)
- ▶ **Veraltete** und/oder nicht mehr wartbare **Software-Infrastruktur** (mit implizitem, undokumentierten "Infrastrukturwissen")
- ▶ Veralteter oder **unverständlicher Programmcode**, der weitergeführt werden muss
- ▶ **Sicherheitslücken** in der Software, die erst im Produktivbetrieb auffallen und einen Weiterbetrieb der Software verhindern
- ▶ **Bugs in der Software**, die erst im Produktivbetrieb auffallen, da vorher keine Softwaretests durchgeführt wurden
- ▶ **Datenverluste oder unerwartetes Verhalten der Software**, da die Anwendung nicht unter Produktiv-Bedingungen (infrastrukturell gesehen) entwickelt wurde
- ▶ **Fehlendes Monitoring** der Forschungssoftware, wodurch Störfälle nicht oder erst spät auffallen



Quelle: Torsten Schrade: Nachhaltige Softwareentwicklung in den Digital Humanities, DHd 2017, Bern, 17.02.2018, URL: <https://digicademy.github.io/2017-dhd-sustainable-software>.

RESEARCH SOFTWARE ENGINEERING

Notwendige Professionalisierung und akademische Leistung



PDF: <http://bit.ly/23jxw7D>

*"Many researchers know how to code, but **few understand the wider set of skills that are needed to develop reliable, reproducible and reusable software.** [...] software engineering should be incorporated [...] at the very start of a research career"* [S.14]

*"We must **raise awareness of the fundamental role of software in research** [...] All stakeholders, from researchers to policymakers, must be included in this awareness raising campaign"* [S.11]

*"Research software should be recognised as a valuable research object in line with the investment it receives and the research it makes possible [...] **Research software should become a citable scientific deliverable** of equivalent value to researchers as that of a publication"* [S.11]

03

NAMING THINGS

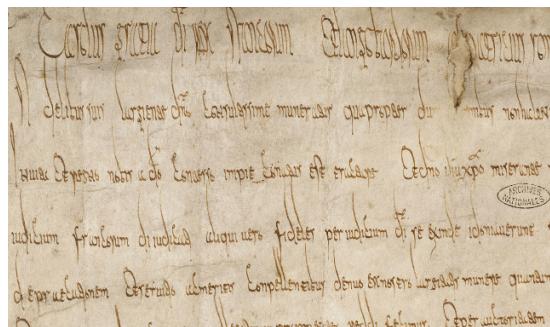
Domain Driven Design als Modellierungsmethode in der geschichtswissenschaftlichen Softwareentwicklung

THERE ARE ONLY TWO HARD THINGS IN
COMPUTER SCIENCE: CACHE
INVALIDATION AND NAMING THINGS.

-- (wahrscheinlich) Phil Karlton (Netscape)

RESEARCH SOFTWARE ENGINEERING

Vom historischen Objekt her gedacht am Beispiel einiger Akademieprojekte



Geschichte: Urkunde Karls des Großen von 797



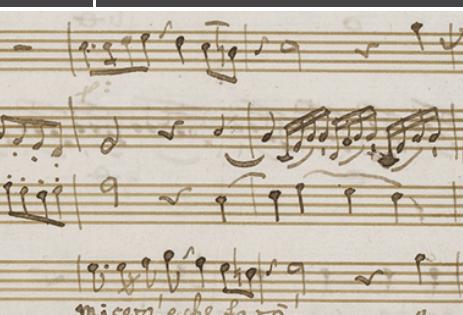
Kunstgeschichte: Glasmalerei



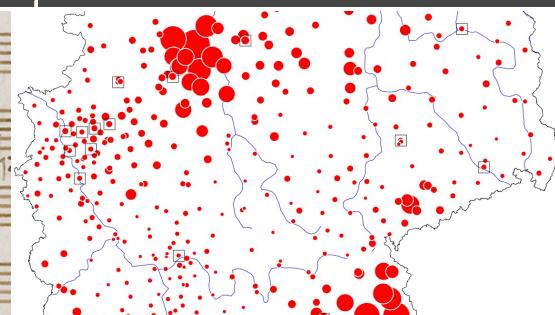
Epigraphik: Inschrift auf dem Mainzer Domportal



