With small steps to the big picture
A method and tool negotiation workflow

Konstantin Freybe[1], Florian Rämisch[1], and Tracy Hoffmann[1][0000−0001−8718−9536]

University Library Leipzig, Germany
{konstantin.freybe,florian.raemisch,tracy.hoffmann}@uni-leipzig.de

Abstract. In this paper we reflect on our research of Japanese video game culture, with focus on strategies of interdisciplinary collaboration. We understand our collaborative research as ongoing negotiation that aims at finding common ground between researchers from different backgrounds. We decided not to work on a single extensive question over the research period. Instead, we chose to work on a number of smaller problems (called Tiny Use Case (TUC)) that are aligned with superordinate research interests. Various methods from both humanities and information sciences were adapted and customized to these needs. A fundamental mutual understanding is essential for the various tasks in the team. The right choice and mix of methods and tools does not only depend on the specific team constellation (age, backgrounds, skills) but also a matter of available resources such as time, and the flexibility to explore.

Keywords: Interdisciplinary Collaboration · Mixed Methods · Tools · Software Development · Game Studies.

1 Introduction

In this study we reflect on our research, focusing on strategies of interdisciplinary collaboration. This is not only about working together, but sharing knowledge and establishing a mutual understanding. Our pursuit of this goal will be contextualized within our current research of Japanese video games.

We found that several adjustments to pre-existing concepts were beneficial to our work. We present our strategies in more detail in later sections, but sending ahead a brief summary hopefully helps drawing the connection from research content to our collaborative strategies more easily.

We understand our collaborative research as ongoing negotiation that aims at finding common ground between researchers from different backgrounds. Flexibility is crucial for collaboration, as long as it means to balance freedom of action with bindingness of reached agreements. We formalized this in what we call *TUC* workflow which is described after a brief introduction of the project diggr. The next section provides the evolution of the research interest and points to different methods we used to collaborate. These methods are presented in the following section. After a reflection about limitations of our approach we will end this paper with a conclusion about our work.

2 The diggr Project

diggr is a research project funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) and conducted by the IT department of University Library Leipzig and the Institute for Japanese Studies of Leipzig University. Our research focuses on Japanese video games in the context of global resp. globalized video game culture. Six members of both the library and the Institute for Japanese Studies with different disciplinary backgrounds (Information Science, Librarianship, Cultural Studies, Japanese Studies) will attempt to integrate expertise in data management directly in the research of humanities scholars from 2017 until 2019. According to Tabak [11], during the project, the researchers take one of two roles as a humanities scholar (H) or digital expert (D) or a combination (DH). "The H-role provides the content and D-role deals with the technical aspects of DH projects." [11]

The project pursues two goals. Firstly, build a data driven research infrastructure that uses e.g. Linked Open Data technologies and provides scholars with best practice solutions. Secondly, generate substantial contributions to video games research in general, research of Japanese video games more specifically. Our two humanities scholars lead their own sub-projects. The other staff members pursue tasks like software development, system administration or data modeling. One of these sub-projects will provide the context for our discussion and should, therefore, be presented as well. But before doing so, we discuss our adjusted use case structure which both sub-projects follow.

3 Tiny Use Cases

The beginning of the project presented itself as a challenge, as both the data situation and the required technologies were unclear, which in turn made it difficult to formulate objectives [6]. Therefore, the development of workflows for joint research was an important first step in this project. In order to be able to test them in research practice right away, we decided not to work on a single extensive question over the research period. Instead, we chose to work on a number of smaller projects (called Tiny Use Case (TUC)) that are aligned with the superordinate research interests. With their help, explorative approaches could be developed which promoted collaboration between information technology and content-oriented researchers. TUCs are designed to be conducted in rather narrow time frames of approximately three to four months. Generally speaking, a TUC is structured as shown in Figure 1 and as follows:

Mediation of the research interest/object H attempt to convey their research interest for each TUC as a research question to the team. This is as sensitive as it is critical. Without at least a basic understanding of the research interest presented to them, D cannot be expected to provide guidance in regards to software solutions that fit H's requirements. In turn, H have to adapt to the perspective of D if they want to be able to evaluate whether a proposed software does actually solve the task. This leads us to the next step, where negotiation shifts from conveying an idea to assessing adequacy of software tools.

