An Online Repository for Educational Resources in HCI-Engineering

Lucio Davide Spano[1][0000−0001−7106−0463], José Creissac Campos[2][0000−0001−9163−580X], Anke Dittmar[3][0000−0003−4810−4573] and Peter Forbrig[3][0000−0003−3427−0909]

[1] University of Cagliari, Cagliari, Italy
davide.spano@unica.it
[2] University of Minho & HASLab/INESC TEC, Braga, Portugal
jose.campos@di.uminho.pt
[3] University of Rostock, Rostock, Germany
[anke.dittmar|peter.forbrig]@uni-rostock.de

Abstract. This paper leverages the outcomes of the first workshop on HCI Engineering Education [4] to create an online repository where the community can share content relevant to HCI. The repository takes advantage of the functionalities of the Git file versioning system to support presenting and adding content. The paper describes the structure of the repository and the process for adding new content. In addition, we propose an adaptation of the framework for presenting teaching samples, supporting more flexibility in the application of educational material for different teaching objectives. The new presentation format starts with describing a design problem and emphasises the students' applied understanding of conceptual and theoretical knowledge. The presentation format is demonstrated and discussed by the example of an end-user design tool for mobile data collection.

Keywords: HCI Engineering, education, teaching, case study, repository, Git, end-user development, model-based design of user interfaces

1 Introduction and Background

The quality of an interactive computing system should be considered along various dimensions. For instance, we must consider the quality of the system from the perspective of its user interface's design. This is the focus of human-centred design processes [11] and leads us to think about the system in terms of its usability [10]. We also must consider how such design is implemented, which leads us to think about the quality of the system's implementation (e.g., its architecture and the technologies used). Each of the dimensions requires a distinct set of skills, making the design and engineering of this type of system an interdisciplinary effort. Interdisciplinary work is characterized by both distributed work by specialized sub-teams (e.g., software engineers, interaction designers, ergonomists) and collaboration in heterogeneous teams [1]. Accordingly, students

Authors' version of the paper published in Design for Equality and Justice. INTERACT 2023. Lecture Notes in Computer Science, vol 14535. Springer, 2024. The final publication is available via: http://dx.doi.org/10.1007/978-3-031-61688-4_19

need to be trained within their chosen discipline, but additionally they should also be exposed to assumptions, values and constraints in other disciplines and to design activities that cross disciplinary boundaries [15].

Research on design pedagogy in Human-Computer Interaction (HCI) has raised considerable interest in recent years (cf. the research agenda by Wilcox et al. [20]). Human-computer interaction engineering (HCI-E) education in particular aims at preparing students in computer science and related areas for collaborative design practices. In this context, McCrickard et al. [16] criticize HCI courses that merely provide an overview of various elements of interface design or have "students building cool interfaces". Oleson et al. [18] point out that computer science students often experience difficulties "to engage meaningfully with designerly aspects of HCI... or erroneously view design as easy, inessential work". In a previous workshop [2], several challenges of HCI-E education were identified, from keeping courses and curricula up to date with fast evolving technologies and methods, to ensuring that computer science students understand that different types of users will interpret and react to interactive systems differently, to the need to consider real-life systems in the learning process. The discussion was grounded on four concrete projects and exercises used in HCI Engineering courses lectured by the workshop participants, and taken as samples of relevant teaching materials to promote the learning of HCI Engineering concepts through practice. The rationale for this was an understanding that students will benefit from "realistic case studies that simultaneously exemplify HCI concepts and model the process of designing [but also developing] usable systems" [19]. An outcome of the discussion was the proposal of a common presentation framework for teaching samples [4]. The immediate goal of the framework was to help characterise and compare the different teaching samples. A medium-term goal was the creation of an online repository of educational resources for HCI-E.

This paper first presents an implementation of such an extensible online resource to where the materials from [4], and others, might be uploaded and made available. The implementation employs existing tools such as the Git file versioning system and the blogging engine Jekyll [12]. The paper describes how teaching samples can be added, modified, and navigated by using an illustrative example. The paper then reconsiders the presentation framework in [4] and proposes an adaptation which allows for more flexibility in the application of educational resources within different course settings. In the adapted presentation format, specific design problems are considered to be the central element. They ideally require theoretical background knowledge from different domains and have the potential to support various teaching objectives. Again, an example is used to illustrate the suggested ideas. The paper closes with a discussion and future work.

2 The Online Repository

The repository's success depends on the quantity and quality of the collected teaching material. On the one hand, the quality requires a detailed and consistent categorisation and description of each sample, including, for instance, its aims, the application domain, the target students, etc. On the other hand, to maximise the number of teaching samples described in the repository, we need a fast and straightforward procedure for uploading data and allowing the authors to update the description if required. Keeping in mind these two requirements, we support uploading a new sample and editing the existing description by leveraging a development tool HCI engineers are familiar with: the Git file versioning system. A first implementation of such a system is available at the following URL: `https://github.com/IFIP-WG-2-7/Teaching-HCI-Engineering`.

