Using Non-Profit Partners to Engage Students in RE

Birgit Penzenstadler, Debra Richardson
School of Computer and Information Sciences
University of California, Irvine, CA, USA
bpenzens@uci.edu

Beth Karlin
School of Social Ecology
University of California, Irvine, CA, USA
bkarlin@uci.edu

Allison Cook
Story of Stuff Project
Berkeley, CA, 94709, USA
allison@storyofstuff.org

David Callele
Experience First Design Inc.
Saskatoon, Canada
dcallele@experiencefirstdesign.com

Krzysztof Wnuk
Department of Computer Science, Lund University
Box 118, SE-221 00
Krzysztof.Wnuk@cs.lth.se

***Abstract*—To improve requirements engineering education and training, experience reports serve as guidance on how courses can be taught and which methods and approaches work with specific types of audiences.**

One problem when teaching undergraduate audiences is often a lack in motivation for the course because of misconceptions about requirements engineering as well as simple work overload by the curricula or other imposing constraints with regard to the students' time budget.

A way to overcome this motivational problem and to make students want to actively contribute to a requirements engineering class project is to let them perform a case study on a socially relevant system in collaboration with an industrial partner.

This paper provides an experience report of such an in-class project on an online-learning platform for civic engagement in cooperation with the Story of Stuff Project. We provide the structure of the course as well as the assignments, excerpts from the students' results, and observations made throughout the course, all of which may serve as input for other instructors.

***Index Terms*—requirements engineering; requirements education; environmental sustainability; civic engagement;**

I. Intro

Requirements Engineering (RE) as a discipline provides the foundation to making successful software systems by eliciting the appropriate information from the relevant stakeholders and by offering the methodological means to analyze and document the findings such that they can be incorporated throughout design and implementation.

In the undergraduate course on RE at the University of California, Irvine, we want to integrate the theory in class with as much real-world experience as the classroom constraints allow for. Therefore, we integrate case studies and examples from ongoing software system development and collaborate with industry and/or non-profit partners to give the students an impression of RE in practice.

Web content has become an increasingly common way of disseminating important information to people and engaging them in social causes [12]. The challenge of educating people online in civic engagement, as developed by the Story of Stuff Project[1], provides a well-suited example case study for students to learn these techniques and to experiment with different ways of achieving a consensus among a variety of stakeholders from a non-engineering background.

With the majority of our students being interested and responsible citizens (as emerging data suggests that Millennials are interested in doing more purpose-driven work or just being "socially-conscious" [5]), we predicted that using a case study on empowering citizens would engage them more thoroughly with the responsibilities of requirements engineering.

Contribution and Outline: This paper reports on experiences gathered with using the Citizen Muscle Boot Camp (CMB), an online learning program currently under development at the Story of Stuff Project for environmental activism, as student project in a requirements engineering course for undergraduate students at the University of California, Irvine. The goal of the Boot Camp is to train a subset of the 500,000 members of the Story of Stuff community into citizen activists to work toward social change on sustainability issues. After introducing the project, we present the course details, how the students performed the assignments, and the observations from their results.

II. Background

A. The RE Undergraduate Course at UCI

The undergraduate course in RE at the University of California, Irvine, has been taught by a number of different lecturers with each slightly adapting the style and focus to personal preferences and current research and practice. In the spring quarter of 2014, it was taught by the first author of this paper.

One of the definitions for RE that is used in the course is the one by Nuseibeh and Easterbrook [13]: Requirements engineering is concerned with interpreting and understanding stakeholder terminology, concepts, viewpoints and goals. Hence, RE must concern itself with an understanding of beliefs of stakeholders (epistemology), the question of what is observable in the world (phenomenology), and the question of what can be agreed on as objectively true (ontology). Such issues become important whenever one wishes to talk about validating requirements, especially where stakeholders may have divergent goals and incompatible belief systems [13].

[1] http://www.storyofstuff.org

Furthermore, RE facilitates the process of consolidating different stakeholders' concerns and agreeing on a system vision [11].

The major learning goals that arise from this view and shall be incorporated in the course are the following:

1) Knowledge areas: RE terminology RE contents, RE principles and methods for eliciting, analyzing, specifying, validating, verifying requirements, and quality assurance
2) Competencies and skills: analysis, abstraction, phrasing, communication, sensitivity for customers, method competencies

The course occurs over 10 weeks, with two lectures per week. Student evaluation is performed by a mid-term and final exam as well as a number of team assignments. The assignments are designed to add up to a complete requirements specification, with students working in a team distributing the work and synthesizing and consolidating individual parts as well as taking responsibility for quality assurance.

B. Student Body

The course is mandatory for undergraduate students from three different major subjects, namely Informatics, Computer Science, and Business and Information Management. Due to the fact that it is a required course, there are many students each year who have to participate, and this made for a large course of 106 enrolled students.

All have had programming experience in their prior coursework, while some have had more extensive experience in software development either in their own projects or in internships.

C. Lectures and Course Materials

The lectures were structured along an outline that followed the paradigm of artifact-oriented requirements engineering, i.e. that used the artifact model presented in Sec. II-D as a backbone to elicit and analyze requirements and to organize the other tasks involved with requirements management. Each lecture was 80 minutes long and the lecture slides were available online beforehand.

Preparation materials for the case study were provided in the form of background information on the Story of Stuff Project as introduced in Sec. III-A and Sec. III-B as well as some general background on Massive Open Online Courses (MOOCs) [9] with examples[2]. Students were asked to familiarize themselves with the Story of Stuff material as well as with examples for MOOCs.