Exploring software solutions In order to enable H to specify their software requirements appropriately, knowledge exchange is key. The basic idea of our approach is that D and H educate each other about the respective domain specific blind spots which leads to a common understanding and a shared technical terminology.

Evaluation A TUC workflow usually concludes with an evaluation. This is not only helpful for tracking progress of the team's work. It also allows us to critically reflect on the research conducted and to determine whether we reached the goals we set for ourselves. By evaluating frequently, as opposed to a single evaluation towards the end of a project, we have opportunity to thoroughly document our work.

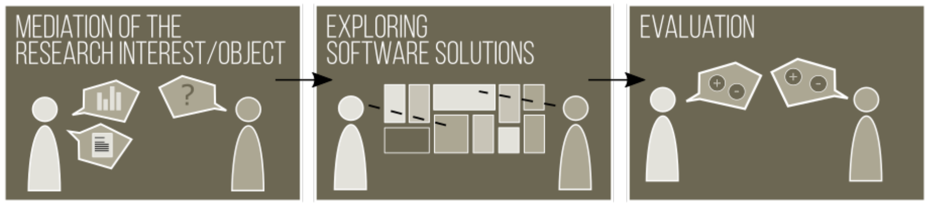
MEDIATION OF THE RESEARCH INTEREST/OBJECT
EXPLORING SOFTWARE SOLUTIONS
EVALUATION

Fig. 1. TUC Workflow

4 Subproject: Video games as Practice – Culture as Negotiation

One researcher in our team is investigating video game fan practices on social media in order to learn about their position and scope of action within the gaming industry. Many video game related practices on social media services like YouTube translate consumption practices into public or semi-public performances. By reconstructing careers of YouTubers, their transition from users to content creators to Influencers[1], H1[2] intends to learn about the relation between fan practices, consumption and labour. What kind of labour is it being a *Youtuber*, which tensions do they face and how do attempt which changes in the gaming industry?

[1] Influencer Marketing is a name for a strategy pursued by advertisers. The presumably stable relation between YouTubers and their audiences is targeted in order to convey advertising messages.

[2] H1 refers to a single H-researcher

4.1 Tiny Use Case 1

H1 chose the video game series *Metal Gear* as a topical frame. In TUC1, our first and rather prototypical TUC, H1 was interested in the reconstruction of canonicity among the 31 titles that are associated with the series. He began his exploration *in* one of the games [8] and suspected that the design of a particular mission is addressing a specific, hardcore fan audience. H1 encountered difficulties explaining the connection between his gameplay findings, canonicity and how databases could be helpful. Our computer scientists found it hard to understand H1's goal from a verbal report of the in-game situation alone. In order to overcome this obstruction, H1 adopted a popular practice on YouTube, and recorded a playthrough of the aforementioned mission, including audio commentary. This video was then uploaded and shared via YouTube.[3]

In abstract terms, H1 found that the design of the *Deja vu* mission showed various references to titles from the Metal Gear series. Regarding gameplay mechanics (i.e. the design of player interactions with the game software), H1 found an odd, normative distinction in how players could interact with these in-game references. While being an optional objective, successful manipulation of the game world is rewarded with a personal message from lead designer and producer Hideo Kojima.