2.1 Implementation Overview

We host the repository on GitHub [9] which allows us to maintain the data, keeping track of the changes in the description of the teaching samples. In addition, we leverage the hosting support for code-project websites available on GitHub, called GitHub pages [8]. Such support exploits a blogging engine called Jekyll [12], which allows transforming text by using different common syntaxes (e.g., Markdown, HTML etc.) into static web pages. Our idea for supporting the repository was to leverage this to automatically create an online readable version of the repository data for easing the teaching material search and navigation without requiring the authors to produce it.

In summary, the teaching sample GitHub repository consists of the following elements:

- A basic Jekyll installation, containing the definition of the home page and the templates for rendering the teaching samples overview (by listing them) and details (reporting the sample's data in a human-readable way). We detail the generation process in Section 2.3.
- A folder containing the teaching samples data. It contains a sub-folder for each sample, and includes a structured description and optional assets (e.g., images, documents, code packages etc.) that may enhance the description. We describe such structure in Section 2.2.

We exploit standard pull requests to the repository to request the insertion of a new teaching sample and modify existing definitions. Such a mechanism allows us to review the proposed repository changes and fix possible problems before accepting the request. We detail such a process in Section 2.4.

2.2 Describing a Teaching Sample

Each teaching sample has a dedicated sub-folder, which must include at least the *description.md* file. As suggested by the name, it contains the description of the teaching sample, containing both meta-data and a textual explanation of the sample.

The file slightly enhances the presentation format discussed in [4] for identifying the relevant information about a teaching sample. It contains a header section, including a YAML[4] structure for metadata. The fields are the following:

- **layout**: A value allowing the engine to select the correct template for rendering the data. Currently, we have only one template for all samples, but this allows us to support sub-categories in the repository and to render the information using the appropriate structure.
- **permalink**: the repository sub-url the author would like to assign to the current sample;
- **title**: The teaching sample title.
- **authors**: a structured list including the information about the teaching sample authors. Each **author** entry contains the following information:
  - **name**: the author's name;
  - **surname**: the author's surname;
  - **url**: the author's homepage;
- **type**: the type of the teaching sample, either *project* (i.e., a piece of work developed throughout the course or as a final assignment) or *exercise* (i.e., a piece of work developed for a sub-part of the course);
- **application-domains**: A list of domains where we can apply the teaching sample.
- **interaction-techniques**: The list of interaction techniques supported in the teaching sample (e.g., WIMP, mobile, gestures etc.).
- **current-students**: The background of the students that currently use the teaching sample (e.g., Computer Science, Design etc.).
- **student-level**: The level of the students that currently use the sample (L1, L2, L3 for first, second and third Bachelor's year; M1 and M2 for the first and second year of the Master's).
- **HCI-theme**: A brief description of the covered HCI themes.
- **tools**: A list of tools used for teaching (e.g., Android toolkit, startup code etc.);
- **brief-description**: A brief abstract of the teaching sample.

The textual explanation of the teaching sample must contain a Markdown-formatted rich text, optionally including media elements (images, videos, links etc.). It must contain the following sections:

1. **Targeted students and prerequisites**. This section contains the description of the students and their pre-requisite levels in the disciplines involved in HCI-E, including HCI and Software Engineering (SE);
2. **Objectives**. A description of the pedagogical objectives of the teaching sample.
3. **Pedagogical steps/monitoring, initial materials and tools**. A description of the pedagogical management of the sample, including the tools used and the initial materials provided.

[4] `https://yaml.org/`, last visited October 09, 2023.

4. **Expected outputs and Evaluation**. A description of the expected outcomes and their evaluation.

Figure 1 shows a concrete example of a *description.md* file describing a teaching sample. The header part, contained between the three minus (hyphen) characters (`---`), contains the YAML structured meta-data. The body part allows formatting the content through headers, bullet lists, etc., using the Markdown syntax. The ellipses in the example (represented by the `[...]` symbol) correspond to parts of the text that have been truncated for brevity sake. The corresponding human-readable rendering of the information is depicted in Figure 3.

2.3 Navigating the Samples

The rendering of the teaching sample consists of two templates for generating an HTML version of the data contained in the repository. The first template displays the list of samples in the repository, showing an overview of their definition. It basically consists of a card-based visualization of the following fields: title, brief description and application domains. Figure 2 shows the layout of the list. People can access the details of a teaching sample by clicking on the title.

Currently, the navigation of the existing teaching samples does not support any filtering or search. However, we have already planned changes to the current listing template to support keyword-based searches. In addition, we plan to add filters at least on the application domains, the student level and background, leveraging on the description meta-data.

The second template provides a detailed view of a teaching sample. It includes the rendering of all the pieces of information in the *description.md* file, as shown in Figure 3. The layout consists of two columns. In the main one (on the left), we show the title, the brief description and all the sections of the textual explanation, reporting the text and the media elements included in the description file. In the sidebar on the right, we display the meta-data information as an "identity card" of the current sample. It also includes the author list and a link to their homepages for giving credit to the authors.