D. Artifact-based RE with AMDiRE

In winter 2014, we introduced AMDiRE [7], an artifact-oriented approach to RE, in the course. The basic idea of artifact orientation consists in defining a reference model of all relevant artifacts and their dependencies while leaving open the way of their creation. The focus thus lies on *what* to create rather than on *how* to create it. In RE, there exist several artifact models, such as the one of Berenbach et al. [4, ch. 2], who describe RE artifact modeling with the key components to be a measurable reference model and respective process guidelines. To tackle the problem of a blurry terminology and to foster the discussions about this paradigm, Mendez et al. introduced a meta model for artifact-based RE [10]. This meta model defines the basic concepts of artifact-based RE, i.e. which elements are necessary to define an artifact (structure, content), or how an artifact relates to further software process concepts like "method" or "role". This supports the systematic creation of artifact-based RE approaches covering all elements of software processes, and thus the integration and customization of an artifact-based RE as part of a software process.

The Artifact Model for Domain-independent Requirements Engineering (AMDiRE) [7], [2], developed on the basis of that meta model, served as framework for the lectures and structured the team project into weekly assignments.

E. Related Work

In the field of sustainability education, Karlin et al. [8] report on a pilot study of the Guided Research Applied Sustainability Project (GRASP) model for sustainability education that provides students with a positive and engaging learning experience.

In RE education, Zowghi and Paryani [16] used role playing to bring real world experiences into the classroom. Penzenstadler and Callele [14] report on experiments with bringing a stakeholder from mechanical engineering into class. Barnes et al. [3] discuss how to foster an attitude of acceptance in their students versus resistance and fear of the unknown and unknowable in RE by including real-world examples and experience in class.

Penzenstadler et al. [15] report on bringing stakeholders from industry into the classroom. In contrast, the course described in the paper at hand did not include interviews with the stakeholders in class but instead information via documents and online research.

III. SoS CMB Project

This section describes the problem domain, the study system, and the student assignments.

A. Problem Domain: The Story of Stuff Project

Story of Stuff started as a film project in 2007 designed to tell the story of "stuff (i.e. consumer goods) from its creation, through its sale and use, and eventually to its disposal" in a catchy and engaging manner. The single film, with an initial viewership goal of 50,000 views, quickly exceeded this goal and sparked the launch of The Story of Stuff Project as a 501(c)3 nonprofit organization. The Story of Stuff Project has since created 8 additional films and has also translated into other mediums, including television (interstitials for PBS), print (Story of Stuff book), online (website, social media, podcast), and curricula (K-12 and religious groups). Their combined films have been translated into 39 languages and

[2]https://www.coursera.org/

seen by over 40 million people and counting. This viewership has also translated into an active online community of nearly a half million people. The initial goal of the Story of Stuff Project was to create and provide media resources to environmental activists and educators as well as to the general public. A community of potential activists formed around the organization, who started to want to become more actively engaged in the issues discussed in the films.

B. Study System: The Citizen Muscle Boot Camp

To address this need for positive citizen engagement through education, The Story of Stuff Project created a new program in 2013 called the Citizen Muscle Boot Camp. The Citizen Muscle Boot Camp (CMB) is a six-week, online course designed to provide Story of Stuff Project community members with the skills, motivation and peer support they need to act on issues related to environmental sustainability. It was designed in response to inquiries and requests from the Story of Stuff community to help members learn, engage, connect, and act on the issues raised in their films. The CMB guides participants through a series of weekly trainings aimed at strengthening their civic activism skills, or "citizen muscle". Each week of the program focuses on a unique skill:

1) Purpose: discovering your change making style and goal
2) Talk: learning how to communicate effectively about your issue
3) Grow: finding and developing a community of allies
4) Focus: getting strategic about how to accomplish your goals
5) Push: figuring out which tactics you'll need to employ to effect change
6) Practice: putting your learning into action.

For a first pilot of the course, members of the Story of Stuff community were contacted via email and invited to register for the CMB free of charge. Those who registered were enrolled in the CMB and received an email with a link to a 2-3 minute video lesson, accompanied by additional resources (e.g., framing, tips, readings) to get them practicing their change-making skills and a homework activity to put their new skills into practice. The CMB is designed so that each weekly module can be completed in 1-2 hours per week, and it was chosen as study system for the students to complete their assignments and develop a requirements specification.

C. RE Course Assignments

This section provides the assignments for the students as well as the evaluation criteria that were used to grade the submissions. The students were divided into 26 teams of four to five students. They were free to choose their teams within the first week of the quarter and the students who had not found teams by then were assigned to one. Each team was serving as customer for one of the other teams, and as a developer for a different team. The timeline of the assignments is depicted in Table I.

We used a reduced version of the AMDiRE artifact model as depicted in Fig. 1.

TABLE I
TIMELINE OF ASSIGNMENTS AND GRADING SCHEME

Business Case (5%)	Jan 27
Stakeholder Model (5%)	Jan 27
Goal Model (5%)	Feb 3
System Vision (5%)	Feb 3
Mid-term exam (20%)	Feb 6
Domain Model (5%)	Feb 10
Usage Model (5%)	Feb 10
Non-functional requirements (5%)	Feb 17
Functional hierarchies (5%)	Feb 24
Peer review (5%)	Mar 3
Presentation of specification (5%)	Mar 10
Final Exam (30%)	Mar 20

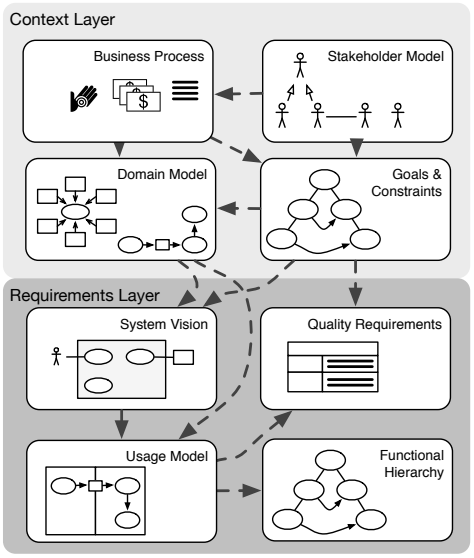

Fig. 1. Reduced Version of AMDiRE for Course Assignments

1) Business Case Analysis and Stakeholder Model: For this assignment, the students had to perform an elicitation meeting with their team's customer group to discuss the case study problem in their role as part of the development team. After the elicitation meeting where they discuss with the customer their understanding and vision of the Story of Stuff Citizen Muscle Boot Camp Online Course, the team develops a business case analysis and a stakeholder model, both as described and discussed in the lecture.

