



A Lean Approach of Managing Technical Debt in Agile Software Projects – A Proposal and Empirical Evaluation

Abdullah Aldaej¹✉, Anh Nguyen-Duc², and Varun Gupta³

¹ Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia
aaaldaeej@iau.edu.sa

² University of South Eastern Norway, Notodden, Norway
angu@usn.no

³ GISMA University of Applied Sciences, Potsdam, Germany
varun.gupta@gisma.com

Abstract. Technical Debt Management (TDM) includes activities such as identifying, measuring, and prioritizing technical debt. It is mainly performed to proactively mitigate the risk of losing the maintainability and evolvability of the software product which results in reducing the team velocity. Despite the importance of TDM, its adoption in software companies remain limited. Software companies are witnessing high market demand and competition that make delivering customer value outweighs the effort invested in TDM activities. Since the impacts of technical debt are uncertain and evident only in the long run, it is more difficult for companies with very limited resources to proactively spend their resources on TDM. In this paper, we propose a lean approach to facilitate the adoption of TDM in software companies with very limited resources. Based on this approach, TDM is driven by project management metrics, such as team or sprint velocity, and velocity variance. We conducted an initial evaluation of the concept of this approach through a short survey of 43 software project/product managers. Most of the survey respondents have a positive impression about our approach, which will encourage us to proceed further using more robust empirical evaluation.

Keywords: Technical debt · Project management · Agile software development · Sprint velocity

1 Introduction

Managing technical debt is reportedly non-trivial tasks for project managers and team leaders in software development projects. Technical Debt (TD) is a metaphor for a work-around technical solution that is not sustainable and requires future rework [1]. It has been shown as an important phenomenon to expedite the development in the short term in both large companies [2] and small and startup companies [3]. Technical Debt Management (TDM) is a process that includes different activities such as identifying, measuring, monitoring, prioritizing, and repaying TD. It has been adopted by some

large software companies, but at different levels of maturity [2, 4]. Although TDM is important to proactively manage the risk of TD, it introduces extra activities to the regular development which leads to extra costs [2, 5]. Consequently, many companies (especially the ones with limited resources) would find TDM difficult to adopt as part of their development process. Furthermore, it is difficult to identify and manage large instances of TD items in agile software development [6].

Managing technical debt is not independent from other aspects of project management and the project development life cycle. When it is managed, TD is often treated as a task in the product backlog along with other tasks such as new features and bug fixes. Measuring and prioritizing TD in the product backlog depends on the evaluation of its negative impact on the product. Some negative impacts of TD has been found from literature, for instance, delaying the delivery of releases [7], increasing the number of maintenance activities [8], and reducing the team productivity [9]. However, knowing the negative impact of TD is only one half of the story. Agile project managers need a precise way of measuring this impact, to better communicate it to other stakeholders.

In agile projects, there are possibilities with using agile productivity metrics, such as velocity, cycle time and takt time [10, 11] for measuring TD implications. Sprint velocity is among the most common productivity metrics in agile projects that use Scrum method [10], by showing the number of tasks (i.e., story points) that the team can accomplish in one sprint. Cycle time is defined as the amount of time that passed from when work actually started to fulfilling the work [12]. Takt time is defined as the average time interval between the start of development of two sequential software versions to meet customer demand [13, 14]. This measure helps to identify if the software team can meet the customer demand, exceeding the demand (over producing) or unable to meet demand (under producing). To the best of our knowledge, there is no study that connects the agile team productivity metric with the TDM process.

In this paper, we propose a lean approach for TDM by incorporating the usage of agile productivity metrics within the TDM process. This lean approach is used specifically for TDM within the agile software development methodology. In general, lean is an approach that focusses on delivering maximum value with minimum waste. Our approach is “lean” in a way that emphasizes on minimizing the effort and complexity of TDM, by focusing on monitoring and addressing TD implications quantitatively. This can facilitate the adoption of TDM by 1) providing a lightweight approach for TDM in agile software projects, and 2) bridging the communication gap between technical and non-technical people for decisions related to allocating resources for TDM. We conducted a short survey of 43 software product and project managers to initially evaluate the concept of our approach. The results reveal that most project managers foresee potential advantages of our approach.

