Community Driven Design in Software Engineering

Article · September 2021

CITATIONS
0

READS
32

2 authors:

Mirco Schindler
Technische Universität Clausthal
19 PUBLICATIONS **49** CITATIONS
SEE PROFILE

Sebastian Lawrenz
Technische Universität Clausthal
28 PUBLICATIONS **66** CITATIONS
SEE PROFILE

Community Driven Design in Software Engineering

Mirco Schindler and Sebastian Lawrenz

Clausthal University of Technology, Germany,
mirco.schindler@tu-clausthal.de, sebastian.lawrenz@tu-clausthal.de,
WWW home page: http://www.isse.tu-clausthal.de

Abstract. The implementation of the circular economy as described in the literature is a slow process. Circular economy goals are usually secondary goals of companies. Furthermore, implementing these circular goals is expensive and challenging, especially for small and medium-sized businesses. This leads to a gap in the implementation of the circular economy and obviously hinders progress toward climate protection goals. However, small and medium-sized businesses could achieve their goals together by cooperating and sharing the risk of the implementation of their circular economy goals. Therefore, this paper presents a novel approach towards a circular economy community-driven ecosystem that supports the cooperation of small and medium-sized businesses for implementing the circular economy. Cooperation, in general, requires companies to share their best practices and processes, but this is contrary to their primary goals (their business models). Accordingly, technical solutions are required to allow information to be exchanged without negatively affecting the primary goals, which leads to a conflict between transparency and privacy, to name just one showstopper. We introduce the term and concept of Community-Driven Design in software engineering, we describe it with an applied example, and we use the mechanisms of the Dynamic Adaptive System Infrastructure, so-called DAiSI, to introduce our concept. Based on the DAiSI, and on the community-driven life cycle, we present five steps for achieving a common circular economy target.

Keywords: Circular Economy, Software Ecosystem, Software Platform, Architecture Description, Dynamic Adaptive Systems, Methodology, Software Engineering, Community Driven Design.

1 Introduction and Related Work

The circular economy describes a framework towards a more sustainable future by reducing CO2 emissions, and keeping products, components and materials as long as possible in the life-cycle. In theory, all these concepts are logical and reasonable. However, since the the circular economy concept is already existing since the 1990s, we are still not living in a circular economy as described in the literature [1].

1.1 Problem Statement and Objective

Why is the implementation of the circular economy still lacking? In some previous works, we already identified some main problems, such as an information gap [2] in the circular economy and a lack of knowledge [3]. However, even in a perfect world with a perfect information transparency, there are still *a)* knowledge gap, and *b)* furthermore most small and medium-sized businesses need their human resource for their primary business and objectives. For these companies, circular economy targets are additional or *secondary objectives*. A *secondary objective* describes a desirable target but not the primary goal of a company. Especially circular economy targets such as reducing CO2 emissions, using secondary resources, and so are desirable but associated with high costs. Management prefers to invest in direct profit improvement or research that promotes the primary objectives of its own company. Correspondingly, we are far from a circular economy, as our economic system aims at other targets.

This leads to the down prioritization of these secondary targets and slows down the implementation of the circular economy in its entirety. These down prioritization can be tackled by regulations, laws, the government's funding, or by providing new opportunities and chances, such as a business ecosystem.

Accordingly, the objective of this paper is to stop the under-prioritization of secondary objectives by reducing the investment cost and risk. Therefore, we propose a community driven business ecosystem that enables the possibility to share the investment and risk with other small and medium-sized businesses to encourage and accelerate innovation.

1.2 Community Driven Design

A community is a social unit with a commonality. These commonalities could be norms, religion, values, or something else [4]. The German sociologist Ferdinand Tönnies described a community as a human association where the individual feels part of a larger social whole. Thus he orients his actions to this overriding purpose [5]. A current example of a community such as described before is *Fridays for Future*. Fridays for Future is an association of pupils and students that skip school on Fridays to demonstrate against the climate crisis [6]. Communities exist in software engineering already since some decades, and a good example is demonstrated by the Linux community or shared development projects on GitLab and GitHub. It is time to combine the ideas behind software communities and social communities to enable the circular economy.