Evaluation criteria:

- Business Case Analysis [40 points]:
  - Is the analysis structured well?
  - Does it contain executive summary, problem statement, analysis, solution options, project description, cost-benefit analysis, risks and recommendations?
  - Is each of these elements described in sufficient detail?
  - Is the information consistent throughout the document?
- Stakeholder Model: [40 points]
  - Have all major stakeholder groups been considered?

  - Have they been analyzed and described to an adequate degree of detail?
  - Are the relationships between the stakeholders clear?
  - Is it clear which stakeholder has which knowledge, skills, priority, and responsibilities?

2) Goal Model and System Vision: For this assignment, the students had to ensure that they captured the critical goals from their elicitation meeting with the customer. If this is not the case yet, they have to further communicate with them and elicit more goals. On that basis, the team develops a goal model and a system vision, both as described and discussed in the lecture. They had to make sure that the goal models contain at least three to four levels of subgoals for more than half of the goals[3] and that appropriate notations are used (goal categories, labeled relations, AND/OR alternatives). The task was not only to submit the diagram but also a textual explanation of how they did it and why they did it this way. This also gives them a chance to report on encountered challenges.

For the system vision, the scoping (system boundary) is important, as well as other systems in the context and the involved stakeholders. It was important to keep the intention of a system vision in mind: It shall communicate the idea of the project to all stakeholders in a way they can agree on and that is easily understandable without technical knowledge.

Evaluation criteria:

- Goal Model [40 points]
  - Is the goal model well structured?
  - Does it contain at least five business goals, five usage goals and five system goals?[4]
  - Are they broken down and refined into subgoals where possible?
  - Is each of these elements related sufficiently to the other goals and are all notations used?
- System Vision: [40 points]
  - Is a clear system boundary / scope visible?
  - Is the vision described and illustrated to an adequate degree of detail?
  - Is it clear which systems are in the operational and business context?
  - Is it clear which stakeholders are involved?

3) Domain Model and Usage Model: For this assignment, the students use the input from the earlier content items (business case analysis, stakeholder model, goal model, system vision) including the feedback they received on those to develop a domain model and a usage model, both as described and discussed in the lecture. It was encouraged to make sure that the domain model contains the important concepts of the domain — business objects, real world objects, and events that transpire — in form of classes (at least seven), attributes (at least two per class), and associations with roles and multiplicities. Furthermore, it was again requested to not only submit the diagram but also a textual explanation of how they accomplished the task. For the usage model, they were asked to provide an overview diagram of all use cases, plus a detailed version of at least one use case, including the full information from the template in the lecture slides. The detailed version of one use case includes one scenario — first described according to the Cockburn template [6] (as main success scenario with extensions and/or variations)—and then choosing one diagram type (activity diagram or message sequence chart) to illustrate it. They were asked to provide a description of the rationale for the domain model and usage model, at least two paragraphs of how they did it and what they found difficult or the most challenging aspect of it.

- Domain Model [40 points]
  - Is the domain model well structured?
  - Does it contain at least 7 classes and 2 meaningful attributes per class?
  - Are all classes connected by associations with roles and multiplicities?
  - Is the domain model complete w.r.t. the criteria discussed in class? Does it provide all important concepts?
- Usage Model: [40 points]
  - Is a use case overview diagram provided that is well structured and includes all important use cases?
  - Are all use cases that are important for the system depicted in that diagram?
  - Is a complete and correct description of one exemplary scenario provided for one central use case in either a UML activity diagram or a message sequence chart?
  - Is a complete and correct description provided for one central use case in a table (according to the Cockburn template)?

4) Non-functional requirements: For this assignment, the students again had to use input from the earlier content items (goal model, domain model, usage model) including the received feedback to develop a small set of non-functional requirements. This should be provided in the form of two examples of each of the following categories: process requirements, deployment requirements, system constraints, and quality requirements. These requirements shall be refinements of the earlier elicited goals, so they were asked to name the goals as rationale.

Evaluation criteria for non-functional requirements [80 points]:

- Is the template filled out completely and correctly for process requirements?
- ... for deployment requirements?
- ... for system constraints?
- ... for quality requirements?

5) Function hierarchy: For this assignment, the students again had to use the input from the earlier content items (usage model and scenarios). On that basis, the team develops a function hierarchy according to the example from the lecture,

[3]To ensure that students practice refinement.

[4]The number is defined arbitrarily, but students kept asking "how much was enough".

structuring them into functions and subfunctions and adding the respective relationships between them.

Evaluation criteria for the Function hierarchy [80 points]

- Are there at least ten functions in the function hierarchy?
- Does the structure of the hierarchy make sense?
- Are at least three of the possible types of relations (precedes/follows, requires, interrupts, activates/deactivates) made explicit?
- Are all relations depicted that exist between the displayed functions or are relations missing?

6) Quality Assurance in Peer Review: For this assignment, students used the input provided by their development team, meaning that:

1) They send their complete specification to their customer in one PDF file (consolidated version of all your assignments in one document) and receive the one from their development team.
2) The team writes up a review of the specification of their development team according to the following IEEE-830 criteria: completeness, consistency, unambiguity, correctness, structuredness, traceability, changeability, understandability

Evaluation Review [80 points]: Is there a rating and rationale for the rating for the above criteria, namely Completeness, Consistency, Unambiguity, Correctness, Structuredness, Traceability, Changeability, and Understandability?

IV. Results & Observations

All teams completed the assignments and handed in their solutions. They also provided the rationale on how the results were developed and which challenges had been encountered while performing the assignment. This section provides illustrative examples of good solutions (not all from the same team) and a discussion of observations made throughout the course and while reviewing the results.