Equipped with a list of titles that were referenced in the aforementioned gameplay situation, we investigated Kojima's contribution to the productions. We approached this by analyzing credit information from transcripts of staff rolls[4]. This allowed us to associate person names with functions and production units, and to count the roles attributed to individual names. H1 interpreted his in-game findings as an attempt to draw an image of Kojima as authorial figure and to enforce the canonic status of a subgroup of games from the *Metal Gear* series. While we could reconstruct that Kojima had the highest number of role attributions, several other staff member executed multiple production roles as well, indicating an inner circle surrounding Hideo Kojima. This dismantles the notion of him being the single author and shows that video game productions are predominantly team efforts. This raises questions on why and how the prominence of Hideo Kojima is maintained.

4.2 Tiny Use Case 3

When it was time to begin TUC3, the focus then shifted to YouTube. This platform was chosen from various social media services. Aside from YouTube, we considered Twitch, Facebook, Twitter and Patreon. But eventually we chose to limit ourselves to researching YouTube due to its relatively high API request quotas and accessibility. This time, the research question was based on the assumption that practices like *Let's Plays* closely relate to consumption practices.

[3] METAL GEAR SOLID V : GROUND ZEROES - Kojimas Kanonisierungsstrategie? https://www.youtube.com/watch?v=Z1frZ-zWptM

[4] This is quite similar to movie credits.

If public consumption of video games generates considerable income for the respective person, is it reasonable to view this as labour?

TUC3 was divided in three episodes, labeled with *a*, *b* and *c*. 3a was designed to perform an assessment of the field, YouTube in this case. H1 stated that he views the relation between YouTubers and their audiences or communities to be of critical importance for social practices on YouTube. Otherwise, efforts like e.g. *Influencer Marketing* would make little sense and could not be as popular and effective. The focus on social interaction determined the kind of data that had to be procured.

TUC3a produced valuable orientation and knowledge on how to cater to the researcher's requirements. The next two phases were headed in a similar direction: 1. extend the subject area, or: how much more data from YouTube can be handled with what tools?, and 2. stabilize the more advanced prototypes.

In order to decide on appropriate software solutions, D commissioned H1 to try already existing tools and document this as user stories. These documents provided the foundation for D to extrapolate H1's requirements. This includes tasks like interface design.

Although diggr strives for re-using existing solutions, this is not always possible. In order to enable individual researchers to deal with millions of comments on their own, semi-automated means of analysis seemed very appealing to us. DH-Tandems were formed in order to educate H1 on basic functionalities and align these methods with his superordinate methodological considerations.

Frequent evaluation of our progress did enable us to negotiate viable solutions. The results of these discussions were documented by H1 as requirement profiles. To experienced software developers, a requirement profile might be a rather common tool. For a humanities scholar who is somewhat distant from IT matters, this might seem rather unfamiliar and by no means self-explanatory.

5 Inventory of Tools and Methods

In this section we describe the methods and tools we used in the TUCs. We made use of various methods from both humanities and information sciences and adapted them to our needs. "As research generally is a creative process, some of the most interesting research questions only develop over time" [10], it demands of us to "welcome changing requirements" [1]. The methods presented here are ordered upon their possible introduction into our workflow, but can also occur at later or earlier stages depending on the course of the TUC.

Figure 2 illustrates the process that lend to our inventory, beginning with selecting and customizing methods from rather specific domains. If an evaluation shows that chosen methods sufficiently helped solve their task, they are introduced to our inventory, which we discuss in the following sections.

5.1 Research Let's Play

Starting diggr's search for traces of Japanese video games turned out to be difficult, at least difficult to explain. Is there a shortcut that bridges the gap