1.3 Scenario

We demonstrate this combination using a small scenario in a domain that is not (directly) related to sustainability - underground mining. One of the most significant cost factors in underground mining is energy consumption for pumps and ventilation. So they plan to increase their investment towards more sustainable processes, in line with the circular economy, reduce CO2 emissions, and save

energy. For a single company, especially when it is a small or medium-sized one, this leads to high investment and high risk. However, these secondary objectives are pursued by many companies. Therefore, all these companies decide to invest together in innovations by sharing cost risks, data, and information. They set up a (business) community to achieve their sustainability goals. *However, how can this community cooperate without weakening each other?* The first step towards cooperation could be exchanging data and information since data and information are usually the base to create knowledge and wisdom [7]. Nevertheless, there is growing a high risk that some companies just try to abuse the information to push their primary goals or downgrade the position of one of the other companies. In our example, based on some energy data from a mine, with the help of a skillful deduction, it could be possible to derive information about the amount of mined raw materials, and as a consequence of manipulation, manipulate market prices. As a result of such an incident, all companies withdraw from the consortium, and the secondary goals are put on the back burner again.

To avoid this, we present a concept idea to overcome this hurdle, so companies A and B can cooperate in a secure environment. Because already in the case of two companies, there is the above-described threat of exposing secrets without being sure that it is worth taking this risk. This is comparable to classic contract-based cooperation. Company A and B share the same goal, to reduce their energy consumption.

Since the companies A and B are related to the same domain (in this case, underground mining), they understand their problems pretty well. However, imagine we have a third company C from a different domain (such as a data center) - This company may have similar challenges and problems (high energy consumption for ventilators), but a different way to express this problem. Since all companies use a domain-specific language to express their problems, they may not recognize that they have similar problems and challenges.

The balancing of risk and benefit and thus defining a strategy to maximize one's benefit is a situation in the sense of game theory [8]. A decision-making situation in which several players influence each other. But how can the game be described? - The development of a city, which serves as a metaphor to inspire our approach, and a Circular Economic model are "games" that are not finite but infinite.

The difference between an infinite game in game theory, which is defined by the infinite choice of strategies to win, and the characteristic for the situation described here is made clear by Cares in his book Finite, and Infinite Games [9]. He answers the question of what makes the game infinite by saying that players exist who play intending to keep the game going - not with the intention to win. The non-acceptance of these playing rules of the game leads to the fact that approaches of the classical game theory do not function anymore.

1.4 Outline

The rest of the paper is structured as follows: Section 2 analyzes the problems and challenges derived from the scenario introduced before. Furthermore, Section

2 introduces the relevant foundations and clarifies some crucial terms used in the paper. Section 3 presents our overall concept of a community-driven adaptive software ecosystem, subdivided into five steps, followed by a discussion, explanation of upcoming challenges, and further research directions in Section 4. Section 5 concludes the paper.

2 Problem Analysis and Background

Sam Newman compared the challenges of software engineering once to those of urban planning [10]. Even if this comparison may not fit perfectly, at least the challenges are related to each other. In a city or a district, there are different actors who act independently of each other. They have different professions, act spontaneously with others, and usually pursue individual (primary) objectives. However, they still are interested in the positive development of their district or city (secondary objectives). The section aims to continue comparing urban planning and our scenario and challenges within the circular economy. For this purpose, we first analyze the problems and challenges from our scenario and introduce later the current state of the art and related work.

2.1 Problem Analysis

Based on the scenario introduced in Section 1-C there, we deducted the following two main issues:

1. A common problem understanding is necessary for an efficient community building. Especially when the companies are related to different domains (such as it is the case with companies A and C).
2. There is always a high-risk present, that some individual wants to harm the community, in order to strengthen its own position.

The first point is more a challenge than a disadvantage. But, it is essential to bring people together and to build a community of collaboration. An abstracted goal (improving the city district, enabling the circular economy) is often unambiguous, but understanding the problem and its expression is often different. In the worst case, the collaboration can turn into resistance. Barbara Wilson describes this gradient adjustment in [11] as resilience vs. resistance.

The second point leads more to the risk by starting cooperation. The sharing of data and information could lead to disadvantages. This is a consequence because most individuals do not know what their data can actually do. This lack of awareness could compromise the primary targets of the individual.

In consequence, a community-building approach is necessary that links people with similar problems but also enables secure cooperation under consideration of individual targets and information. In the approach presented here, the focus is on the conceptual aspect of how communities can be formed without exposing all information and secrets; technical mechanisms to protect the platform or components against malevolent attacks are not discussed further.

2.2 Community Driven Design and Software Ecosystems

Barbara Wilson introduces the term *Community-Driven Design* [11] in the domain of urban planning. In contrast to a project characterized by a defined start and end, there is no end in urban development, and neither a clear starting point. The development of a city, or a district, is an infinite activity in a constant state of change. An essential part of Community-Driven Design is resilience, and equity [11].

A software ecosystem is an analogy to biological ecosystems and describes the relation ad the balance between organisms and their environment. The environment influences directly or indirectly the life and the development of the organisms, and vice versa [12]. Software or IT Ecosystems are also based on the balance between individuals (autonomy) and rules (control) that define equilibria within an IT ecosystem. The maintenance and continuous development of IT ecosystems requires a deep understanding of this balance [13], [14], [15], [16].

