# A Benefit/Cost Analysis Tool for Better Benefits Management

Simon Lyngar[2], Martin Storvoll[2], and Jo E. Hannay[1][0000−0001−6217−758X]

[1] Simula Metropolitan Center for Digital Engineering, Dept. of IT Management, OsloMet,
Pb. 4 St. Olavs plass, 0130 Oslo, Norway
johannay@simula.no
[2] University of Oslo, Dept. of Informatics, Pb. 114 Blindern, 0314 Oslo, Norway
simonly@ifi.uio.no marstorv@ifi.uio.no

**Abstract.** The purpose of *benefits management* is to improve the probability of delivering value from development initiatives. Effective project management with a focus on benefits is important in software development; especially for optimizing resources and aligning project outcomes with business objectives. To succeed in this, one needs methods to analyze projects with a focus on realizing benefits in addition to the usual focus on time and costs. Building on an existing tool prototype to support such analyses, we developed an application implementing support for *benefit/cost analysis management* (BCAM) and investigated its potential for effectiveness and adoption. We asked how the accessibility of the tool affects the adoption of the supported practices, how tool support can simplify benefit-cost management activities, and how visual representations within these tools impact decision making in projects. The results show that the application was received favourably by the participants, with many expressing a willingness to consider its use in future Jira projects. Visual representations, including charts, tables, and timelines, were found to be particularly useful.

**Keywords:** Benefits management · Benefit/Cost Ratio · Tool Support · Jira · Project Management

## 1 Motivation

*Benefits management* is a loose set of practices designed to ensure that the outcomes of development initiatives enable the generation of benefits to stakeholders. Empirical evidence suggests that benefits management can be instrumental to the success of IT system development projects [1]. Despite this, and despite the extensive investment in software-based development, the realization of the anticipated benefits for stakeholders are not always guaranteed [2]. Moreover, project failure is often attributed to failures of time, cost, and scope, rather than on considerations of benefit [3,1].

The causes of software project failure seem to have remained relatively consistent since the early days of software development [1], implying the existence of some sort of persistent gap in the application of project management principles and desired outcomes; and in particular, a short-coming in the application of benefits management practices.

This motivates the call for a renewed focus on benefits management [4,5,6] as a mechanism to enhance project success rates.

Furthermore, the relatively recent development of a particular benefits management technique, *benefit/cost-driven agile development* [7] and its, as of yet, limited incorporation into the agile project toolbox highlight a significant opportunity for improvement. It has been argued that the integration of benefit-cost estimation and monitoring, and similar techniques, within agile projects can improve the efficiency and effectiveness of software development processes and benefits realization [7,8,9,10].

Projects that do not keep tabs on how much benefit they are generating can end up spending money, effort, and time on functionality that is never used. One such example is a modernization program [11,12], where 36 million euros were spent on unused functionality. Similarly, the UK Office of Government Commerce found that 30–40% of systems to support business change deliver no benefits whatsoever [**?**].

Companies that apply benefit management practices during project execution, apply core agile practices of frequent delivery to clients, and have clients engaged in these practices often had a higher success rate [1]. This suggests that employing appropriate benefits management activities at the right time can be essential to realize benefits and create successful projects. Despite benefit management emerging 25 years ago, and despite studies suggesting that benefit management increases the probability of success, the discipline has not been adopted on a large scale [13].

## 2 Benefit-Cost Management

We describe the design, development, and evaluation of tool support for *benefit/cost analysis management* (BCAM), implementing a generalization of concepts in [7]. The tool support was implemented as an extension to Jira, a widely-used project management and issue tracking tool from Atlassian.[3]

Benefit estimation is more complex than cost estimation, in that benefit is assessed as the system under development's potential to achieve a set of specified goals. The basic concept is that benefit estimates are expressed in benefit points—an analogy to story points for cost estimation [14,15]. Moreover, goals can be organized in tiered structures; examples being *Objectives and Key Results* (OKR) [16], *Lean Value Tree* (LVT) [17] and more traditional project goal hierarchies. A Jira add-on to manage goal hierarchies was presented in [18,19]. The present work focuses on assigning benefit points and cost points relative to a given goal hierarchy.

Table 1 illustrates the basic concept underlying this work. Epics (E1–E3 leftmost) are high-level user stories specifying integral pieces of system functionality (*minimum marketable features* [20] or *minimum viable products* [21]). The epics are assessed by assigning benefit points for their potential effect on O1–O3, which might be effect goals expressing the desired effects of the system on the stakeholders' business and life processes. For each goal, the epics are assigned relative benefit points on a scale from 0–100 according to stakeholder group estimates of the contribution the epics will have on the goal. Goals can be expressed in nominal, ordinal, interval and ratio

[3] https://www.atlassian.com/software/jira

| | O1 (20) | O2 (10) | O3 (30) | Benefit Points | Cost Points | Benefit/Cost |
|---|---|---|---|---|---|---|
| E2 | 85 | 50 | 25 | 49.05 | 25 | 1.96 |
| E1 | 5 | 20 | 70 | 40.05 | 50 | 0.8 |
| E3 | 10 | 30 | 5 | 10.9 | 25 | 0.44 |
| Total | 100 | 100 | 100 | 100 | 100 | 3.2 |

Table 1: Benefit points and cost points assigned to Epics E1–E3, sorted by benefit/cost

terms, which means that achievement of qualitative goals can also be measured. This is extremely important for being explicit on balancing the abundance of efficiency goals with "true" goals; e.g., public well-fare goals [22,23]. In addition, goals can be of unequal importance or *worth*. In Table 1 this is shown by the weights (20, 10, 30) adjoining the goals. This then gives a weighted and normalized sum of total benefit points per epic shown in the benefit points column. Each epic is also assigned a relative lifecycle cost estimate in terms of cost points. Both benefit points and cost points can be estimates using planning poker techniques [14]. The rightmost column in the table shows the benefit/cost ratio which can be used to prioritize epic backlogs and for other benefits management activities.