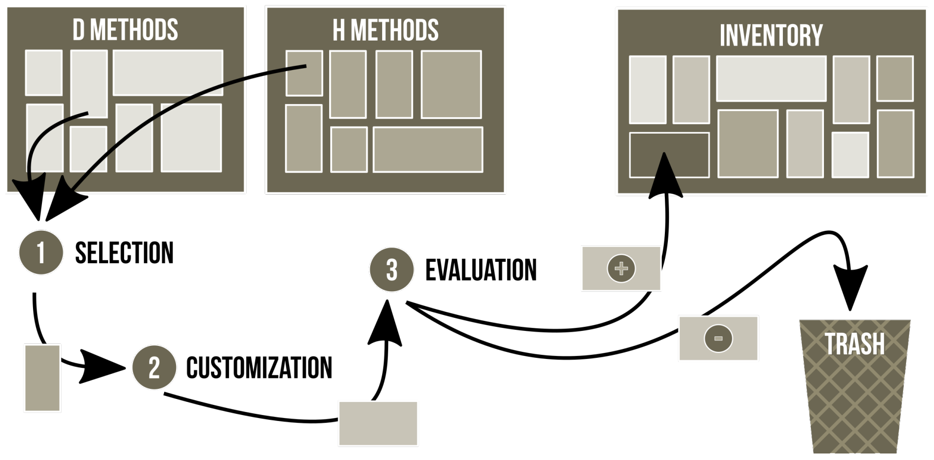

Fig. 2. Inventory Flow

between H1, who spent many hours with the games, and D, who might only know the titles from trailers or hearsay? How can in-game research be presented to team members so they provide a starting point for further research? H1 adapted a common practice of video game enthusiasts called *Let's Play*. While videos of this genre may be very diverse, they all have in common that gameplay is recorded as video and then uploaded online for others to view. Our team benefited from the low threshold and relative immediacy of this approach and, accordingly, we could soon start working on solutions. As the *Research Let's Play* can be seen as a protocol of an action or event, it is to be preferred over a live demonstration as it can be used as reference at a later time.

5.2 User Stories

Finding a common language is essential, when it comes to accurate description of expectations on functionality and user interface of software tools. D tend to overestimate the IT skills of H, while solid knowledge of the current research is expected. On the other hand, H have to be careful not assume that D are aware of Humanities specific presuppositions. Simultaneously, H depend on D's assessments regarding practicability of technical solutions.

In our group user stories turned out to be a good method to perform requirements engineering. H describes his expectations and wishes on the tools to be built, in simple and clear sentences. No attention is paid to the actual or presumed effort necessary to realize these. This is essential as some features appear simple to implement but aren't and vice versa. The user stories are then discussed in a feedback round in the group. D and H together agree on the desired and realizable feature set.

However, these procedures might yield the insight that some requirements cannot be met. In our case, this occurred when researching various social communication services in regards to their API accessibility. We discussed Facebook, Twitch, Twitter, YouTube and Patreon. Restrictive APIs and issues of privacy protection were reasons to focus on YouTube. The decision on a subject area therefore is the result of frequent adjustments and ongoing negotiation between the involved parties within the team. Collaboration has to be intensified in order to develop solutions when requirements can neither be fully met nor abandoned. In the following section, we present one way that helped us solving tasks that require a stronger focus.

5.3 DH Tandem

The DH Tandem consist of two persons: D and H. It is a temporary working group committed to a specific and small work package, while the rest of the team works on other topics. In our case, the Tandem worked on the design of the research dataset.

In the process both, H and D become domain experts, as H gets a better understanding of the process of acquisition and composition of data, while D is introduced to the scope of the research question. During the tandem sessions, both parties develop a common vocabulary, which does not consist of new words, but technical terms from the fields of both D and H. The tandem members function as translators and contact persons outside the tandem for the rest of the team. They can be seen as the collective product owners of a research question. Therefore in the tandem sessions, both parties, D and H, are on eye-level, but with a clear understanding of their roles. Therefore, it is possible for them to negotiate on the requirements of the software and research dataset. While cost here usually is not the limiting factor, it is human resources and temporal constraints that pose limitations on the final feature set and extent.

5.4 User Interface Design

Traditionally, when it comes to software, magic is allowed to happen – even desired – to surprise the customer and enhance its user experience. Capability, usability, performance, reliability, installability, maintainability and documentation appear to be the only metrics to be accounted for when evaluating customer satisfaction with software products [7].