As mentioned before, a software ecosystem has the goal of creating a balance between the interests of the participants. This leads to the fact, that a (software) ecosystem tries to fulfill multiple objectives. On the other hand a community, has one central objective - in an urban environment, the community is build up due to the geographical proximity. The implementation of the circular economy, or sustainable objectives in general, cannot be implemented as a consensus. Accordingly, a community-driven design in software engineering is necessary.

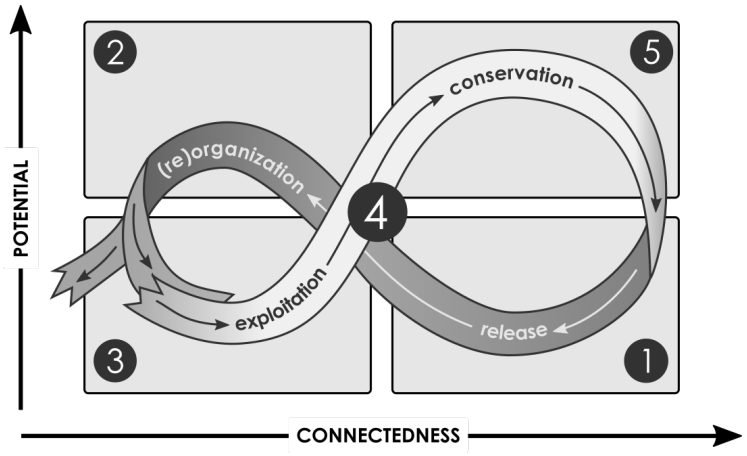

Fig. 1. Community-driven Software Design Lifecycle (based on [11]), [17]

Figure 1 shows an overview of the community-driven (software) lifecycle (see [11], [17]), adopted for our purpose to enable the circular economy by building up new communities. The lifecycle consists of four different phases or five events:

1. *Release and Production*: The Status Quo - describes the current situation, with the degradation of the actual processes. Furthermore, new problems, players, and challenges are coming up, as well as potentials for better solutions.
2. *(Re)organization*: In this phase, the players understand their problems and build new connections to solve these problems together. This phase is the matchmaking phase of individuals with similar problems.
3. *Exploitation*: Here, the problems will be solved together by combining best practices and finding innovations.
4. *Solution Development*: The solutions will be tested and applied.
5. *Conversation*: The developed solutions will be established. It is the transformation towards the new processes.

The numbers given in Figure 1 can be mapped directly to the steps in the approach in Section 3.

2.3 Related Work

Before introducing the overall technical solution concept, we shortly introduce some related work as a base for our solution.

Domain-Driven Design is a way for a structured software model and development process, developed by Eric Evans. The modeling of the software is significantly influenced by the technicalities of the application domain [18].

A *Role and Process Model for a Domain-Driven IT Ecosystem* are introduced by Schindler et al. in [16] and [15] for an underground mining ecosystem. They describe an organizational framework centered on the integrator's role, responsible for technology integration in terms of development and operation (DevOps, see [19]) concept. At the same time, this role is also to be understood as a mediator between the individual stakeholders. Another concept they introduce to drive the more traditional development structures in this domain towards more agile methods is the application of micro-projects.

Self-adaptive Systems consist of different independent components. These components can collaborate and organize themselves in autonomous and self-adaptive ways. A self-adaptive system encompasses an environment where the system itself can modify its behavior and/or structure according to its perception of the environment and the system itself [20]. Our technical approach is especially based on the previous work of our institute around the DAiSI (Dependable Dynamic Adaptive System Infrastructure). DAiSI is an infrastructure which is capable of wiring dynamic adaptive systems from a set of components in order to provide a dynamic and adaptive behavior to the user [21], [22]. (User) requirements can be specified based on so called component templates and service application specifications [23]. To address the issue of application architecture conform system configuration, interface roles are introduced that allow the consideration of component behavior during the composition of an application. Moreover, these interface roles and application specifications are extended by a quality of service concept. [24]. We just shortly introduced the main features

from the DAiSI, which are necessary to understand our overall concept. Still there are more features described in detail in the references.

Community Driven Software Engineering as already shortly mentioned in Section 1-C - Community-Driven Design is already somehow established in Software Engineering. However, the communities here are more focused on developing products, such as Open-source projects like Linux, LibreOffice, etc. Furthermore, communities exist to improve already existing products, such as computer games, by providing mods and in-official add-ons.