Theologie: Religiöse Streitschriften



Musik: Autograph aus Glucks Oper Alceste

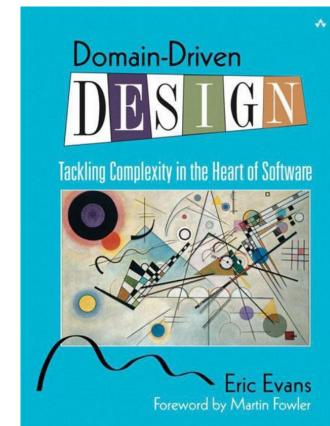


Linguistik: Karte der Verteilung des Namens 'Meier'

METHODENBEISPIEL

Softwaremodellierung mittels Domain Driven Design (DDD)

- ▶ Domain-Driven Design (DDD) ist eine **konzeptionelle Herangehens- und Denkweise für die Modellierung von Software**. Der Ansatz wurde 2003 von Eric Evans in seinem gleichnamigen Buch geprägt. Das Hauptaugenmerk in DDD fällt dabei auf die **Einführung einer ubiquitären (allgemein verständlichen) Sprache**, die auf allen Stufen der Softwareentwicklung - von den Konzeptionsgesprächen bis hin auf die Code-Ebene - verwendet werden sollte.
- ▶ DDD legt besonders großen Wert auf die **Fachlichkeit einer Anwendungsdomäne**, die mittels der ubiquitären Sprache abgebildet werden soll. Daher steht die **Kommunikation zwischen SoftwareentwicklerInnen und DomänenexpertInnen** im Zentrum der Methode. Ein **Domänenmodell** basiert auf Objekten, die für die einzelnen Bestandteile der zu modellierenden Fachdomäne stehen und unterschiedliche Eigenschaften haben.
- ▶ Durch **regelmäßige Iterationen** nach dem DDD-Prinzip kann sich die Software kontinuierlich mit der sich stetig wandelnden Projektrealität verändern. Die Codebasis bleibt dabei im Einklang mit der Konzeptions- bzw. Modellierungsebene.

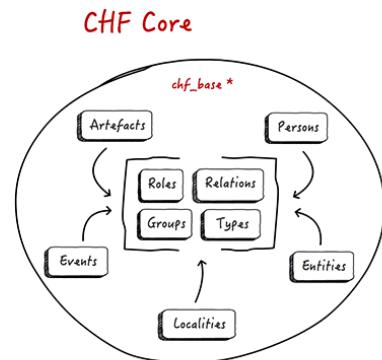


Eric J. Evans, Domain
Driven Design. Tackling

Complexity in the Heart of
Software. Boston 2003.

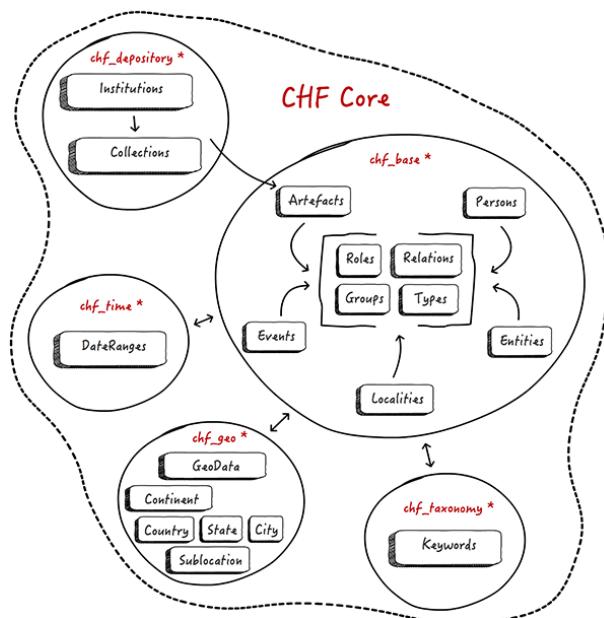
DOMAIN DRIVEN DESIGN

Am Beispiel des Cultural Heritage Frameworks (CHF)