2 The Lean Technical Debt Management Process

2.1 The Management Approach

The proposed approach introduces a new paradigm of dealing with TD. Based on this approach, TDM is driven by a popular metric in agile project management such as sprint velocity. In addition to its usage for measuring the development velocity, we use this

metric as a quantitative measure for TD implication. Since dealing with TD involves high uncertainty about its future impacts (i.e., difficulty to measure TD interest), TDM is proceeded based on a control threshold of the development velocity.

Our approach applies a lean mechanism that simplifies the way of dealing with TD. It assumes that issues that directly affect the development velocity are considered TD issues. This assumption is based on evidence from previous studies that TD can be manifested in multiple aspects (beyond software code and architecture) such as process, people, social [11, 15]. Also, TD related to process and people are found to be more interested to project management practitioners [16]. In addition, the development speed (i.e., delivery speed) is identified as the most common effect of TD [7].

Figure 1 shows our lean approach for TDM. It connects some project management activities (PM layer) with TDM activities (TDM layer). At the PM layer, the agile project managers monitor the team velocity at the end of each sprint using some metrics (e.g., sprint velocity) which is usually available in the PM data. In addition, the project managers need to establish control thresholds for the velocity that determine the optimal velocity range. At the end of each sprint, the project managers check the variance between the actual and optimal sprint velocity. If the actual velocity is below the optimal, then the project managers can measure the impact of such low velocity by calculating the difference (i.e., TD interest). The following formulas are used to measure velocity and TD interest:

$$Actual (V) = Actual\ velocity\ of\ the\ current\ sprint \quad (1)$$

$$Optimal (V) = MD(optimal\ velocity\ range) \quad (2)$$

$$Current (TDI) = Optimal (V) - Actual (V) \quad (3)$$

$$Future (TDI) = Current (TDI) * \#Planned\ release \quad (4)$$

where V indicates Velocity, MD refers to Median, TDI stands for TD Interest, and $\#$ is the Number of future released to be considered in the prediction.

Since our approach is driven by TD implication (which is measured using the development velocity), It begins by measuring TD interest before identifying TD items. The TD interest is precisely measured for the present as the difference between current and optimal velocity (step 2.1 in Fig. 1). After that, the TD interest can be estimated in the future considering the most recent TD interest and the number of future releases/sprints that captures the future timeframe (step 2.2 in Fig. 1). This information can help project managers to communicate with non-technical stakeholders (e.g., high management or investors). It also supports decision making related to allocating resources for TDM. If the non-technical stakeholders decided to spend resources for TDM, then the project managers can begin the process of TDM (i.e., move to the TDM layer). If not, then the project managers will remain on the PM layer and continue 1) monitoring the development velocity, and 2) informing the non-technical stakeholders about the velocity variance and its related TD impacts.

TDM layer includes the main TDM activities [4], which are proceeded after the agreement to allocate resources for TDM. This layer includes some TDM activities

such as TD identification (step 3 in Fig. 1), TD measurement (step 4 in Fig. 1), TD prioritization (step 5 in Fig. 1), and TD repayment (step 6 in Fig. 1). The TDM layer is adopted from the TDM frameworks in [4, 17]. However, we eliminate activities such as TD monitoring and TD prevention since our approach concentrates on monitoring TD implications (not TD items). Also, our approach applies a different mindset for TDM that is driven by a control threshold of TD implications (with less emphasis on TD prevention). In the TDM layer, the project managers begin the TDM process by identifying TD items which are (according to our approach) the causes that are directly associated with the velocity reduction. Then, TD items are measured as the cost of fixing each item (i.e., TD principal). After that, TD items can be prioritized based on their fixing cost and their impact on the development velocity. Finally, TD repayment is performed starting from TD items with highest priority.