Nevertheless, Community-Driven Software Engineering is mainly product-oriented instead of problem-oriented. Communities are built to solve a common issue (bugs in a computer game), improve an existing software library, and so on. These improvements are mainly based on user feedback (out of the community). This concept of problem-driven community development or improvement is also described in Donger et al. [17]. However, a problem-oriented community-driven software engineering is still missing!

3 Overall Concept

This section proposes a novel approach, so-called *Community Driven Design in Software Engineering*, for tackling the problems and challenges in the circular economy as introduced in the previous sections. The fundamentals for this approach are the infinite model from Figure 1 and the topics discussed in section 2.3. In particular, the concepts part of the Dynamic Adaptive System Infrastructure (DAiSI) plays an essential role. They are applied within the individual steps. We explain the five steps of the Community-Driven Design approach with the motivating example introduced in the introduction.

3.1 Domain-Driven Problem Descripton (Trigger)

Regarding Figure 1, this step refers to the phase *release* in which new challenges are arising. These challenges are the trigger for innovations, furthermore the problem had to be described within the domain context.

In our example, the mine as initiator has to describe its challenge. For this, the Application and Template concept of DAiSI are used. As visualized in Figure 2, the application (*EnergyConsumptionForVentilation*) is described, which contains all components necessary for the description.

For the application it is irrelevant to have the state *Runnable* because additional components would be required for this. These components exist within the mine's application landscape to describe the problem; only the relevant fragment is presented publicly. The components are not fully described; otherwise, the mine might expose company secrets. Nevertheless, the concept of a template is that if the component meets the template's requirements, then it can be easily integrated into the application.

To describe an improvement step, we extend the DAiSI meta-model (see [23]) to enable the assignment of properties, including default values, to templates.

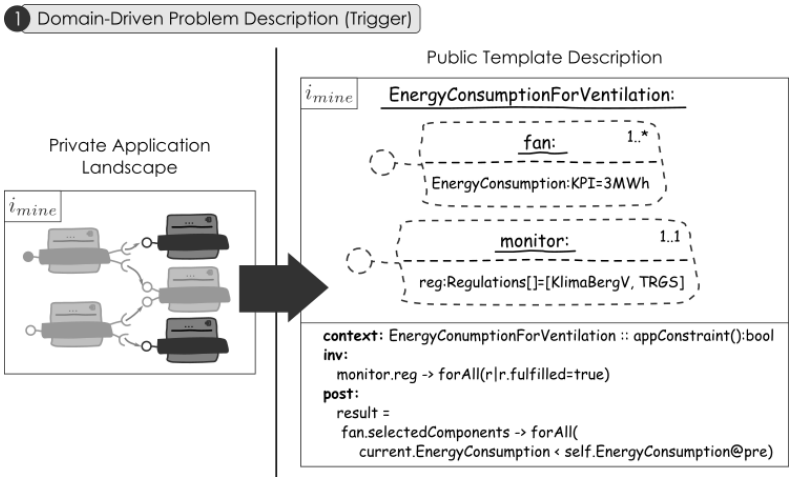

Fig. 2. Domain-Driven Problem Description (Trigger)

With this extension, it is not only possible to formulate application constraints (see also [23]), which make assumptions about the number or role of components, but also about traits of individual components. In the example described here, these are two attributes that describe the challenge of the mining company. As an invariant, firstly, all legal regulations that apply to ventilation and work safety must be met. Secondly, the energy consumption for all fans must be lower than in the current case. Thus, both the problem and an optimization criterion are defined as specified in Figure 2.

3.2 Community Matching (Loosely Coupled)

After defining the challenge, the platform must be able to generate a community out of the problem description, the individuals joined through this platform, and the CE model (see [25]). The community is a subset of individuals with the collective purpose of contributing to the problem (see Figure 3).

This community is described as loosely coupled, as there is only the possibility that the actors could cooperate but do not have to. On the other hand, in step five *Community-Building*, we will not talk about a loosely coupled community anymore since there is an assurance from the participants that they will collaborate.

In our example scenario, the community ($ECfV$) consists of the initiator (i_{mine}), and other community members C. In this case, other mines (c_{mine_1}, c_{mine_2}) with the same or similar problems, a specialist for simulation of underground ventilation (c_{sim}), a software company (c_{ai}), and two fan manufacturers (c_{eng_1}, c_{eng_2}).