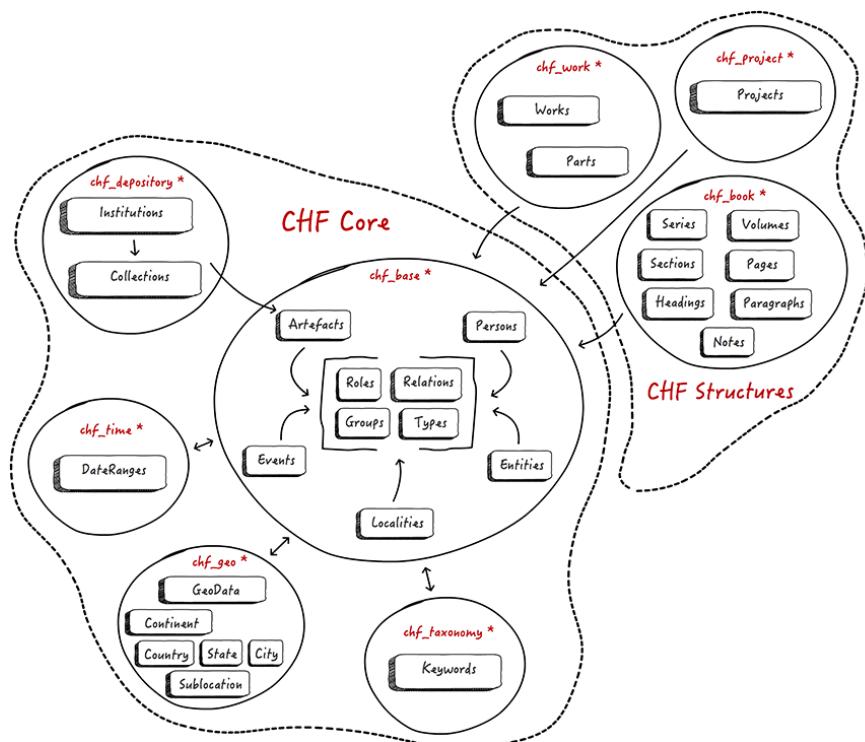
DOMAIN DRIVEN DESIGN

Am Beispiel des Cultural Heritage Frameworks (CHF)



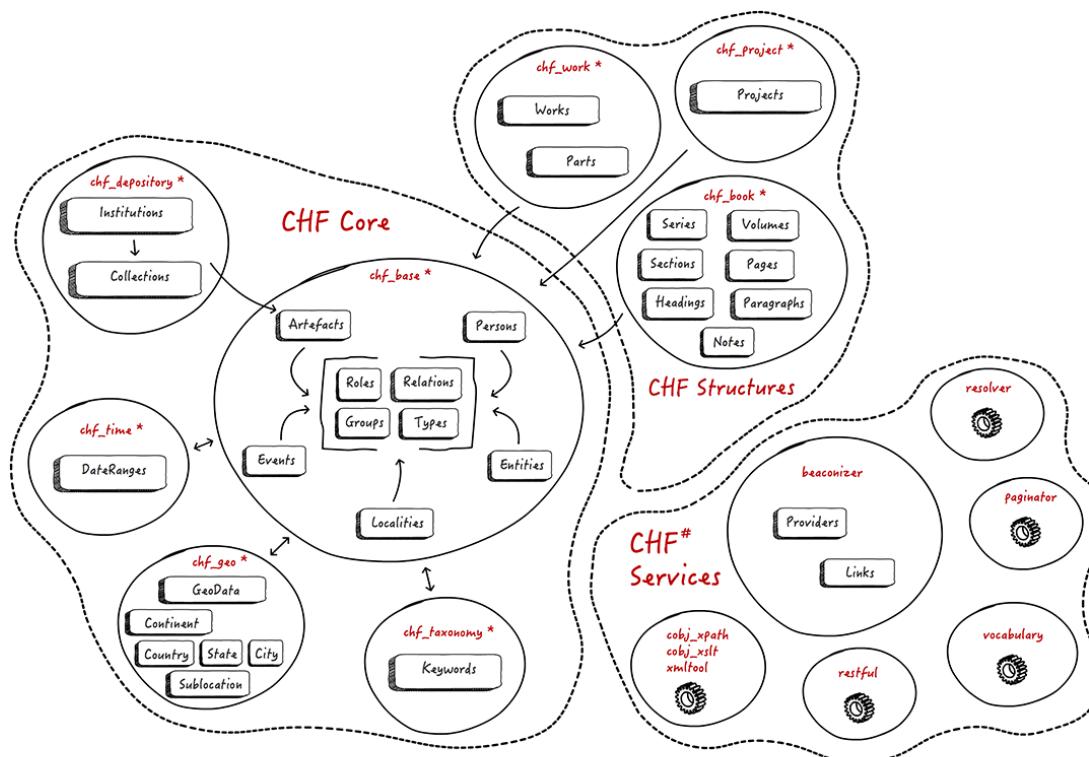
DOMAIN DRIVEN DESIGN

Am Beispiel des Cultural Heritage Frameworks (CHF)