2.4 Adding or Modifying a Teaching Sample

Providing an easy yet effective way to propose new teaching samples or modify the description of existing ones is crucial for the repository's success. As already mentioned above, we leverage a well-known process for managing this process, the *pull requests* provided by the Git versioning system.

Usually, pull requests represent a notification to team members that a developer completed a feature, requesting its integration into the main development branch. Adding a new feature may require adding new code files or changes to existing ones. Thus a pull request may contain a combination of both. In open-source code projects, developers often use pull requests to ask the community to adopt some changes to the codebase, allowing "external" developers (i.e., people who are not the core maintainers) to contribute actively to the project. A

```
---
layout: case-study
title: Electronic Prescription System
authors:
   - author:
        name: José
        surname: Campos
        url: "https://www.di.uminho.pt/~jfc/index.shtml"
type: Project
application-domains:
   - Information systems
   - Health
interaction-techniques:
    - WIMP
current-students: "Computer Science"
student-level: L3
HCI-theme: "UI design and implementation"
tools:
     - None
brief-description: "The Project consists in developing a system to
↪  support doctors in prescribing medicines. [...] "
---

## Targeted students and prerequisites
The electronic prescription system project was first used in a third-year
↪  course on object-oriented analysis and design (OOAD).
[...]

## Objectives
As stated above, from an HCI Engineering perspective, one objective of
↪  the project is to raise students' awareness of user-centered
↪  concerns.
[...]

## Pedagogical steps/monitoring, initial materials and tools
The students self-organize in groups of 3 to 5 to carry out the project.
↪  The project is executed during the semester as the topics are worked
↪  on in class.
[...]

## Expected outputs and Evaluation
The expected outputs are the models mentioned above and the corresponding
↪  system implementation.
[...]
```

Fig. 1. A sample *description.md* file from the repository.

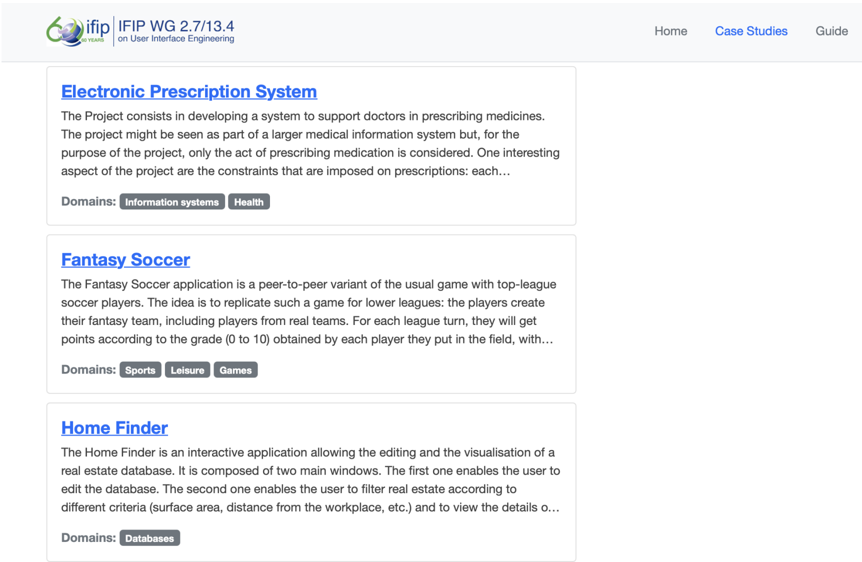

Fig. 2. Overview of the teaching samples in the repository.

pull request must be accepted by the code repository owner(s) to be included in the codebase. All major code hosting services (e.g., GitHub) provide means for discussing such changes, asking for modifications to the originally proposed changes until a consensus for accepting or discarding the request is reached.

Such a supervised process for incorporating changes proposed by the community fits well with our needs. On the one hand, it allows people to proactively suggest new content to be inserted into the repository collection. The only requirement is having a GitHub account. On the other hand, it allows us to enforce a quality standard on the published content by reviewing the proposed changes and discussing them with the author and the community before accepting them. Through such review, we ensure that all the proposed samples are relevant for the repository, contain the required meta-data, and the textual description is clear enough for the reader and other sanity checks on the content.

For proposing or changing the description of a teaching sample, who proposes new content (the *author* from now on) should perform the following steps:

1. *Fork the Teaching HCI Engineering repository on GitHub.* This creates a personal copy of the repository in the author's computer, where s/he will perform the changes. The author may use the same copy for more than one pull request.
2. *Create a new branch for the current request.* A branch is a separate version of the main Git repository, which allows working on different features without affecting the main project. In our case, we use it for working on a specific teaching sample.