It does not matter if the members already have a business relationship with each other or if they are even competitors. Therefore, this community is called

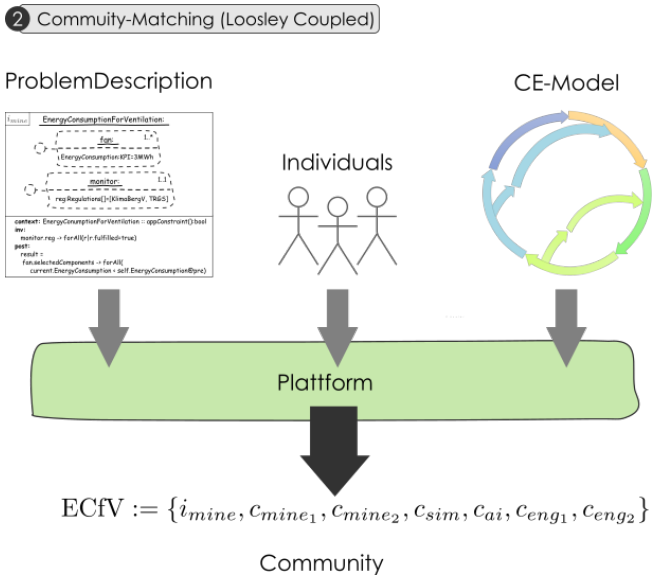

Fig. 3. Community-Matching (Loosely Coupled)

Loosely Coupled because this community's creation was not based on a contract, only on the contribution that a member can add to the solution.

3.3 Community Driven Solution Description (Concept Design)

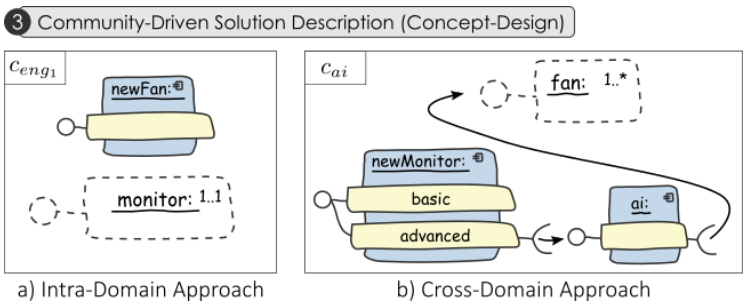

Fig. 4. Community Driven Solution Description (Concept Design)

Each community member can now generate concepts for solving the problem; not all members need to work together. Any number of solutions can be generated in which subsets of members participate. The solution is described by specifying concrete DAiSI components that fulfill the template. This can be

an obvious solution in which a fan manufacturer offers a fan that consumes less energy. As the initiator, we call this an intra-domain approach, as the solution provider comes from the same domain Another possible solution could be developed from a community made up of partners from different domains (cross-domain approach). In this example, it might be a software company (c_{ai}), which provides an ai-based ventilation-on-demand algorithm to the data center C introduced in Section 1.3.

A template component can be understood as the minimum configuration of a component. The configuration concept of the DAiSI allows providing different kinds of services depending on the available components. Figure 4 b) shows the component *newMonitor* offering a basic configuration and an advanced configuration (yellow bar in Figure 4). We can imagine that the *basic* configuration implements the current functionality, and the *advanced* configuration provides the monitoring data for use in the *ai* component. In the event of a failure of the component *ai*, it is possible to switch back to the state of practice, and the component also integrates seamlessly into the application landscape. More information about the configuration mechanism can be found in Klus et.al.[22]

3.4 Evaluation according to CE model (Shared Goal)

4 Evaluation according to CE model (Shared Goal)

```
context: EnergyConumptionForVentilation :: compareTo(a:Application):int
post:
  result = let m : CEModel = commonModel.getModel() in
      if m.getValue(self) < m.getValue(a) then
         1
      elseif m.getValue(self) = m.getValue(a)
        0
      else
         -1
      endif
```

Fig. 5. Evaluation according to CE model (Shared Goal)

From the set of possible solutions, the best solution has to be selected. The CE model provides the framework for this so that the platform is able to evaluate the best solution from the given information in the context of this model.

As Figure 5 shows, this is realized by implementing the *compareTo* method, which is part of an *application* concerning the DAiSI meta-model. The solutions can be compared pairwise to find the best in the sense of the CE. In our concrete case, this would be implementing the *VentilationOnDemand* (Figure 4 b)) solution since it is not in the sense of the CE to exchange a working fan. It would probably not even be economical in this example case.