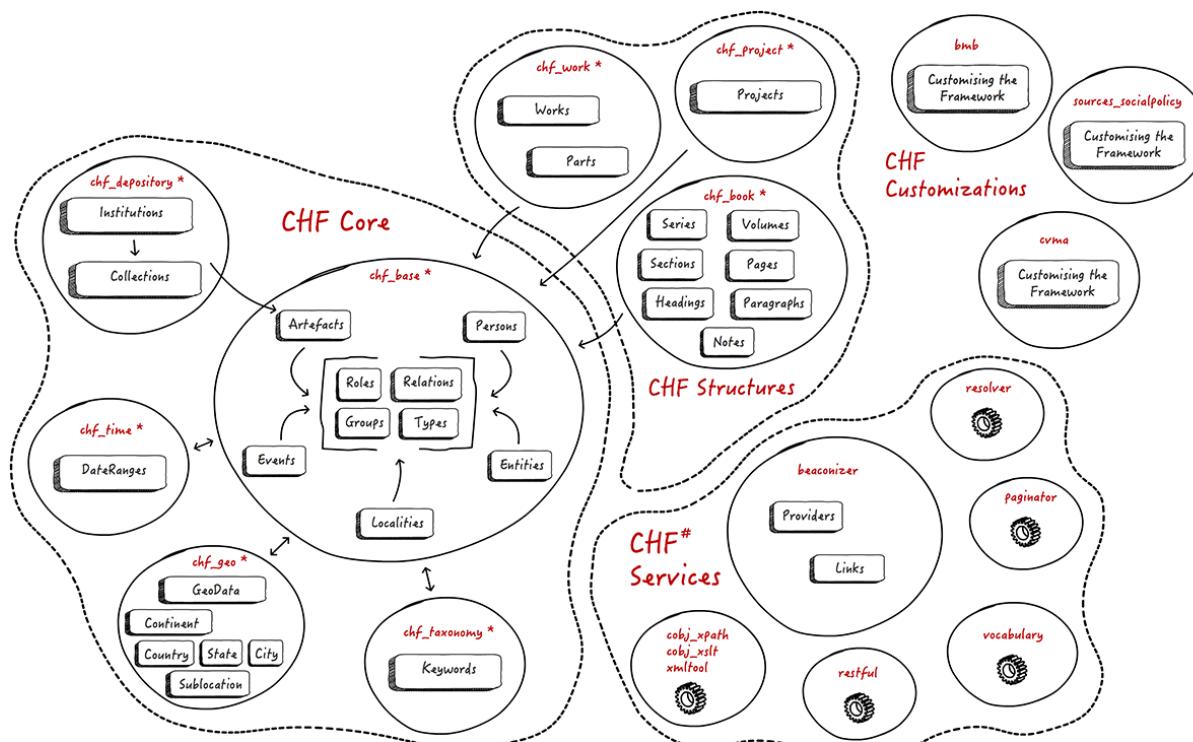
DOMAIN DRIVEN DESIGN

Am Beispiel des Cultural Heritage Frameworks (CHF)



DOMAIN DRIVEN DESIGN

Am Beispiel des Cultural Heritage Frameworks (CHF)



DOMAIN DRIVEN DESIGN

Implementierungsbeispiele

The screenshot shows the homepage of the Regesta Imperii website. It features a header with the logo and navigation links like 'Startseite', 'Unternehmen', 'Regesten', 'OPAC', 'ePublikationen', and 'Kontakt'. Below the header is a search bar with placeholder text 'Suche: Stichworte' and a button 'Suchanfrage absetzen'. A sidebar on the left contains sections for 'Expertenrecherche', 'Forschungsdienstleistungen', and 'Weiterführende Links'. The main content area displays a grid of document thumbnails.