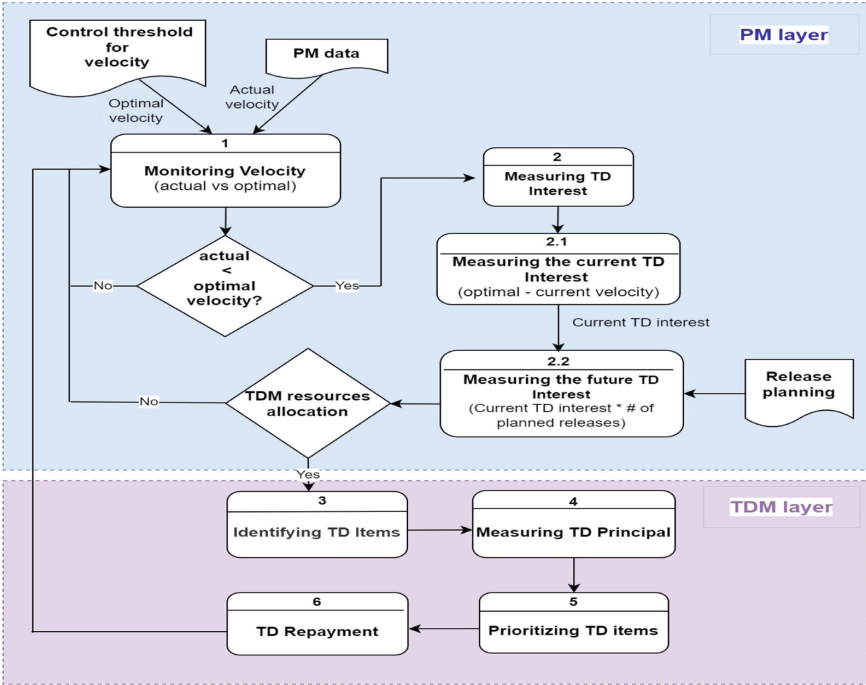


Fig. 1. Lean TDM process

2.2 Use Scenarios




Table 1 shows an example of how the lean TDM approach can be applied. In this example, we classified the development velocity into three risk levels: low risk, moderate risk, and high risk. The low-risk indicates the optimal speed range based on the company’s goal. At this level (low risk), the team can confidently skip TDM. It means that it is not efficient to allocate resources for TDM since no symptoms appears. When the speed is

at the moderate-risk range, it is an indication for early TD symptoms. At the moderate risk level, the team needs to keep non-technical stakeholders informed about this early TD symptoms and their impact by calculating the TD interest as the difference between current and optimal/planned speed. This high-level of measuring TD interest considers the observed (fact) consequences of TD that is the gap in the development speed. In case the optimal speed is determined based on a range of value, the median value of the optimal range can be used to compare with the actual value.

In addition, the lean TDM approach can be used to provide a guidance (recommendation) for decisions to allocate resources for TDM. For example, at optimal velocity range, the recommendation would be spending resources on TDM is not mandated. When reaching the moderate range, it is recommended to allocate resources for TDM, but TD repayment is optional or can be partially implemented. At the low velocity, TD repayment is needed. Based on this example, the team should not wait until reaching the low velocity. But rather proactively informing the high management about TD consequences when reaching the moderate range, using precise fact-based measures of TD interest.

The example in Table 1 is used only to illustrate one way of applying our approach. But it is flexible for any context to select its own metric and control threshold for the development velocity. At the end, the overall concept is the same that to have a common control threshold for the development velocity that is applied within the team, and such a threshold drives the TDM process and TD interest measurements.

Table 1. Lean TDM Approach at PM Layer – An Example.

The dev speed	Risk level	Current TD interest	Future TD interest	Guidance for effective TDM resource allocation
 Optimal speed	Low risk	--	--	No resources should be spent on managing or repaying TD
 Moderately deteriorated speed	Moderate risk	Equation (3)	Equation (4)	The team should allocate some resources for managing TD
 Highly deteriorated speed	High risk	Equation (3)	Equation (4)	The team should allocate some resources for managing and repaying TD

3 Empirical Validation

To initially validate our proposed lean TDM approach, we surveyed 43 software project/product managers from different software companies around the world. We targeted software project/product managers because our approach focusses on the project management activities. For example, seeking resources for the product from other stakeholders is the main role for the project/product managers. Sampling of this population was performed using convenience sampling method [18]. In addition, we recruited some participants (22 out of 43) from a professional recruitment channel, which is Prolific¹.