3.5 Community Building (Micro-Project)

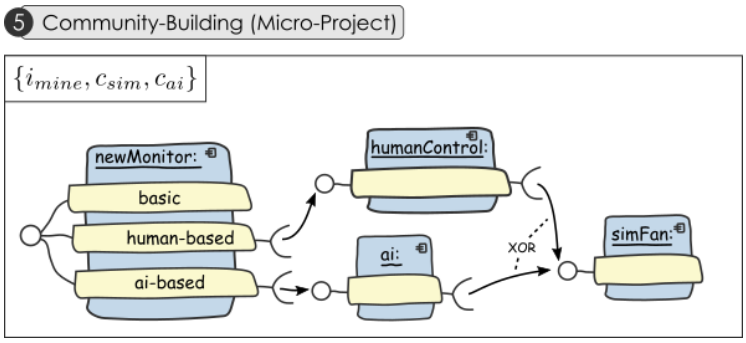

Fig. 6. Community Building (Micro-Project)

In our final step, the essential community-building takes place. Once the platform has supported the decision-making process and generated a ranking for the solutions, the community members can now commit to a *Micro-Project* in order to continue on a specific solution. A *Micro-Project* because the focus is on a short runtime and an experimental character of the project. Both the risk and the investment are shared among the members. This *Micro-Project* can be seen as a pre-stage to new products, processes and innovative solutions and as a catalyst to convince even the competitors to work on a standard solution.

Figure 6 shows an evolved solution, to minimize the risk. The community extends the initial solution by the specialist's know-how to simulate underground ventilation. This allows creating a test environment and evaluating the solution without risking human lives due to insufficient ventilation caused by a faulty or not well-trained AI. To obtain additional resilience, also a control component (*humanControl*) has been introduced, which can replace the *ai* component in case of exceeding the limits for hazardous gases. The mechanism of the DAiSI that can be used for this purpose is the extended *InteraceRole* described by Herrling et al. in Extending Interface Roles to Account for Quality of Service Aspects in the DAiSI.

4 Discussion, Challenges and Further Research Directions

In the last Section, we presented a novel approach towards *Community Driven Design in Software Engineering*. Based on our introduced approach, this section discusses the main obstacles of the approach and derive future social and technical challenges, and additional research directions.

4.1 Discussion and Challenges

Our approach is a combination of a social approach, obtained by setting up communities and a technical one based on the concepts of the DAiSI Middleware. It is expected that the concepts of the DAiSI won't solve all challenges. We will discuss two essential aspects, steps two and four of the described approach, for which DAiSI can only provide the technical framework.

In addition to the technical description of the problem, it is also a challenge to understand the semantic context, and there must be a direct mapping of community members to the phases of the CE model. The geographical location of the actors also plays an important role here, even if we are describing a distributed system. Since the transport, production, and processing (repair, reuse, rebuilt) impact the CE's performance, the actors inherently form regional communities that are not static and fixed but very individual and dynamic. The development of new transport opportunities, infrastructure development, and the sustainability of processing and production directly influence this.

This is why the approach presented here can be implemented on a large scale nowadays. Because in step four, two solutions have to be compared with each other, and for this analysis, pieces of information are required, which are not available today for every existing thing and every process. To seriously realize this approach, it must be possible to determine both the state of health and the sustainability factor of the production and processing of the existing components and the impact factor that the new components have on the system - in other words, it must be possible to assess how something is produced, how repair-, reuse- and rebuilt-friendly something is, and how well the materials it contains can be recycled at the end of its life cycle.

4.2 Further Research Directions

The range of subsequent research topics arising from this approach is extensive, and the main ones are briefly listed below:

Privacy vs. Transparency: The tension between data worthy of being protected and transparency already becomes evident in the motivating example. With the amount of data collected in all areas of life today, it is almost impossible to estimate the potential information hidden in that data. For this reason, it is necessary to implement mechanisms for the exchange, quality determination and assurance, and the processing of data in secured and protected environments - to protect the know-how of both the data provider and the data consumer.

Shared vs. Concentrated Ownership: Today, it is the case where the one who takes the risk will have the positive or negative outcome. To get everything in case of success (winner-takes-it-all mindset), it is obvious to secure this success with, e.g., patents. It is not unusual for a competitor to profit from innovation because the risk of improving an existing product is much lower than the risk of developing innovation on their own. The results are patents and legal actions, which are time-consuming, expensive, and in the worst case, results in only losers. Ultimately new solutions are required so that it is not only a singular

goal (profit maximization) worth following during development, especially if the market volume is limited.

Quality Assurance vs. Emergence: There are various methods to increase quality, but they are not suitable for emergent behavior, desired in dynamically adaptive systems. This leads to the need to think about other mechanisms for quality management. Nevertheless, not only that but also in the area of standardization, new ways have to be found because standardization processes are not compatible with short spontaneous projects.

Economic vs. Ecology: An obvious tension exists between a company's mostly economic goals (first goal) and the secondary goals, as mentioned in the introduction. As long as it is an economic risk to focus on sustainability, there will be only a few sustainable business models. Often these models are driven by non-profit organizations, but the effects of scale that a global acting company can achieve only in the rarest cases achieved by a non-profit association.