In contrast, H do not strive for surprises, but comprehensibility. Magic is not allowed. For reproducible science, the flow of data in the program, the intermediate steps, manipulation, modifications, enrichments etc. need to be made transparent to H. Lack of transparency of the methods and transformations applied appears to be a common problem with research software, as Gibbs et al. [5] point out.

Burghard et al. [3] identified two main problems in the design of linguistic annotation tools: Wrong or counter intuitive feedback provided by the User Interface as well as unconventional controls. Both cases never occurred in our software projects so far, as the control elements as well as feedback by the software are designed in close cooperation between D and H. Requirements changes were negotiated in the DH-Tandem.

5.5 Provenance

In contrast to commercial software development, scientific development requires transparency and comprehensibility in regards to solutions and results. While individual TUCs are designed to be of short duration and intentionally limited scope, they line up iterative steps to contribute to superordinated research interests. Accordingly, this means including previously used data. In terms of comprehensibility, this iterative progression leads to the requirement to trace back how data was used and manipulated by whom. This is important for assessment and critique of the methods used and in extension the research conducted.

To allow for comprehensibility under these premises, this metadata about the origin and modifications of a file or data set is stored alongside the files. It is shipped with every research dataset in our group. To ease creation and use of provenance data the diggr team developed a tool for the creation, modification, display and export of provenance information: *provit.* [9] The tool is designed to be used in small research groups or by individuals. It aims to make retrieval and creation of provenance information as easy as possible.

5.6 Graphical User Interfaces

Graphical User Interfaces (GUIs) are essential for accessible research infrastructures. Lower entry bars, allow more scholars to work on their research questions empirically [10]. The reasons why matters of interface design are important for H are twofold. Firstly, H are commonly not accustomed to navigating through code or command line interfaces. Interfacing via frontend is less time consuming for H than to learn the required skills for pursuing alternatives. Secondly, H might lack the experience in operating software code and therefore may be unaware of risks. In order to avoid setbacks in the form of accidental deleting of files or causing fatal errors, interface design also becomes a precaution.

In contrast usable GUIs require a lot of effort in creation and maintenance. Insisting on GUIs for every experiment in the research process conflicts with the fast paced creative research process in general. (ib.) Often the tools D develop are only used once, or for a very limited amount of time. To verify our intuition, we developed GUIs for two applications: Human verification of automatically created links of entities of different databases and research of missing information on Japanese video game companies. After the tasks were completed, it was concluded, that the effort required to develop and maintain these tools was to much compared to their utility.

Not developing GUIs is not an alternative we could afford, so we decided to use parameterized GUIs, as they provide easy to use graphical representation of the data, with no need to navigate through code or command line interfaces. The advantage is, that instead of building hardly recyclable specialized GUI we now combine existing widgets, spend less time coding and more time educating H how to use the tools. Collaborating in this way was perceived to be way more sustainable, as the skills H learns in here are also useful outside the limited scope of a TUC, which cannot be stated for specialized GUIs. We acknowledge that other projects might come to different solutions for their projects, as our workflow is quite special and fast paced.

Elasticsearch, Logstash, Kibana (ELK) represents one of the most heavily used software stack in data science. Its popularity comes from its combination of powerfulness and ease of use. While Elasticsearch is a very powerful search engine, Logstash is a data processing pipeline collecting and aggregating data. Kibana is a frontend which can be used by the end user to operate Elasticsearch. With its integrated filtering and graphing tools it is useful to explore new datasets. ELK's great benefits for H lie in emphasis on data exploration and dynamic visualizations. In our case, H require a solution that allowed for exportable visualization that maintain the connection to the referenced or aggregated texts. Since H intend to employ various software tools for different purposes (preselection of data, orientation, visualizations outside of ELK), ELK proved very useful as hub where all research data is stored and from where derived datasets can be send to other tools for further processing.