www.regesta-imperii.de

The screenshot shows the homepage of the Corpus Vitrearum Deutschland website. It features a header with the logo and navigation links like 'STARTSEITE', 'PROJEKT', 'CVMA DIGITAL', and 'KONTAKT'. Below the header is a search bar with placeholder text 'Recherchewörter' and a button 'Suchanfrage absetzen'. A sidebar on the left contains sections for 'Forschungsdienstleistungen' and 'Weiterführende Links'. The main content area displays a grid of stained glass window images.

www.corpusvitrearum.de

The screenshot shows the homepage of the Deutsche Inschriften Online website. It features a header with the logo and navigation links like 'HOME', 'GLOSSAR', and 'KONTAKT'. Below the header is a search bar with placeholder text 'Suchbegriffe eingeben' and a button 'Suchanfrage absetzen'. A sidebar on the left contains sections for 'PROJEKT', 'THEMEN', and 'GEOGRAPHIE'. The main content area displays a grid of historical inscription images.

www.inschriften.net

The screenshot shows the homepage of the Controversia et Confessio website. It features a header with the logo and navigation links like 'EINFÜHRUNG', 'C&C DIGITAL', and 'PUBLIKATIONEN'. Below the header is a search bar with placeholder text 'Durchsuchen Sie die Datenbank:' and a button 'Aktuelles'. A sidebar on the left contains sections for 'Einführung in den Forschungsgegenstand' and 'Projekt'. The main content area displays a grid of historical document thumbnails.

www.controversia-et-confessio.de

The screenshot shows the homepage of the GluckGesamtausgabe website. It features a header with the logo and navigation links like 'PROJEKT', 'BÄNDE', 'WERKVERZEICHNIS', and 'LITERATURSUCHE'. Below the header is a search bar with placeholder text 'Durchsuchen Sie das Werkverzeichnis:' and a button 'Suchen'. A sidebar on the left contains sections for 'Das Projekt „Christoph Willibald Gluck – Sämtliche Werke“' and 'Weitere Informationen zum Projekt'. The main content area displays a grid of historical document thumbnails.

www.gluck-gesamtausgabe.de

The screenshot shows the homepage of the Digitales Familiennamenwörterbuch website. It features a header with the logo and navigation links like 'WÖRTERBUCH', 'PROJEKVORSTELLUNG', 'MITARBEITERINNEN', and 'AKTUELLES'. Below the header is a search bar with placeholder text 'Name', 'Rang', and 'Teknos'. A sidebar on the left contains sections for 'Wörterbuch' and 'NAMENSUCHE'. The main content area displays a grid of historical document thumbnails.