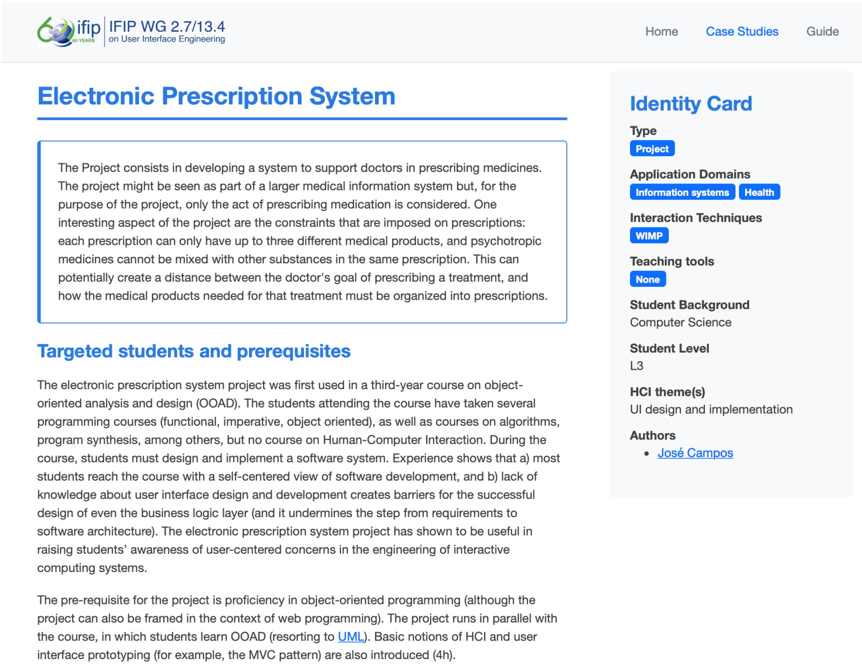

Fig. 3. Sample detailed visualization of a teaching sample.

3. *Create or edit the sample's content.* In the personal copy of the repository, the author adds a new sub-folder to create a teaching sample, or s/he edits the files to update the description of an existing one. In both cases, changes must be compliant with the structure introduced in Section 2.2.
4. *Commit the changes in the personal copy.* The author commits the changes in the personal copy, adding a small description of the new version.
5. *The author sends the pull request to the original project.* This notifies us, as the repository owners, that there is a new request.
6. *The owners review the requested changes.* In this phase, we review the changes contained in the request to ensure that the content meets the quality standards for being accepted. If needed, we initiate a discussion with the author through GitHub for clarification. We may propose changes to the request or ask the author to perform them. Eventually, we will make a decision about accepting or discarding the request.
7. *Merge on the main branch.* In case of a positive assessment, we accept the request, and the new content is merged into the main project branch. This results in the update of the readable version. discussed in Section 2.3.

3 The Presentation Framework Revisited

The above described implementation of the online repository builds on the framework in [4] with teaching samples being organised around student projects or exercises (see Section 2.2). In the process of providing a repository for maintaining and sharing the teaching samples, we also dedicated effort to reviewing their format. In this section, we examine, through an example, a modification of the presentation format in which specific design problems or design tasks are considered to be the central element of teaching samples. The design tasks ideally require theoretical background knowledge from different domains and have the potential to support various teaching objectives. The original template mentions application domain and HCI theme but these are not rich (detailed) enough to capture the fact that, from an educational perspective, we will be interested in conveying specific concepts and knowledge.

3.1 The New Format

We propose a new format for adapting the description of teaching samples to different educational usage scenarios, i.e., different ways for using the sample and the related material to reach specific pedagogical objectives. In addition, we propose also to explicit the conceptual and theoretical issues addressed or exemplified through the sample. In detail, the new format contains the following items:

1. Identity card with a title, a brief description of the design task, application domain, interaction techniques,
2. Description of the design problem,
3. Related conceptual and theoretical issues
   - Issue 1: description and references
   - Issue 2: ...
   - ...
4. Materials (provided to or created by the students),
5. Educational usage scenarios
   - Context 1: teaching format, participants and pre-requisites, pedagogical objectives, activities
   - Context 2: ...
   - ...

The original template in [4] characterises an application context but the same resource might be used in multiple educational usage scenarios. Currently to represent that the item would have to be replicated multiple times, one for each context. We argue that the adapted format allows for more flexibility in the application of teaching samples within different course settings. This is illustrated in the following example of a design tool for mobile data collections.

3.2 A Teaching Example Described through the Revised Presentation Framework

Identity Card

– Title: MDC design tool,
– Design task: A design tool that allows end users to create mobile applications for in-situ data collections,
– Application domain: in-situ data collections,
– Interaction techniques: WIMP, WIMP mobile.

Description of the Design Problem

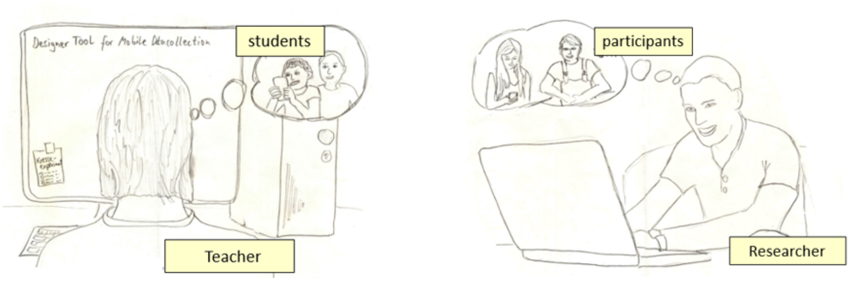

Fig. 4. A teacher and a researcher using the MDC design tool to create an app for their students and study participants respectively.