Jupyter Notebooks are used by all team members at almost all stages of the software development process and research workflow prototyping process. "Jupyter notebooks are one means to make science more open." They "embody the FAIR (Findable, Accessible, Interoperable, Reusable) principles for digital objects and assess their utility as viable tools for scholarly communication." [2] In recent years they become popular for sharing research results with their underlying data and algorithms in one citeable research object. With this Jupyter Notebooks support the transparency and reproducibility of the research process.

Jupyter Notebook is a web based development environment and interactive user interface. The notebook server can be run in a data center. This turned out to be useful, as the computers H uses, sometimes are not powerful enough to run complex tasks in a reasonable amount of time. The notebooks are linear lists of cells, where each cell can be a piece of code, documentation, formulas, tables, images, plots and videos. With that, it can even be used to create interactive collages or even publications. The cells are executed one after another. Changes in one cell do not require the other cells to be rerun. E.g. to train a complex machine learning model, and then prototype the further data processing pipeline is very easy. This feature makes it a great tool for prototyping workflows and experimenting with datasets.

Jupyter notebooks also can be used in combination with repositories such as Github and Zenodo (for versioning and publishing). They offer a great intermediate step between providing a Graphical User Interface and exposing the scientists to a Command Line Interface. Getting in touch with source code in an environment which, through enrichment with pictures, explanations, plots and instructions can be way more appealing to a novice than a classical Integrated

Development Environment (IDE). Technical details are not hidden away, which makes the whole workflow more transparent for the whole research group [10].

From H's perspective, this is a valuable solution because H's actions are limited to small customizations at specific locations in the source code, as opposed to full access. The risk of causing damage to the software by unskilled editing or other reasons can be quite burdensome for H. Being freed of the need for permanent caution, H can contribute to the research process safely while also benefiting from working software – even at prototypical development stages. This is especially useful when refering to data analysis and (quasi-) dynamic visualizations.

Yet, being able to understand a scripting language is a valuable skill for H (ib.). We learned, that it helps H to follow the software development process more closely and get a better understanding of overall mindset.

5.7 YAML as configuration language

H needs to be able to configure programs according to their research interests and requirements, without having to build a GUI. A comparison of different machine data formats and data serialization formats led to the conclusion that *YAML Ain't Markup Language* (YAML), a data serialization format, is easy to be written by H and to be processed by our tools. YAML is with a clear syntax which shortens the time required by H to learn how to use it within the project.

In contrast to Jupyter Notebooks, which are used for analysis purposes, maintenance of configuration files turned out to be more practical when it comes to data acquisition. When assembling a dataset, e.g. from selection of YouTube channels, maintaining and managing YAML configuration files are relatively simple tasks which H can learn to solve more quickly than it takes D to write and provide Jupyter Notebooks – not to mention fully fledged GUIs.

5.8 Markdown as markup language

For texts which are not to be printed, like README files and documentation, blog posts, text drafts, meeting protocols, etc. document oriented file formats like Open Document Format and Office Open XML appeared unsuitable, as we often ran into formatting and paging issues. While H mostly used Microsoft Word and other WYSIWYG text processors, the information scientists preferred LaTex. As a compromise we decided to use Markdown as markup language for text in general. This has the advantage, that it can be written in a collaborative manner with CodiMD, and the results are easily and predictably convertible to Redmine Markup (for the issue tracker), HTML (for blog posts), PDF (for documents) latex (for publications) etc.

While having many output options, the clear and minimal syntax make it easy to read and write. It can be used directly within Jupyter Notebooks, or semi-WYSIWYG editors like CodiMD. There is immediate feedback on the correctness of the markup used, which helps both D and H to learn and remember the new languages.

Using Markdown and YAML has proven to be an effective approach for H and D to collaboratively work on projects with the same tools. This helps both the H and the D to better assess the skills and expectations of each other.