The survey instrument is available in our supplemental material². It is a short survey that primarily intended to communicate and get feedback about our lean TDM approach. It consists of three sections. The first section presents a brief explanation about our idea to participants, and shows a table that illustrates our idea. The second section includes some questions that characterize the participants, such as company location, company size, and years of experience in the project/product management role. Finally, the third section asked respondents to initially evaluate our idea based on three technology acceptance metrics from TAM (Technology Acceptance Model) [13], which are perceived ease of use, usefulness, and predicted future use. Before disseminating the survey, we validate the survey internally that each author read the survey’s questions and provide his feedback. At the end, we conducted a consensus session where we discuss the comments and reach consensus about the survey.

A total of 43 participants filled the survey. Figure 2 shows the characteristic of our participants. Most of the participants works in companies in Europe with less than 5 years of project management experience. In terms of company size, our participants work in different sizes of company, but most of them in small companies.

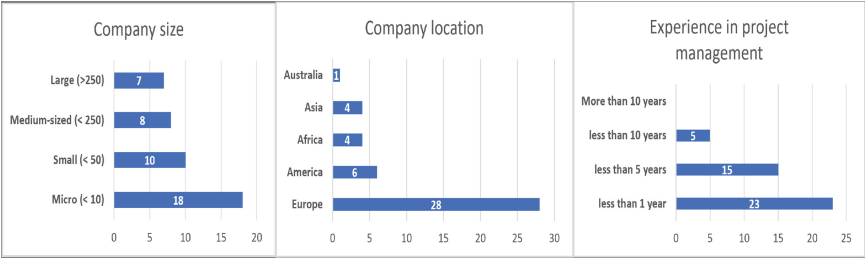


Fig. 2. Characteristics of participant

Figure 3 summarizes the participants’ opinion on our proposed approach. It presents their answers to the closed questions in the third section of the survey (TAM related questions). On average 61.6% of the respondents perceived that the proposed approach would be easy to use (strongly agree or agree). In assessing the usefulness, on average 79.05% of the respondents strongly agree or agree that the proposed approach would

¹ <https://prolific.co/>.
² <https://bit.ly/3JCWQzy>.

be useful for project managers. Finally, on average 54.7% of the respondents strongly agree or agree that they will use the proposed approach in future to manage TD.

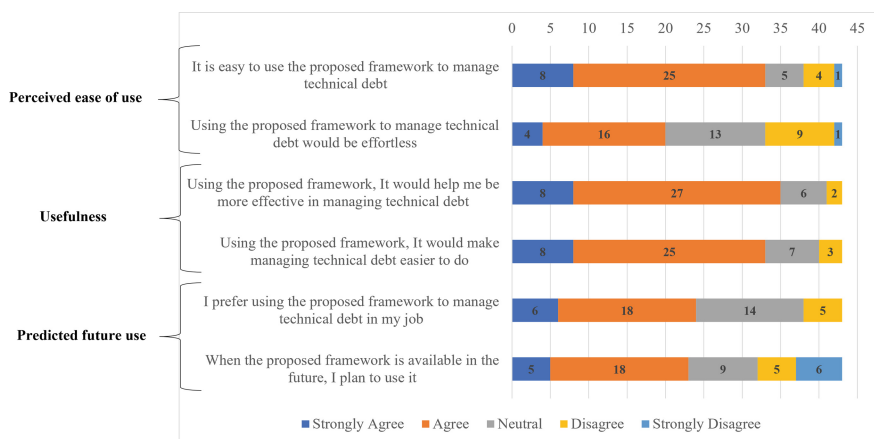


Fig. 3. Results of the initial evaluation to lean TDM approach

Some participants reported in the open question that the proposed approach is more helpful for an organization with very limited resources. For example, small organizations or startups usually do not have dedicated quality assurance roles. This kind of organization often relies on external support (e.g., consultancy agency) when such quality assurance is required. However, other participants were a bit skepticism as the actual usefulness and ease of use could only be predicted after using the proposed process. The dataset is available in our supplemental material³. In sum, this initial results provides early insights into the potential usefulness of our approach by software project managers.