In the example, we consider tool support for conducting in-situ data collections - a technique which can be used for diverse purposes. For example, a teacher may wish to ask students to collect data during a biology field trip or a researcher may want to apply the experience sampling method in their empirical study [14]. Mobile devices can facilitate such in-situ data collections. How can the teacher, the researcher or another questioner be supported in creating apps for their respondents according to the needs of the specific situation? In other words, what we are looking for is a design tool for mobile data collections (in short MDC design tool) that can be used by end-users. Figure 4 illustrates the use of the MDC tool (a desktop application) by the teacher and by the researcher.

Related Conceptual and Theoretical Issues

Understanding Roles in End-User Development. In the considered application domain (in-situ data collections), we can distinguish between questioners (e.g., the teacher and the researcher in Figure 5) and respondents (e.g., the students and the study participants in Figure 4). The MDC design tool supports end-user development and, more specifically, end-user design. A simple distinction between users, designers and developers is not sufficient and students need to understand the dual role of the questioners. They are end users of the MDC design tool, and at the same time, designers of MDC apps for the respondents (see Figure 5). This dual role is referred to as designer-user in [6].

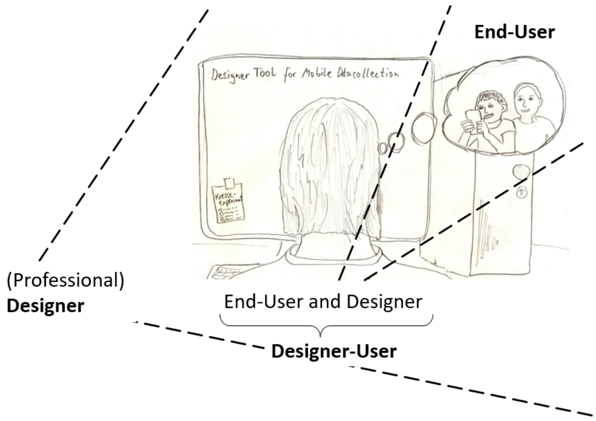

Fig. 5. The dual role of questioners and nested design spaces.

A key assumption in end-user development concerns the differences in the contexts and goals of people acting in the role of a professional developer or an end-user developer. Professional developers are responsible for the quality of the system under development, but typically they are not the users of the system. In contrast, end-user developers have less interest in or understanding of quality criteria, but want to modify or create a system for situated use [13]. A questioner/designer-user in our example wants to create a MDC app for a specific situation, for example, for a biology field trip with their school class.

Design Spaces. The concept of design space is considered to be a valuable tool for understanding design processes in terms of design options and design constraints. In the example, the professional designer needs to be aware of the nested design spaces that are indicated in Figure 5. The designer-users move in their own more situated design spaces. The way the MCD design tool is shaped has a profound influence on how designer-users can shape the in-situ data collections and their interaction with the end users. The professional designer has to understand the goals and needs of the designer-users (both in their role as end-users and as designers), the goals and needs of the app users, and the relationships between them [6].

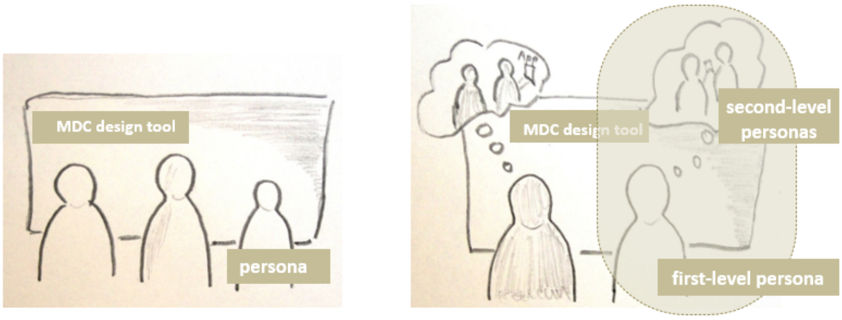

Fig. 6. Left hand side: application of "classical" persona method, right hand side: use of two-level personas – a first level persona represents a group of designer-users and its second-level personas represent the corresponding groups of end users (from [6]).

Design Representations for Multidisciplinary Work. Software development processes can be characterized as processes of creating, refining, modifying or discarding design representations or models. Basically, two types of models can be distinguished: system models describing the interactive software system under development itself and contextual models describing the context of use (both in current situations and in future). Among common system representations for multidisciplinary collaboration are user interface sketches and prototypes, popular contextual models are personas and scenarios. The example helps students to appreciate the value of shared design representations. Moreover, it can help them understand that contextual models also have to be adapted to address design problems more adequately. For example, two-level personas consisting of a first-level persona and some second-level personas (Figure 6) are more effective in capturing design problems with nested design spaces [6].