A. Business Case

Figure 2 is a screenshot of the rather elaborately designed business case brochure one of the teams created. The hardest part in the business case was that the system was for a non-profit organization. This led some of the student teams to inventing options for voluntary donations while taking the online course, but made it hard to develop an actual business case with convincing numbers apart from trying to keep the costs low. The business case in Fig. 2 is a good, concise solution. While the layout effort is quite impressive, that part of the effort could have gone to something more directed to the task at hand. Constraining the students to focus on content over presentation seems to be useful in such an early and short course. However, this is one of the three "fancy" examples out of 26 solutions, so most of the students did stick to a simpler presentation format. Apart from that, students put a lot of effort into researching some background statistics on the web that would back up their data in the business case.

The challenge of finding sufficient background information is not so much related to learning the actual techniques of

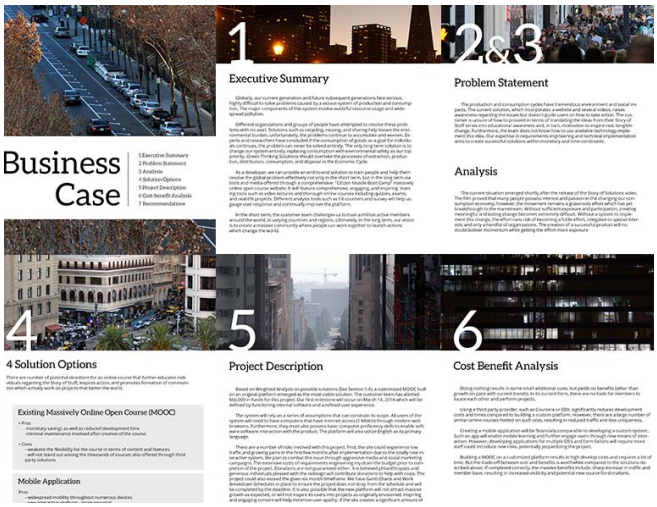

Fig. 2. Business Case Outline for the SoS CMB

requirements engineering, but very much in the day-to-day practice of business analysts who also perform requirements engineering.

B. Stakeholder Model

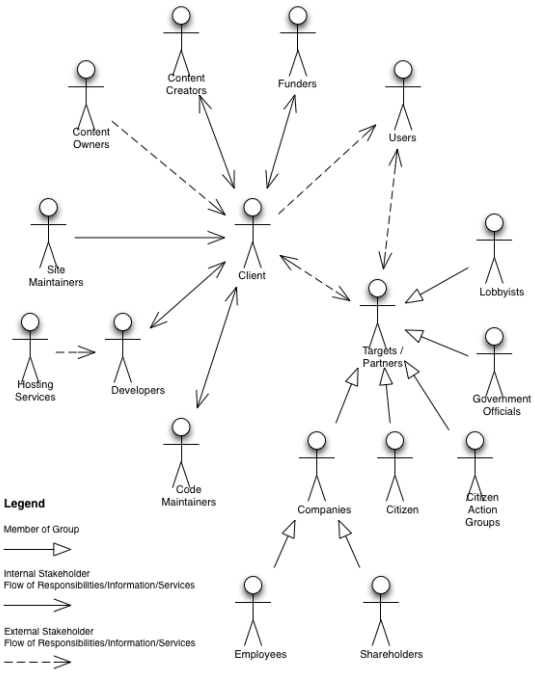

Fig. 3. Stakeholder Model for the SoS CMB

The stakeholder model in Fig. 3 is a simple yet sufficiently extensive representation of the major stakeholders of the CMB system. Students often neglected representing the relations between stakeholders, as for example in Fig. 3, where there are arrows to "hold together" the figure but no labels on the

arrows that would indicate the actual relation between the stakeholders.

From the discussion on the solutions afterwards, the major challenge was to infer relations if the checklists and examples provided in class didn't already include those relations.

C. Goal Model

The goal model in Fig. 4 depicts business goals (in red), usage goals (in green), and system goals (in blue) and their relations. The model is structured to support a goal hierarchy with respect to the scope of the goal. Business goals are more primary, relating to the actual purpose of creating the system, so they are located toward the top of the diagram. They are supported by usage goals which define the intent and function of the system and are thus supported by system goals which demonstrate the characteristics of the system. The model contains a multitude of antigoals, or constraints, as well as obvious goals. Anything that is not marked with the double minus sign is an uninhibiting subgoal of its parent goal. Each constraint is marked with a double minus sign (- -) meaning it can inhibit the functionality or achievement of it's parent goals. Double plus sign (++) means the subgoals lead up to their parent goal.

Major challenges were to distinguish between business, usage, and system goals and to identify relations between the goals.

D. System Vision

Figure 5 is a pictorial representation of the system vision of the CMB that all stakeholders (in this case, customer team and developer team members) agree on and that serves as common reference point in discussions. The system vision was agreed upon by all stakeholders (i.e., each developer team and the respective customer team) in order to define the functionality and characteristics of the Story of Stuff CMB. The vision contains passive and active stakeholders centering around the CMB and a business and operational context, separated by a dotted line on the diagram. Most active stakeholders are highlighted by colored fields. Community leaders and users are among the active stakeholders, but they are not members of the Story of Stuff organization. Each group is highlighted according to the legend on the diagram. Passive stakeholders, along with community leaders and users, are not highlighted by any boundary.

The major challenge with the stakeholder model was to consider all categories of stakeholders and to determine influences between them.

E. Domain Model

The domain model in Fig. 6 is a class diagram that lists the most important business (real-world) objects that have to be represented in the system. The classes and attributes needed to not only follow what was written in the business case and the goal model, but also had to be consistent with the usage model. The classes are associated with each other when they have direct interaction with each other.