6 Limitations

The methods presented here have proven to be effective within our research group. Yet, they are far from being recommendable as best practices. The characters in the team and our environment aid intense collaboration. The relatively low demographic diversity in our team (all members between 28 and 38) may have contributed to finding common ground and a shared terminology. Team building events, such as gaming sessions in the GamesLab of the University Library Leipzig helped to build our team and increase the understanding for each other and the research context.

Spatial conditions might have made some methods and approaches more favorable than others. The whole team shares an office, which is (almost) exclusively used by diggr. Therefore face to face communication is common and problems often can be solved immediately without using an issue tracker.

Our TUC workflow has proven beneficial for collaboration in our team. But the applicability of methods outlined in this study is likely to depend on further customizations and adaptions by the adopting teams. A continuous evaluation of the research process helps to find compatible methods.

7 Conclusion

In this paper we presented our experiences, methods and tools for collaboration inside the digital humanities project diggr. A fundamental mutual understanding is essential for the various tasks in the team. To choose the right mix of methods and tools for the specific team constellation is a challenge and requires time, flexibility and the willingness to experiment.

It has shown that the humanities can benefit from agile approaches and methods in computer science. Computer science can also be enriched by the humanities, for instance through the continuous reflection of one's own work [4] and the need to make each step transparent and reproducible.

With this paper we hopefully contribute to further studies about the collaboration in digital humanities projects.

References

1. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D.: Manifesto for agile software development (2001), http://www.agilemanifesto.org/

2. Boscoe, B.M., Pasquetto, I.V., Golshan, M.S., Borgman, C.L.: Using the jupyter notebook as a tool for open science: An empirical study. CoRR **abs/1804.05492** (2018), http://arxiv.org/abs/1804.05492
3. Burghardt, M.: Annotationsergonomie: Design-empfehlungen für linguistische annotationswerkzeuge. Information - Wissenschaft & Praxis, vol. 63 (2012). https://doi.org/10.1515/iwp-2012-0067, https://www.degruyter.com/view/j/iwp.2012.63.issue-5/iwp-2012-0067/iwp-2012-0067.xml
4. Freybe, K., Hoffmann, T.: Iterative bearbeitung von forschungsfragen. In: Burghardt, M., Müller-Birn, C. (eds.) INF-DH-2018. Gesellschaft für Informatik e.V (2018). https://doi.org/10.18420/INFDH2018-04
5. Gibbs, F., Owens, T.: Building better digital humanities tools. Digital Humanities Quarterly **6**(2) (2012), http://digitalhumanities.org:8081/dhq/vol/6/2/000136/000136.html
6. Hoffmann, T., Freybe, K., Mühleder, P.: Workflows zur datenbasierten videospielforschung - am beispiel der populären videospielserie metal gear solid. In: Eibl, M., Gaedke, M. (eds.) INFORMATIK 2017. pp. 1113–1124. Gesellschaft für Informatik, Bonn (2017). https://doi.org/10.18420/in2017_113
7. Kekre, S., Krishnan, M.S., Srinivasan, K.: Drivers of customer satisfaction for software products: implications for design and service support. Management science **41**(9), 1456–1470 (1995)
8. Konami Digital Entertainment: Metal Gear Solid V: Ground Zeroes (Mar 2014), Sony PlayStation 4
9. Mühleder, P., Rämisch, F.: provit (Dec 2018). https://doi.org/10.5281/zenodo.2268521, https://github.com/diggr/provit
10. Reiter, N., Kuhn, J., Willand, M.: To gui or not to gui? In: Eibl, M., Gaedke, M. (eds.) INFORMATIK 2017. pp. 1179–1184. Gesellschaft für Informatik, Bonn (2017). https://doi.org/10.18420/in2017_119
11. Tabak, E.: A hybrid model for managing dh projects. Digital Humanities Quarterly **11**(1) (2017), http://www.digitalhumanities.org/dhq/vol/11/1/000284/000284.html