End-User Software Engineering and Model-Based Approaches. End-user software engineering supports end-user developers by systematic and disciplined activities that address software quality issues. There are two general approaches: dictating proper design practices and injecting good design decisions, e.g., by a combination of constraints and generation mechanisms [3]. As for our example, the latter approach is interesting and can be implemented by a domain-specific model-based approach such as in [7] following the Cameleon Reference Framework [5].

Materials

Materials provided to or created by the students can include:

– description of the application domain (e.g., in-situ collections, question types)
– personas and scenarios (Figure 7),

– user interface sketches of the MDC design tool and generated apps (Figure 8),
– paper and software prototypes (Figure 10),
– meta-models, transformation rules and models for generating apps (Figure 9).

Educational Usage Scenarios

We show how the same teaching sample may serve two different educational scenarios by detailing two usage contexts. Each one has its specific participants, pedagogical goals and activities.

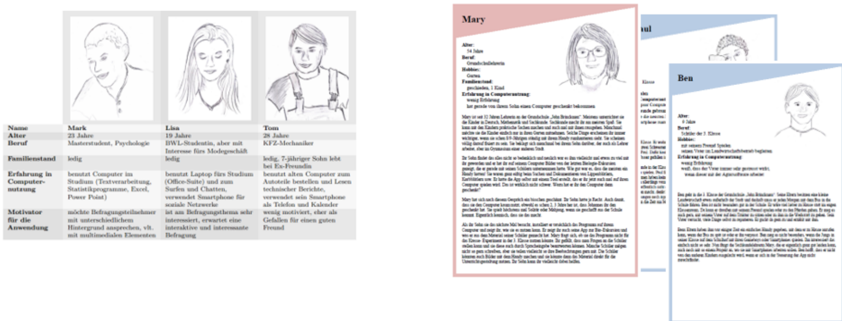

Fig. 7. Two two-level personas: Mark with Lisa and Tom (template format), Mary with Paul and Ben (narrative format).

Context 1. User-centred design courses consisting of a lecture part and an exercise part.

– *Participants:* Students of computer science and related engineering disciplines.
– *Pedagogical objectives:*
  - appreciation of user-oriented design approaches,
  - applied understanding of UCD methods and design representations.
– *Activities:* Topics in lecture include those described in Section 3.2. The example is used throughout the exercise part of the course.
  - introduction of the design task,
  - creation of two-level personas (Figure 7),
  - brainstorming with scenarios and UI sketches of MDC apps for end users,
  - design ideas for the MDC design tool (how can the designer-user design the structure of MDC forms and MDC navigation?): individual UI sketching, sharing of ideas and consensus building by using the personas (Figure 8),

- refinement of scenarios,
- creation of paper-prototype,
- prototyping session.

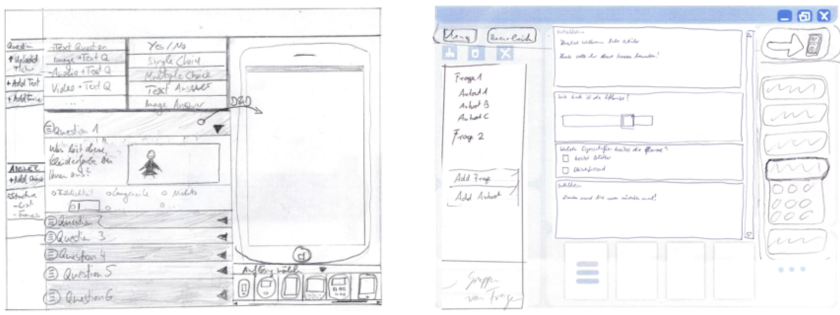

Fig. 8. Two individual UI sketches of the MDC design tool.

Context 2. Project work (within or in connection with HCI engineering course).

– *Participants:* Students of computer science and related engineering disciplines.
– *Prerequisites:* Students are familiar with ideas of model-based development of user interfaces and tool support (e.g., EMF[5], ATL[6]).
– Pedagogical objectives:
  - knowledge and skills in model-based development,
  - use and appreciation of contextual models.
– *Activities:*
  - introduction of the project task,
  - students are provided with (two-level) personas, scenarios, UI sketches,
  - creation of meta-models (domain, abstract UI, concrete UI, final UI) and transformation rules (Figure 9),
  - embedding in simple UI (Figure 10).

We used the example design task in scenarios as described above but other scenarios are possible. For example, interaction design students could use 'traditional' personas and two-level personas to reflect on the effectiveness of these variants (see also [6]).