4 Discussions

Previous work proposed different TDM frameworks and strategies to encourage the use of TDM in practice [19, 20]. Other studies introduced different techniques to support software teams to perform specific TDM activities, such as TD identification [21], TD measurement [22], TD prioritization [23]. However, most of the previous studies focus on TDM from certain dimensions (e.g., code, design, and architecture, etc.). Since TD is a multidimensional phenomenon, it can be difficult to have a framework that manages all aspects of TD. What distinguish our paper from previous work is that it addresses the multidimensionality of TD by increasing its level of abstraction. That we consider TD as issues that are associated with the development speed. We think that this approach can simplify the adoption of TDM in practice. It can also amplify the communication bridge between TDM and agile PM activities.

³ <https://bit.ly/3Y0PJ8J>.

4.1 Threats to Validity

There are four main limitations in our approach. First, we assume that all issues contributed to the reduction of development velocity are considered TD. Although this assumption might simplify the adoption of TDM, it can introduce a risk of including some issues not related to TD. To address this risk, we plan (in our future work) to add a risk factor in our equation to account for this risk. Second, our approach depends on existing metrics that are used to measure team velocity. We assume that project managers have an existing metric in place related to team velocity. However, a metric such as team/sprint velocity is found to be a common one in agile software project management [10] which can minimize this threat in the agile software development context. Third, the process of identifying TD items (step 3 in Fig. 1) depends on people experience and how thoughtful they investigate the issues that cause the development speed reduction. So, the quality of TD identification depends on people experience. Lastly, we recruited some participants using Prolific, a professional channel for research participant recruitment. Despite the limitation for its lack of explicit design, it has emerged as a viable and cost-efficient option for researchers [24].

5 Conclusions

The application of TDM in practice remains limited especially in organizations with limited resources (i.e., those organizations that do not have a mature quality assurance team). In this paper, we proposed a lean approach to facilitate the adoption of TDM in such organizations. Our approach introduces a new mindset for managing TD that is driven by TD implications/symptoms (primarily the velocity of development). It introduces an alternative way of measuring TD interest, which is based on fact-based consequences of TD (i.e., the gap between the actual and planned development speed). We sent a short survey to software project/product managers to get their initial feedback about our approach. The majority of them have positive expectations for our approach. For future work, we plan to perform a longitudinal case study, which includes interviews at different points of time, to empirically validate and refine our approach.

References

1. Kruchten, P., Nord, R.L., Ozkaya, I.: Technical debt: from metaphor to theory and practice. *IEEE Softw.* **29**(6), 18–21 (2012). <https://doi.org/10.1109/MS.2012.167>
2. Martini, A., Besker, T., Bosch, J.: Technical debt tracking: current state of practice: a survey and multiple case study in 15 large organizations. *Sci. Comput. Program.* **163**, 42–61 (2018). <https://doi.org/10.1016/j.scico.2018.03.007>
3. Cico, O., Souza, R., Jaccheri, L., Nguyen Duc, A., Machado, I.: Startups transitioning from early to growth phase - a pilot study of technical debt perception. In: Klotins, E., Wnuk, K. (eds.) *ICSOB 2020. LNBIP*, vol. 407, pp. 102–117. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-67292-8_8
4. Yli-Huumo, J., Maglyas, A., Smolander, K.: How do software development teams manage technical debt? – an empirical study. *J. Syst. Softw.* **120**, 195–218 (2016). <https://doi.org/10.1016/j.jss.2016.05.018>