The tension listed above rises to other technical challenges for implementation, such as:

Managed Process Transformation and Evolution: The buzzword that is stressed here is digitization! It is essential to provide a technological and technical infrastructure and enable processes to be digitized as they are to transform existing processes to a better version of that use case. Only through a regular review and managed transformation of each process the potentials of digitization can be achieved.

Semantic Composition and Orchestration: The approach presented here abstracts very firmly from the technological stack in some parts. It is clear that the connection and adaptation mechanism of the DAiSI explored here does not scale to large ecosystems. The main challenge here concerns the description of the services and their composition and orchestration. Here, a good trade-off between being understandable and usable by the end-user and easily processable by an algorithm must be achieved and therefore part of current research.

Identity and Traceability: One aspect that has become relevant not only since to the hype about blockchain technologies is the allocation and management of identities and the possibility of tracking every change in such an ecosystem without any gaps, which is necessary to calculate the state of health, for example. Through the introduced scenario, it has become quite clear that there is no one all-inclusive solution. An optimal solution for the CE model is a very individual solution and depends on various factors. Furthermore, the validity of a solution is also limited in time. If it is possible to extract certain materials by a newly developed recycling process, this can lead to another way, which is more optimal in terms of the Circular Economy.

5 Conclusion

In the scope of this paper, we presented a community-driven approach for tackling sustainability problems and supporting the achievement towards the circular

economy. Circular Economy objectives are mainly secondary objectives of companies since their main business is settled in another domain. Accordingly, it is understandable that these companies prefer to invest and conduct research in their primary objectives. Nevertheless, climate protection, sustainability, and achieving the circular economy concern all of us! For this matter, we have presented our approach for a *Community Driven Design in Software Engineering*: A combination between a social approach, by building communities, that have similar problems and challenges, as well as shared goals, and a technical approach, based on the DAiSI. The DAiSI is a Dependable Dynamic Adaptive System Infrastructure. These concepts enable and support our approach. Our approach is based on five steps, beginning with a *Domain-Driven Problem Description*. The Domain-Driven Problem descriptions support community building by bridging gaps between the problem understanding. It enables matchmaking between parties that may never hear from each other but have similar problems and challenges in their domain. In the following steps, the community will be matched (loosely coupled) and develop a community-driven solution description. This leads to a proposal that will be evaluated against the Circular Economic model (a shared goal). Finally, the solution will be developed and established in microprojects. All in all, we evaluated our approach by using the example of the Circular Economy as a shared goal. However, this is still just one example, and cross-domain. Our approach is also transferable to other goals. The same risky situation can also be described for the second use of data, where the benefit for the individual is estimated to be lower than the risk of giving the data away.

Out of the scope of our discussion in this paper were more social and business-related questions, like fairness in the distribution of profits and risks. These need to be more investigated in further research, as well as the open technical challenges. Furthermore, we need solutions for the mentioned problems related to data and information, such as privacy, traceability, and many (research)topics. Here, digital twins could provide a solution, but not as used at the moment in the production. A clear separation between the digital twins and the digital shadows is necessary, as well as methods for a secure data exchange. For the latter, we are planning to adopt a sandbox approach, as we already presented for a data marketplace [26].

References

1. M. S. Andersen, "An introductory note on the environmental economics of the circular economy," *Sustainability science*, vol. 2, no. 1, pp. 133–140, 2007.
2. S. Lawrenz, M. Nippraschk, P. Wallat, A. Rausch, D. Goldmann, and A. Lohrengel, "Is it all about information? the role of the information gap between stakeholders in the context of the circular economy," *Procedia CIRP*, vol. 98, pp. 364–369, 2021.
3. S. Blömeke, M. Mennenga, C. Herrmann, L. Kintscher, G. Bikker, S. Lawrenz, P. Sharma, A. Rausch, M. Nippraschk, D. Goldmann *et al.*, "Recycling 4.0: An integrated approach towards an advanced circular economy," in *Proceedings of the 7th International Conference on ICT for Sustainability*, 2020, pp. 66–76.
4. G. Barzilai, *Communities and law: Politics and cultures of legal identities*. University of Michigan Press, 2010.