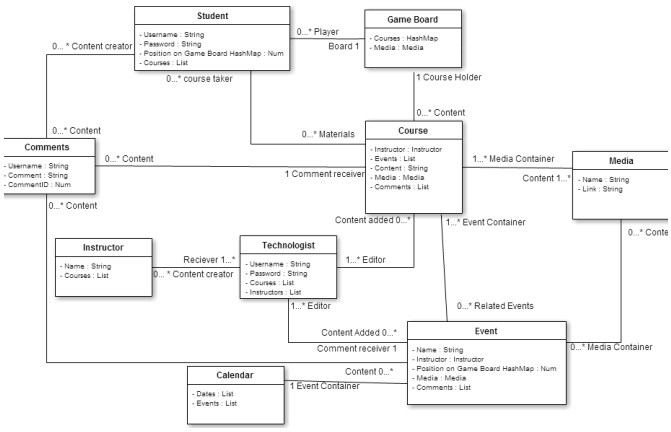

Fig. 6. Domain Model for the SoS CMB

The students encountered many challenges including but not limited to: having consistent terms between the domain model and previous documents or models; determining whether classes were necessary or arbitrary; and figuring out if an attribute was part of the correct class. They resolved the terminology issue by discussing the model with their teammates and coming to a consensus as to which terms to be used in the model. As for the classes, they decided which ones should become classes and which ones should remain attributes while creating the diagram.

F. Usage Model

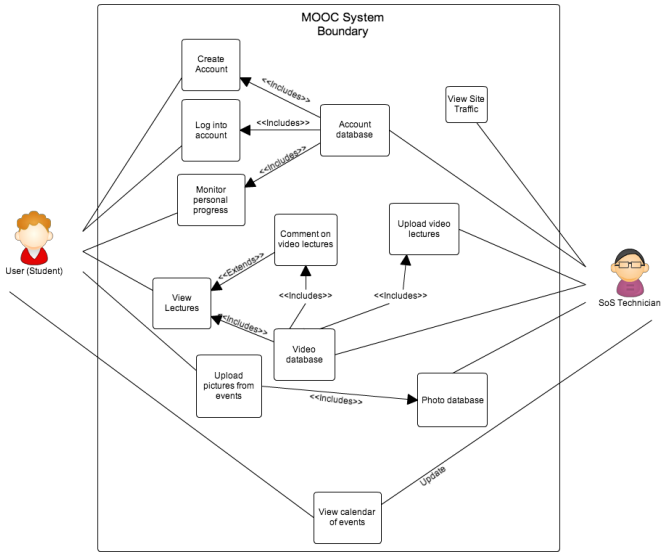

Fig. 7. Use Case Overview for the SoS CMB

Figure 7 and Figure 8 depict excerpts of the usage model, representing different paths of how a user can interact with the website. In the activity diagram (Fig. 8), the user is placed in the middle as a way to symbolize their role in determining whether or not data would be transmitted between the system