[5] https://eclipse.dev/modeling/emf/
[6] https://eclipse.dev/atl/

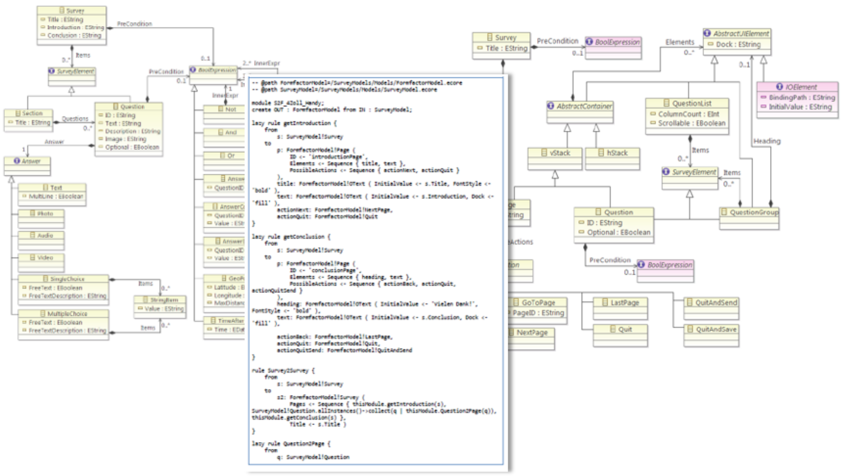

Fig. 9. Meta-models for domain models and abstract UI models (Ecore models in EMF) and some transformations rules (ATL).

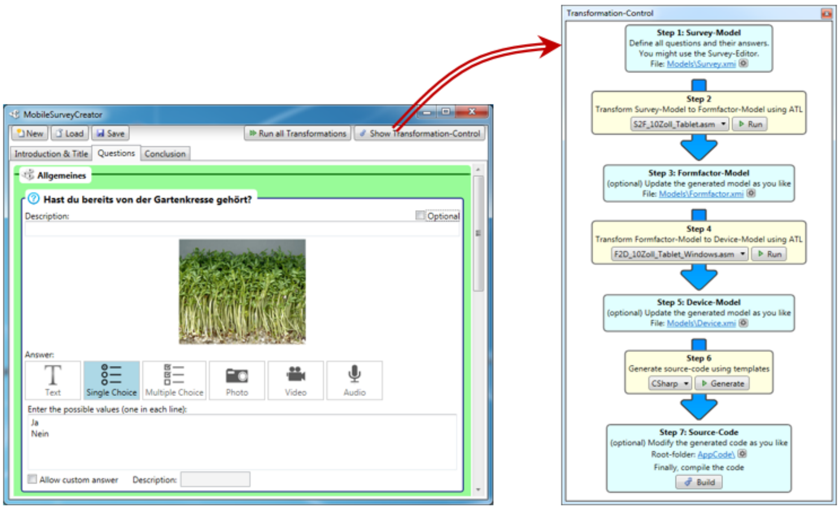

Fig. 10. Software prototype with implemented stepwise generation of apps (using Ecore models and ATL in the model-based approach).

4 Discussion and Future Work

The solution for managing the sample repository presented in this paper highlights the feasibility of implementing ideas from previous workshops with minimal effort, an essential aspect in the context of a repository predominantly relying on voluntary contributions. In this section, we discuss some further aspects related to the ongoing effort of collecting and sharing the samples, considering factors such as tool familiarity among software engineers, effective presentation of educational resources, the role of community discussions, and the extensibility of the repository.

Tools Familiarity and Support for Discussion. One of the strengths of the proposed solution is the use of tools that are readily familiar to software engineers. This familiarity ensures potential contributors can use these tools comfortably without a steep learning curve. The reviewing tools usually employed in collaborative coding also facilitate the discussion of teaching samples and a final quality check by the repository owners. This is a key advantage, as it lowers the barriers to entry for new contributors and encourages a more diverse and engaged community of participants.

Furthermore, these tools provide a platform for discussing and refining teaching samples. Through collaboration with other peers, contributors engage in discussions, share their insights, and collectively improve the quality of educational resources. This collaborative aspect is essential, particularly considering that voluntary contributors rely on a sense of community and shared purpose to drive progress.

Effective Presentation of Educational Resources. The question of how to present these educational resources effectively is another fundamental consideration for continuing our work. The paper discusses two alternative approaches. On the one hand, concrete ideas and examples are proposed as valuable resources. These concrete examples serve as practical models for educators and learners alike, providing actionable insights into how to approach various topics.

On the other hand, it is equally important to present material in a way that allows for adaptation to the specific teaching and learning contexts of diverse users. One-size-fits-all approaches often fall short in education, where individualisation and contextualisation are key. By emphasising the adaptability of our educational resources, we cater to a wider audience with varied needs and preferences. This flexibility aligns with the overarching goal of making the repository a valuable resource for many users. One idea we would like to leverage again from software engineering tools is *branching* the learning resources, allowing us to maintain the original content but providing means to develop further, personalise and share tailored versions of the same sample.

Semantic navigation of the teaching samples. One noteworthy idea discussed during the workshop was the creation of an ontology for teaching examples. This ontology aims to enhance navigation within the repository by providing a structured and standardised way to categorise and discover educational resources. On the one hand, providing semantic relationships in the collected data helps find more significant content and navigation. However, building an ontology requires a considerable amount of work. Therefore, it may be worth considering adopting or adapting existing ones.

Extensibility and Format Adaptation. Beyond the initial implementation of ideas from the HCI-E^2 workshops, the repository needs to address the issue of the moving targets [17] evolving and adapting to the changes in the needs of the community. Two critical aspects of this evolution are content extensibility and format adaptability.