5. F. Tönnies, *Tönnies: Community and civil society.* Cambridge University Press, 2001.
6. M. Wahlström, M. Sommer, P. Kocyba, M. de Vydt, J. De Moor, S. Davies, R. Wouters, M. Wennerhag, J. van Stekelenburg, K. Uba *et al.*, "Protest for a future: Composition, mobilization and motives of the participants in fridays for future climate protests on 15 march, 2019 in 13 european cities," 2019.
7. J. Rowley, "The wisdom hierarchy: representations of the dikw hierarchy," *Journal of information science*, vol. 33, no. 2, pp. 163–180, 2007.
8. D. Fudenberg and J. Tirole, *Game theory.* Cambridge, Mass.: MIT Press, 1991.
9. J. P. Carse, *Finite and infinite games.* New York: Penguin Books, 1986.
10. S. Newman, *Building microservices: designing fine-grained systems.* " O'Reilly Media, Inc.", 2015.
11. B. Wilson, *Resilience for All: Striving for Equity Through Community-Driven Design.* Washington, DC: Island Press/Center for Resource Economics, 2018.
12. P. Monga, Radhika, and D. Sharma, "Structural and Functional Unit of Environment: Ecosystem," in *International Conference on Recent Innovations in Engineering, Science, Humanities and Management (ICRIESHM*, 2017, pp. 275–280.
13. C. Deiters, M. Köster, S. Lange, S. Lützel, B. Mokbel, C. Mumme, and D. Niebuhr, "Demsy–a scenario for an integrated demonstrator in a smartcity," *NTH Computer Science Report*, vol. 1, 2010.
14. A. Rausch, J. P. Müller, D. Niebuhr, S. Herold, and U. Goltz, "It ecosystems: A new paradigm for engineering complex adaptive software systems," in *2012 6th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*. IEEE, 2012, pp. 1–6.
15. M. Schindler, S. Schoone, and E. Clausen, "Towards an evolving software ecosystem in the mining industry," in *ADAPTIVE 2020, The Twelfth International Conference on Adaptive and Self-Adaptive Systems and Applications*, 2020, pp. 76–84.
16. M. Schindler, A. Rausch, S. Schoone, and E. Clausen, "Projekt mamma – digitalisierung am beispiel der vorausschauenden wartung," in *9. Kolloquium Fördertechnik im Bergbau*, IBB, Ed., 2020, pp. 97–114. [Online]. Available: https://www.bergbau.tu-clausthal.de/fileadmin/Foet_2020/Tagungsband_Foet_2020.pdf
17. P. Dongier, J. van Domelen, E. Ostrom, A. Ryan, W. Wakeman, A. Bebbington, S. Alkire, T. Esmail, and M. Polski, "Community driven development," *World Bank Poverty Reduction Strategy Paper*, no. 1, 2003.
18. E. Evans and E. J. Evans, *Domain-driven design: tackling complexity in the heart of software.* Addison-Wesley Professional, 2004.
19. R. Jabbari, N. bin Ali, K. Petersen, and B. Tanveer, "What is devops?" in *Proceedings of the Scientific Workshop Proceedings of XP2016.* New York, NY: ACM, 2016, pp. 1–11.
20. R. De Lemos, H. Giese, H. A. Müller, M. Shaw, J. Andersson, M. Litoiu, B. Schmerl, G. Tamura, N. M. Villegas, T. Vogel *et al.*, "Software engineering for self-adaptive systems: A second research roadmap," in *Software Engineering for Self-Adaptive Systems II.* Springer, 2013, pp. 1–32.
21. H. Klus, D. Niebuhr, and A. Rausch, "A component model for dynamic adaptive systems," in *International workshop on Engineering of software services for pervasive environments in conjunction with the 6th ESECFSE joint meeting*, A. L. Wolf, Ed. New York, NY: ACM, 2007, pp. 21–28.
22. H. Klus and A. Rausch, "Daisi - a component model and decentralized configuration mechanism for dynamic adaptive systems," in *ADAPTIVE 2014, The Sixth International Conference on Adaptive and Self-Adaptive Systems and Applications*, 2014, pp. 27–36.

23. H. Klus, A. Rausch, and D. Herrling, "Component templates and service applications specifications to control dynamic adaptive system configurations," in *AMBIENT 2015*, M. Weyn, Ed. Wilmington, DE, USA: IARIA, 2015, pp. 42–51.
24. D. Herrling, A. Rausch, and K. Rehfeldt, "Extending interface roles to account for quality of service aspects in the daisi," *International Journal On Advances in Software*, vol. 9, no. 1 and 2, pp. 37–49, 2016.
25. S. Lawrenz, L. Lux, M. Nippraschk, P. Sharma, A. Rausch, and D. Goldmann, "A transition framework towards a circular economy illustrated by the example of electrical and electronic equipment."
26. S. Lawrenz and A. Rausch, "Dont buy a pig in a poke a framework for checking consumer requirements in a data marketplace," in *Proceedings of the 54th Hawaii International Conference on System Sciences*, 2021, p. 4663.