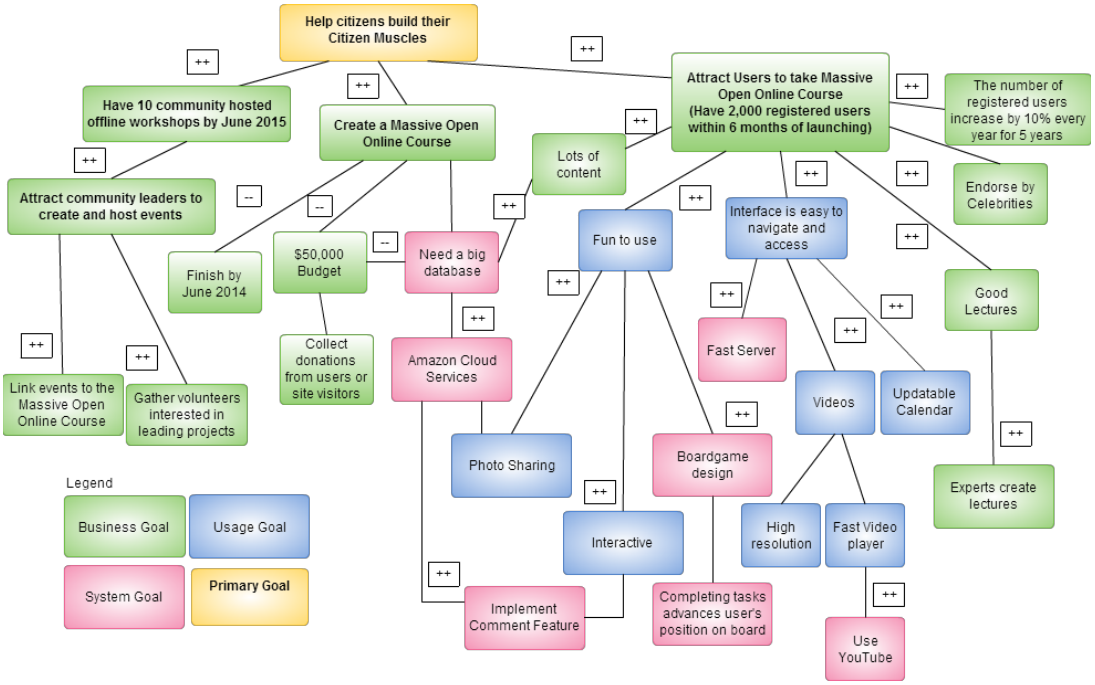

Fig. 4. Goal Model for the SoS CMB

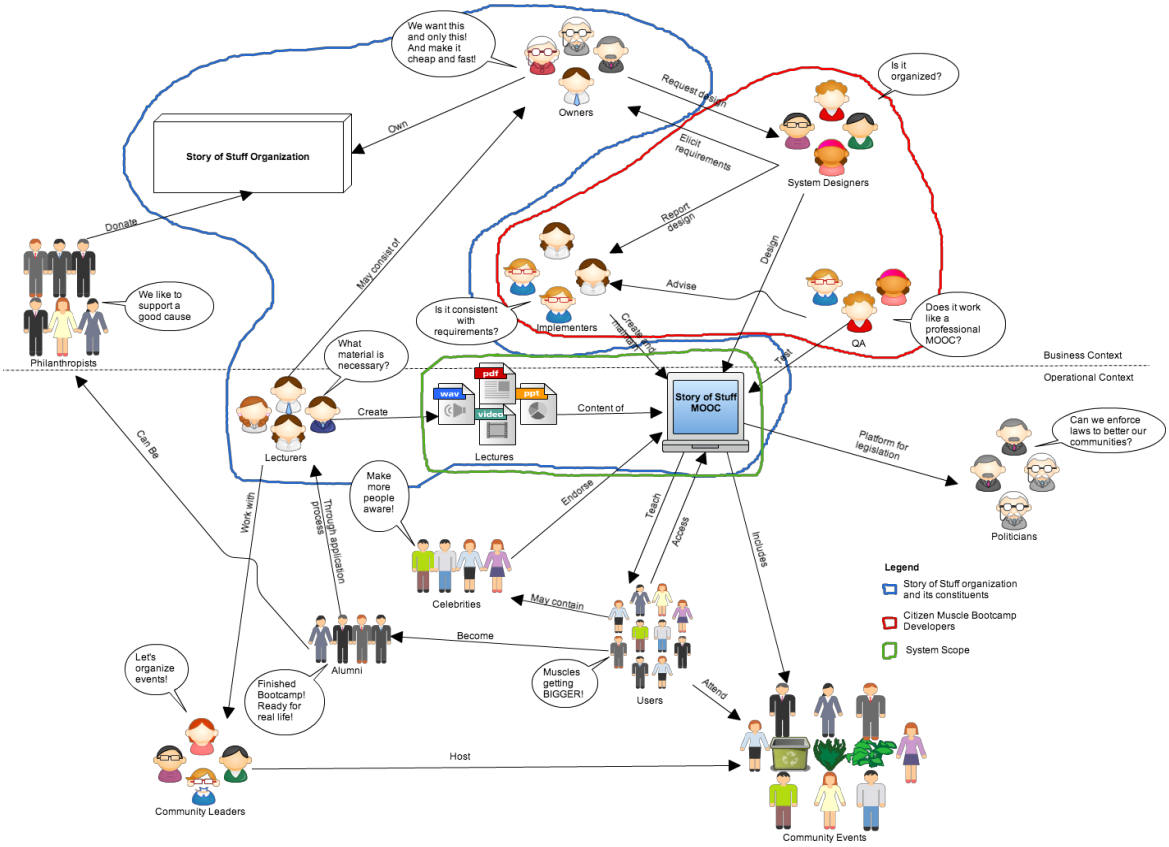

Fig. 5. System Vision for the SoS CMB

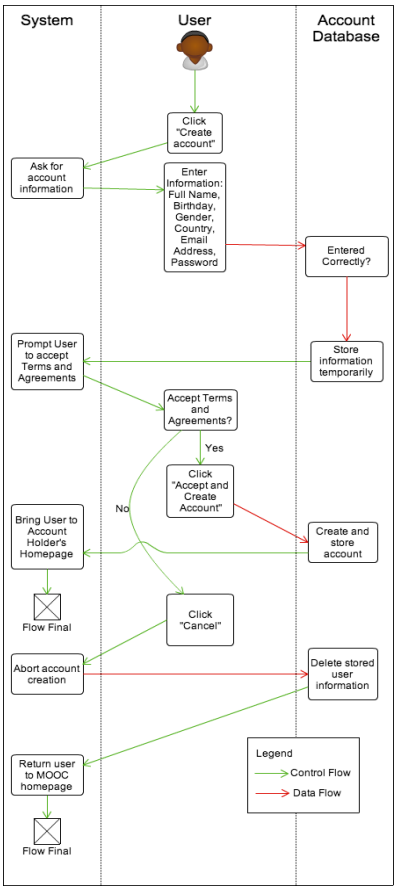

Fig. 8. Use Case Scenario as Activity Diagram for the SoS CMB

and the account database. The students also considered the failure condition, in which an account is not created. Including these two end conditions is necessary to demonstrate the destruction of data upon failure.

The most challenging aspect of designing activity diagrams was determining when and where data flow occurs, as the students had difficulty in defining the interaction between the system and the accounts database.

G. Non-functional requirements

NFR	All code must pass code review
Rationale	Contributes toward stability goals
Satisfaction criterion	Later code refactoring
Measurement	Percentage of code that has been refactored
Risk	Site may suffer unintended downtime

NFR	The development team must use open-source tools
Rationale	Cheap, easy to replicate for future use, transparency
Satisfaction criterion	All development tools are open-source
Measurement	Review of development tool copyright licenses
Risk	Tools used will be of lower quality than they might be otherwise

NFR	Server must allow for the export of data
Rationale	Allows data to be withdrawn and analyzed
Satisfaction criterion	Settings export coverage
Measurement	Percentage of hosting settings that are saved in export
Risk	The site would be impossible to transfer to another host

NFR	Site must be up consistently
Rationale	The website should be fully functioning so that users will not be turned off by the idea that website is always defective.
Satisfaction criterion	Site uptime must be greater than 99.99%
Measurement	Randomly pinging the site and counting percentage of responses
Risk	Users may leave to other MOOCs and dismiss our website as a desirable location for online education.

Fig. 9. NFRs for the SoS CMB

The non-functional requirements in Fig. 9 provide examples of a simple template specification form used to detail the NFRs and their validation. The non-functional requirements included quality requirements, process requirements, deployment requirements, and system constraints.

The students reported slight difficulty in eliciting non-functional requirements, along with phrasing them in a way that was coherent with the goals. They referred mostly to their goal models, breaking down each element in order to determine some non-functional properties their client prescribed for the system. The NFRs themselves were rather high-level quality goals broken down to a level that could be applied to the overall software system. The measurement (see Fig. 9, 4th attribute of every table) was often underspecified so that testers would have to make assumptions about how exactly to measure this satisfaction criterion.