Content extensibility refers to the repository's capacity to grow by accommodating an increasing variety of teaching samples. The repository must remain open, updated and inviting to new contributions. The success of this effort lies in maintaining a welcoming and user-friendly environment, ensuring that potential contributors are encouraged to share their expertise and experiences.

Additionally, the repository's ability to adapt its format is paramount. As educational methodologies and technologies evolve, so should the repository's presentation and delivery methods. The proposal of a revised format in this paper is indeed an example of such need to adapt. By keeping the process of format adaptation relatively straightforward, we enable the repository to remain relevant and effective as educational practices and technologies continue to evolve.

References

1. Baker, M., Détienne, F., Burkhardt, J.M.: Quality of collaboration in design: articulating multiple dimensions and viewpoints. In: 1st Interdisciplinary Innovation Conference, Telecom ParisTech (2013)
2. Baumann, K., Campos, J., Dix, A., Nigay, L., Palanque, P., Vanderdonckt, J., van der Veer, G., Weyers, B.: Hci-e2 : Hci engineering education - for developers, designers and more. In: et al., C.A. (ed.) Human-Computer Interaction - INTERACT 2021. Lecture Notes in Computer Science, vol. 12936, pp. 542–547. Springer (2021). https://doi.org/10.1007/978-3-030-85607-6_74
3. Burnett, M.: What is end-user software engineering and why does it matter? In: End-User Development: 2nd International Symposium, IS-EUD 2009, Siegen, Germany, March 2-4, 2009. Proceedings 2. pp. 15–28. Springer (2009)
4. Caffiau, S., Campos, J.C., Martinie, C., Nigay, L., Palanque, P., Spano, L.D.: Teaching HCI Engineering: Four case studies. In: Ardito, C., Lanzilotti, R., Malizia, A., Larusdottir, M., Spano, L.D., Campos, J., Hertzum, M., Mentler, T., Abdelnour Nocera, J., Piccolo, L., Sauer, S., van der Veer, G. (eds.) Sense, Feel, Design. pp. 195–210. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-030-98388-8_18
5. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A unifying reference framework for multi-target user interfaces. Interacting with computers **15**(3), 289–308 (2003)

6. Dittmar, A., Hensch, M.: Two-level personas for nested design spaces. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. pp. 3265–3274 (2015)
7. Dittmar, A., Kühn, M., Forbrig, P.: A domain-specific model-based design approach for end-user developers. In: Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems. pp. 161–166 (2014)
8. GitHub: GitHub Pages (2020), `https://pages.github.com`, last accessed April 19, 2023
9. IFIP WG2.7/13.4: Teaching HCI Engineering Repository (Feb 2022), `https://github.com/IFIP-WG-2-7/Teaching-HCI-Engineering`, last accessed April 19, 2023
10. ISO: ISO 9241-11: Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts. International Organization for Standardization (2018)
11. ISO: ISO 9241-210:2019 Ergonomics of human-system interaction – part 210: Human-centred design for interactive systems. International Organization for Standardization (2019)
12. Jekyll Team: Jekyll blog engine (2022), `https://jekyllrb.com`, last accessed April 19, 2023
13. Ko, A.J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B., Rosson, M.B., Rothermel, G., Shaw, M., Wiedenbeck, S.: The state of the art in end-user software engineering. ACM Computing Surveys (CSUR) **43**(3), 1–44 (2011)
14. Larson, R., Csikszentmihalyi, M.: The experience sampling method. New Directions for Methodology of Social and Behavioral Science **15**, 41–56 (1983)
15. Mackay, W.E.: Educating multi-disciplinary design teams. Proc. of Tales of the Disappearing Computer pp. 105–118 (2003)
16. McCrickard, D.S., Chewar, C.M., Somervell, J.: Design, science, and engineering topics? teaching hci with a unified method. ACM SIGCSE Bulletin **36**(1), 31–35 (2004)
17. Myers, B., Hudson, S.E., Pausch, R.: Past, present, and future of user interface software tools. ACM Trans. Comput.-Hum. Interact. **7**(1), 3–28 (mar 2000). https://doi.org/10.1145/344949.344959, `https://doi.org/10.1145/344949.344959`
18. Oleson, A., Solomon, M., Ko, A.J.: Computing students' learning difficulties in hci education. association for computing machinery, new york, ny, usa, 1–14. In: Proc. of the 2020 CHI Conference on Human Factors in Computing Systems. pp. 1–14. ACM (2020)
19. Rosson, M.B., Carroll, J.M., Rodi, C.M.: Case studies for teaching usability engineering. SIGCSE Bulletin **36**(1), 36–40 (March 2004). https://doi.org/10.1145/1028174.971315
20. Wilcox, L., DiSalvo, B., Henneman, D., Wang, Q.: Design in the HCI classroom: Setting a research agenda. In: Proceedings of the 2019 Designing Interactive Systems Conference (DIS'19). p. 871–883. ACM (2019). https://doi.org/10.1145/3322276.3322381