5. Guo, Y., Seaman, C., da Silva, F.Q.B.: Costs and obstacles encountered in technical debt management – a case study. *J. Syst. Softw.* **120**, 156–169 (2016). <https://doi.org/10.1016/j.jss.2016.07.008>
6. Holvitie, J., et al.: Technical debt and agile software development practices and processes: an industry practitioner survey. *Inf. Softw. Technol.* **96**, 141–160 (2018). <https://doi.org/10.1016/j.infsof.2017.11.015>
7. Ramač, R., et al.: Prevalence, common causes and effects of technical debt: Results from a family of surveys with the IT industry. *J. Syst. Softw.* **184**, 111114 (2022). <https://doi.org/10.1016/j.jss.2021.111114>
8. Kruchten, P., Nord, R., Ozkaya, I.: Managing Technical debt - Reducing friction in software development. in *SEI Software Engineering*. Pearson Education (2019)
9. Besker, T., Martini, A., Bosch, J.: Software developer productivity loss due to technical debt - a replication and extension study examining developers' development work. *J. Syst. Softw.* (2019). <https://doi.org/10.1016/j.jss.2019.06.004>
10. Kupiainen, E., Mäntylä, M.V., Itkonen, J.: Using metrics in agile and lean software development – a systematic literature review of industrial studies. *Inf. Softw. Technol.* **62**, 143–163 (2015). <https://doi.org/10.1016/j.infsof.2015.02.005>
11. Malakuti, S., Heuschkel, J.: The need for holistic technical debt management across the value stream: lessons learnt and open challenges. In: 2021 IEEE/ACM International Conference on Technical Debt (TechDebt), pp. 109–113 (2021). <https://doi.org/10.1109/TechDebt52882.2021.00021>
12. Budacu, E.N., Pocatilu, P.: Real time agile metrics for measuring team performance. *Informatica Economica* **22**(4) (2018)
13. Taghizadegan, S.: Design for lean/kaizen six sigma. In: Taghizadegan, S. (ed.) *Essentials of Lean Six Sigma*, pp. 59–101, Butterworth-Heinemann, Burlington (2006). <https://doi.org/10.1016/B978-012370502-0/50008-4>
14. Thollander, P., Karlsson, M., Rohdin, P., Wollin, J., Rosenqvist, J.: 14 - energy management using lean. In: Thollander, P., Karlsson, M., Rohdin, P., Wollin, J., Rosenqvist, J. (eds.) *Introduction to Industrial Energy Efficiency*, pp. 259–287. Academic Press (2020). <https://doi.org/10.1016/B978-0-12-817247-6.00014-6>
15. Martini, A., Stray, V., Moe, N.B.: Technical-, social- and process debt in large-scale agile: an exploratory case-study. In: Hoda, R. (ed.) *XP 2019. LNBP*, vol. 364, pp. 112–119. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30126-2_14
16. Gomes, F., dos Santos, E.P., Freire, S., Mendonça, M., Mendes, T.S., Spínola, R.: Investigating the point of view of project management practitioners on technical debt - a preliminary study on stack exchange. In: 2022 IEEE/ACM International Conference on Technical Debt (TechDebt), pp. 31–40 (2022). <https://doi.org/10.1145/3524843.3528095>
17. Seaman, C., Guo, Y.: Measuring and monitoring technical debt. In: *Advances in Computers*, p. 22 (2011)
18. Baltes, S., Ralph, P.: Sampling in software engineering research: a critical review and guidelines. *Empir. Softw. Eng.* **27**(4), 94 (2022). <https://doi.org/10.1007/s10664-021-10072-8>
19. Nikolaidis, N., Zisis, D., Ampatzoglou, A., Chatzigeorgiou, A., Soudris, D.: Experience with managing technical debt in scientific software development using the EXA2PRO framework. *IEEE Access* **9**, 72524–72534 (2021). <https://doi.org/10.1109/ACCESS.2021.3079271>
20. Wiese, M., Rachow, P., Riebisch, M., Schwarze, J.: Preventing technical debt with the TAP framework for technical debt aware management. *Inf. Softw. Technol.* **148**, 106926 (2022). <https://doi.org/10.1016/j.infsof.2022.106926>
21. Tu, H., Menzies, T.: DebtFree: minimizing labeling cost in self-admitted technical debt identification using semi-supervised learning. *Empir. Softw. Eng.* **27**(4), 80 (2022). <https://doi.org/10.1007/s10664-022-10121-w>

22. Avgeriou, P., et al.: An overview and comparison of technical debt measurement tools. *IEEE Software* (2020). <https://doi.org/10.1109/MS.2020.3024958>
23. Lenarduzzi, V., Besker, T., Taibi, D., Martini, A., Fontana, F.A.: A systematic literature review on technical debt prioritization: strategies, processes, factors, and tools. *J. Syst. Softw.* **171**, 110827 (2021). <https://doi.org/10.1016/j.jss.2020.110827>
24. Palan, S., Schitter, C.: Prolific.ac—a subject pool for online experiments. *J. Behav. Exp. Financ.* **17**, 22–27 (2018). <https://doi.org/10.1016/j.jbef.2017.12.004>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