H. Functional hierarchies

The functional hierarchy in Fig. 10 decomposes the user-perceived functionality from a system point of view and thereby facilitates the transition to design. While the students listed out each function, they tended to think too broadly and in general terms about the CMB, instead of the specific requirements that were requested by the clients. They found it difficult to determine the subfunctions of certain parent functions and figuring out whether *activates*, *precedes*, or *requires* was more suitable for describing how certain functions interacted with each other in the hierarchy. Another problem that they encountered was making the functional hierarchy diagram readable. The user was required to go through the "signin to account" function before being allowed to use the rest of the functions in the MOOC system. It was the subfunction to most of the other functions, so there were many arrows pointing to that function and many boxes surrounding it. As a result, they chose to make the "signin to account" function "activate" all the other functions, which solved the problem and made the diagram much easier to read.

Early design decisions like this were the most discussed point about function hierarchies and show where RE and design have to be integrated or may overlap.

I. Peer Review

In the peer review, students tended to be very generous to their developer teams. They commented on a few minor mistakes or how the specifications could have been extended, but in general they were satisfied with the work of their peers. Most of the teams had gone through the effort of reworking their specifications after the initial feedback from the teaching assistants, so those efforts apparently had paid off, as the specifications had improved considerably.

J. Presentation of the Specification

The teams presented excerpts of their solutions within five minute presentations during the last two sessions of the course before the final exam. At this stage of their curriculum, the students do have experience in presenting in front of a large

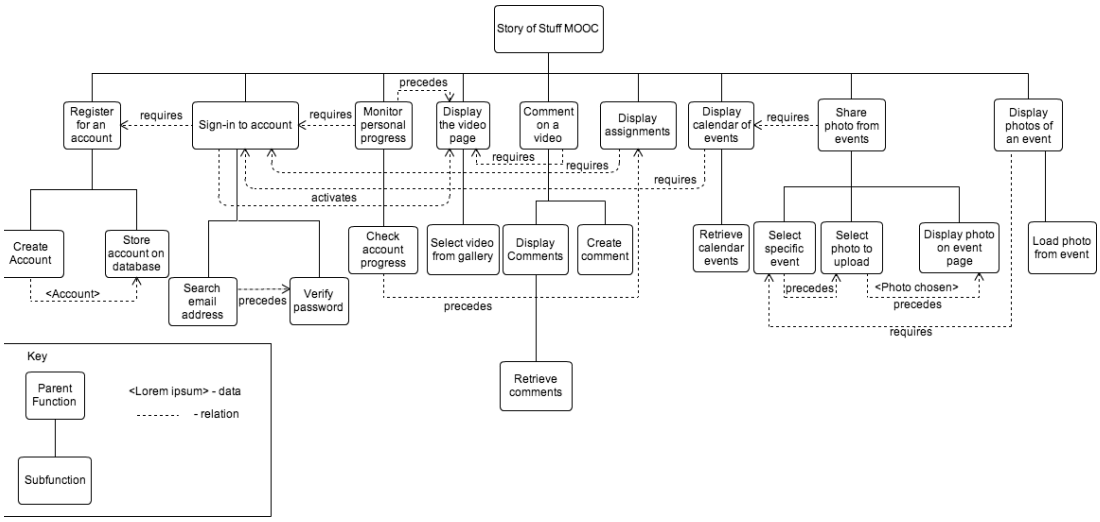

Fig. 10. Function Hierarchy for the SoS CMB

audience, but their presentation skills varied considerably. While some teams made an effort to make their presentation engaging, others only put in the minimum required effort to deliver an acceptable summary. Presentation skills are of critical importance to successful customer communication and should therefore be part of an RE course.

V. Discussion and Lessons Learned

A. Overall Quality of Results

The range of solutions extends from bad to good, as well as from little effort to major effort that was put into the development of the specification.

The grades in the mid-term and the final exam were average for a RE course, not much difference was perceived from other years (comparison as perceived by teaching assistant) or from earlier teaching experience by the first author at other universities. However, this is not dependent on the teaching method but on the students' population — badly designed course will generate failures but well designed courses may also generate failures if students are not motivated.

B. Team Dynamics

There is a strong correlation between the students level of experience and the quality of the work that they produced on the project. Students who had experience in software development projects and even worked in industry managed their teams in a very effective way and provided medium to high quality specifications. On the downside, they often strongly focussed on the technical aspects of the system (solution-orientation) and neglected stakeholder communication (problem-orientation).

There was a perceivable difference between the groups that were self-selecting and the groups that were assigned in terms of the work they produced. Teams who had self-selected were usually composed of friends who therefore had established communication habits, while new teams first had to get to know each other and find common denominators.

C. Solution Space

The design space used by the students was rather limited compared to what will actually be implemented. Were the students simply not interested enough or not knowledgeable enough? Or were the re-documenting what they saw elsewhere without considering other options? This depends partially on their experience, but also on their creativity and motivation for coming up with new ideas in this course.

Having chosen a problem domain that would likely increase motivation (see Sec. III-B), we were interested in how much sustainability showed up in their results. The answer is: only in System Vision and Goal Model, if at all. Some even reduced it only to the economic sustainability of the business. It is hard to determine whether this is due to the lack of familiarity with requirements techniques, due to lack of interest in the specific concern for the system under development, or due to a lack of knowledge of sustainability issues. Other disciplines that are also trying to incorporate questions on sustainability into existing structures, like UC Santa Cruz's engineering program [1] have reported a high motivation amongst students for sustainability causes, therefore we conclude that this is mainly due to inexperience of how to factor such a value into RE.

D. Lessons Learned

This section sums up the lessons learned of the experience report at hand.