www.namenforschung.net

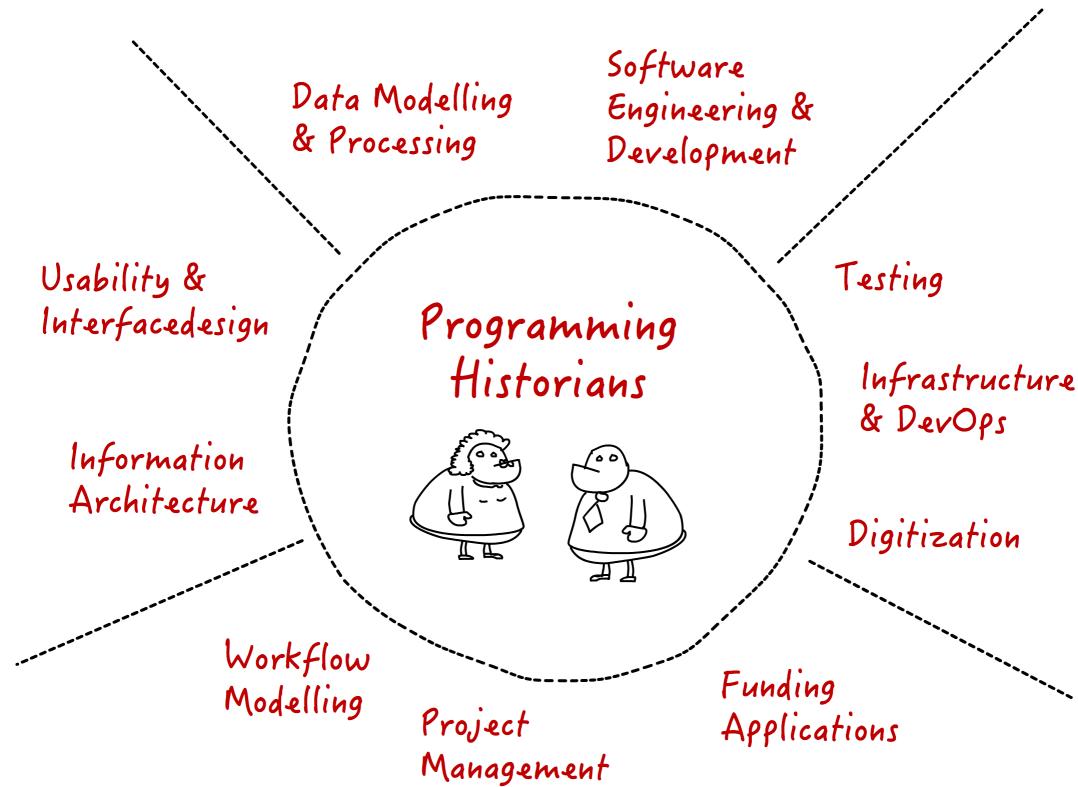
04

FAZIT

Geschichtswissenschaften im Spannungsfeld zwischen Software Engineering und digitaler Quellenkritik

PROGRAMMING HISTORIANS

Kompetenzfelder jenseits von Programmierkenntnissen



FAZIT

Programming Historians im Spannungsfeld von Software Engineering und digitaler Quellenkritik

- ▶ Source Code kann schon heute als **historische Quelle** betrachtet werden und wird als solche in Zukunft eine zunehmend große Rolle spielen. Für eine sinnvolle Auswertung dieser neuen Quellengattung müssen HistorikerInnen eine eigene **code literacy** aufbauen.
- ▶ Es geht **weniger um konkrete Programmiersprachen** oder bestimmte Verfahren **als um computational thinking** – eine Kompetenz für das Verständnis, die Analyse und die Auswertung (digitaler) Quellen.
- ▶ Die **Zeitinvestition**, die für den Aufbau entwicklerischer Kompetenzen getätigt wird, ist grundsätzlich **individueller Natur**. Hierin unterscheidet sich der Aufbau von Programmierkenntnissen nicht vom Aufbau spezifischer Kompetenzen im Bereich der historischen Grundwissenschaften.
- ▶ Gleichzeitig ist die **Verankerung eines Kompetenzaufbaus im Bereich des wissenschaftlichen Software Engineerings** in geschichtswissenschaftlichen Lehrplänen geboten.
- ▶ Hierbei geht es **weniger um konkrete Implementierungen von Algorithmen als um (software-)technische Architektur- und Denkmodelle**, also ein grundsätzliches Verständnis für die Zusammenhänge, die zu bestimmten Ergebnissen bei der Durchführung softwaregetriebener historischer Analysen führen.
- ▶ **Reproduzierbarkeit, Zitierfähigkeit und Offenlegung** von Software und Code, mit denen historische Forschungsergebnisse erzielt werden, sind Pflicht.

PROGRAMMING HISTORIANS

Unverzichtbare Kompetenz, methodisches Plus oder nicht zwingende Eigenschaft künftiger HistorikerInnen?

- ▶ **Unverzichtbar** ist ein Wissen um **informatische Denkmodelle, Begriffe und softwaretechnische Zusammenhänge**
- ▶ **Programmierkenntnisse** sind ein **methodisches Plus**
- ▶ **Nicht zwingend sind Kompetenzen in allen technischen Wissensdomänen**, die mit Software Engineering zusammenhängen (z.B. Infrastruktur, Testing, Interfacedesign etc.)

FINIS

Vielen Dank für die Aufmerksamkeit!

LINKS & SOFTWARE

Links

- ▶ [The Programming Historian](#)

Software

- ▶ [Impress.js](#) (Präsentation)
- ▶ [jQuery](#) (Animationen)
- ▶ [Unite Gallery](#) (Bildergallerien)
- ▶ [Magnific Popup](#) (Lightbox)
- ▶ [Skeleton CSS](#) (CSS Boilerplate)

Download

- ▶ HTML Version: <https://metacontext.github.io/2018-programming-historians/>
- ▶ PDF Version: <https://github.com/metacontext/2018-programming-historians/resources/pdf/programming-historians.pdf>
- ▶ Github Repository: <https://github.com/metacontext/2018-programming-historians/>
- ▶ License: [CC-BY 4.0](#), Torsten Schrade