Reduce Number of Assignments: Effort estimation in student projects is hard, but as first-timers every artifact takes more time to elaborate and team work requires a lot of discussion if done properly. The conclusion is to ask for less artifacts when

the course has only 10 weeks—it was too much stuff to deal with in such a short time—and to probably cut out function hierarchies because they are strongly design-oriented.

Real Stakeholders Motivate: While the students liked having a "real system" to work on, they would have preferred to talk to a real person involved in the project as opposed to having to rely on documents, online research, and future users. Consequently, for the next iteration of the course, we will again put a real stakeholder in class (compare to the BMW DriveNow case study [15]) for one or two interviews during elicitation and analysis, even if they are playing a role. According to our experience, a real person who is only available for that specific time, considerably increases the motivation to understand the problem domain and system under development.

Choice of Problem Domain: There are challenges when motivating the pragmatic topic of RE with a project that is, itself, motivated by what might be perceived as an ethical choice. Does the ethical choice resonate with the students (positive motivator), is it ignored (neutral, but not achieving your personal goal to enlighten them), or does it annoy them? When looked at from a software system perspective, RE is about the content to be represented in the system, not the system purpose. The question of whether or not the content of the project enhances or detracts from the mechanics of the course material is hard to answer without back-up by significant empirical data. It was perceived that the students were enjoying the topic of the problem domain but there might have been students who were negatively motivated by the topic without providing feedback on that. Next time we will include a question on this in the course evaluation survey.

Value-based Design: Sustainability is not a concern that visibly showed up in the solutions other than in the goal model and system vision. Environmental sustainability was used as example for a value that could be supported in value-based design. While it was not the intention of the course to promote any political agenda, this was a value that students would generally be willing to accept and that was a little easier to grasp than a value like "fun". The fact that the value was not made very explicit in the solutions shows that for practicing any kind of value-based design, this needs to be put into an exercise more explicitly, for example by making different teams design for different values and comparing the differences amongst their solutions.

VI. Conclusion & Future Work

This paper reported on the experiences with using an online course, the Citizen Muscle Boot Camp, under development at the Story of Stuff Project, as study system for the assignments in an undergraduate RE course at the University of California, Irvine. The major lessons learned are that artifact-oriented RE works well to structure assignments, but also needs to be restricted to the given time frame of the course, that real-world systems do motivate students, but even stronger motivation and commitment to developing a good requirements specification can be achieved by inviting real stakeholders, and that the choice of the problem domain might influence students' motivation. There are a few follow-up questions to be explored in future work, for example, whether value-based design should be included as a concept when teaching requirements engineering, and how significantly the choice of problem domain affects the motivation of the students in such a course. For the next iterations of the course at UCI, we will try to allow the students to perform interviews with real stakeholders for a system under consideration in practice.

Acknowledgements

This work is supported by the DFG EnviroSiSE project under grant number PE2044/1-1.

References

[1] Patricia Allen and Martha Brown. Sustainable Agriculture at UC Santa Cruz. http://casfs.ucsc.edu/about/History%20and%20News%20Archive/sustainable-agriculture-at-ucsc.html, 2014.

[2] D. Mendez Fernandez B. Penzenstadler, J. Eckhardt. Two replication studies for evaluating artefact models in re: Results and lessons learnt. In *Proc. of the 3rd International Workshop on Replication in Empirical Software Engineering Research (RESER '13), IEEE, 2013. Baltimore, USA*, 2013.

[3] R.J. Barnes, D.C. Gause, and E.C. Way. Teaching the unknown and the unknowable in requirements engineering education. In *Requirements Engineering Education and Training, 2008. REET '08.*, pages 30–37, Sept 2008.

[4] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer. *Software & Systems Requirements Engineering: In Practice*. McGraw-Hill, Inc., 2009.

[5] Emily C Bianchi. Entering adulthood in a recession tempers later narcissism. *Psychological science*, page 0956797614532818, 2014.

[6] Alistair Cockburn. *Writing effective use cases*. Pearson Education, 2001.

[7] Daniel Mendez Fernandez and Birgit Penzenstadler. Artefact-based Requirements Engineering: The AMDiRE Approach. *Requirements Engineering*, 2014.

[8] B. Karlin, N. Davis, and R. Matthew. GRASP: Testing an Integrated Approach to Sustainability Education. *Journal of Sustainability Education*, Spring 2013(Experiential Education, Part One), 2013.

[9] Rita Kop. The challenges to connectivist learning on open online networks: Learning experiences during a massive open online course. *The International Review of Research in Open and Distance Learning, Special Issue-Connectivism: Design and Delivery of Social Networked Learning*, 12(3), 2011.

[10] D. Mendez Fernandez, B. Penzenstadler, M. Kuhrmann, and M. Broy. A Meta Model for Artefact-Orientation: Fundamentals and Lessons Learned in Requirements Engineering. In D. Petriu, N. Rouquette, and O. Haugen, editors, *Proceedings of the 13th International Conference on Model Driven Engineering Languages and Systems (Models)*, volume 6395, pages 183–197. Springer-Verlag Berlin Heidelberg, 2010.

[11] Andrew Monk and Steve Howard. Methods & tools: the rich picture: a tool for reasoning about work context. *interactions*, 5(2):21–30, 1998.

[12] Norman H Nie and Lutz Erbring. Internet and society. *Stanford Institute for the Quantitative Study of Society*, 2000.

[13] Bashar Nuseibeh and Steve Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 35–46. ACM, 2000.

[14] Birgit Penzenstadler and David Callele. Prototyping re experiments in the classroom: An experience report. In *Requirements Engineering Education and Training (REET), 2010 5th International Workshop on*, pages 7–16. IEEE, 2010.

[15] Birgit Penzenstadler, Martin Mahaux, and Patrick Heymans. University Meets Industry: Calling in Real Stakeholders. In *26th IEEE-CS Conference on Software Engineering Education and Training*, 2013.

[16] Didar Zowghi and Suresh Paryani. Teaching requirements engineering through role playing: Lessons learnt. In *Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International*, pages 233–241. IEEE, 2003.