The tool support described in the following implements a number of simple benefit-cost management techniques based on benefit points and cost points, which, to our knowledge, are not available in project management tools today.

## 3 Research Questions

We pose the following research questions, which are addressed by the tool application development and the ensuing study described below.

**Research Question 1:** *How can the availability of an easily accessible tool help software developers adopt benefit/cost management?*

**Research Question 2:** *How can a tool be simplified in terms of flow to fully utilize benefit/cost management?*

**Research Question 3:** *Can visual representations of benefit/cost metrics in the form of graphs help influence project prioritization decisions?*

## 4 Design Approach

We had the following contextual requirements when we set out to develop the application.

1. The application should be built on an existing project management platform; to avoid introducing yet another project management tool.
2. The project management platform must be widely used and used by relevant organizations.
3. The project management platform must allow for creating native applications.

Although we decided on Jira, several other platforms met the first requirement. For example, GitHub, perhaps the most popular version control service, published a project management tool called GitHub Issues made publicly available in 2022. GitHub Issues fulfils the second requirement, as it has a lot of users. However, GitHub Issues does not meet the third requirement, as it does not provide an API to create a native solution. Jira, one of the most widely used tools developed by Atlassian supports customization and the creation of components that are used by teams on the platform. Also, Jira has functionality to manage and create epics, making it easy to integrate our tool functionality into existing Jira projects. Jira easily meets the second and third requirements.

Within the contextual requirements above, we established a set of design principles for the application itself. The knowledge base of regular requirements is often insufficient, whence designers may rely on empirically or theoretically founded principles, and also intuition [24,25]. In our case, the workings and possibilities of BCAM were largely unknown among practitioners, and eliciting requirements for its tool support was not a sensible option. Rather than relying on the elicitation of end-user requirements, our design principles are based both on those of the techniques themselves in [7] and on our own intuition and experience as professional developers:

1. **User friendliness** – The application should be easy to use and should not contain an overwhelming amount of information regarding text, buttons, and settings.
2. **Integration in Jira** – The application should have the look and feel of the Jira platform and recognizable in terms of visual feedback and flow; including adhering to the Jira colour scheme and using components that match the Jira user experience.
3. **Simplicity** – The application should be simple and not difficult to understand. This is important to make sure that the user can easily grasp how to use the application. Simplicity is vital for acceptance and adaptation [26].
4. **Efficient** – The application should be efficient, minimizing the loading time between different pages. It is crucial for the application to load quickly to reduce load times during project evaluation and analysis, preventing a sense of stagnation for the user.
5. **Scalability** – Whether managing small, agile teams or large-scale agile projects, the architecture should support expansion and contraction without compromising performance or usability.
6. **Security** – The application should be secure and ensure that data are only available to relevant Jira users. This includes following the security guidelines set by Atlassian to ensure that the data is not leaked or accessible to anyone who is not authorized.

## 5 The Implementation

The tool application's codebase is extensive, with the React application consisting of 101 files totaling 9,712 lines of code, and the resolver containing 46 files with 2,277 lines of code. This amounts to approximately 12,000 lines of code in total.

The application was developed using Atlassian's Forge framework; a powerful serverless compute platform designed specifically for the Atlassian ecosystem. It provides a built-in storage API and robust infrastructure that facilitate the development of custom applications with custom user interfaces and functionalities. This framework enabled us to integrate our application with Jira efficiently, ensuring a high degree of scalability without the complexities of managing server infrastructure. Using this platform, we had access to the Jira API, making it easier to connect to existing projects created in Jira. As Jira already has a way of creating, storing, and using epics, we can reuse these epics in our application, without the need to reenter them.

Forge is compatible with React, a widely used JavaScript/TypeScript library for building user interfaces in a modular and component-based manner, supported by a vast and active community, contributing to a substantial knowledge base. React is built in *js*, which simplifies the integration of external packages, which can be prebuilt components or libraries to be integrated seamlessly into React applications, enhancing functionality and reducing development time. The combination of React's robust ecosystem and Forge's serverless architecture enabled us to develop a highly responsive and scalable web application. Since we were dealing with many objects, we decided to go with TypeScript, guarding against type errors when storing data and objects.

Packages for pre-built components and functions can be added to React applications. One such example is Atlaskit, which is developed by Atlassian and contains components similar to native Jira components. This contributes to achieving the user-friendliness design principle, as it uses the Jira colour scheme and component style.

Graphs are powerful tools for clearly and effectively displaying information, and our solution employs several packages related to graph visualization. As Forge and Atlaskit do not provide native functionality for displaying graphs, this had to be sourced from other packages. After reviewing various graph libraries, we opted for *apexcharts* and *react-chartjs-2* due to their distinct advantages and the variety of chart options. We also use the package *ml-regression-polynomial* which provides a set of functions for regression analysis, used for one of the graphs.

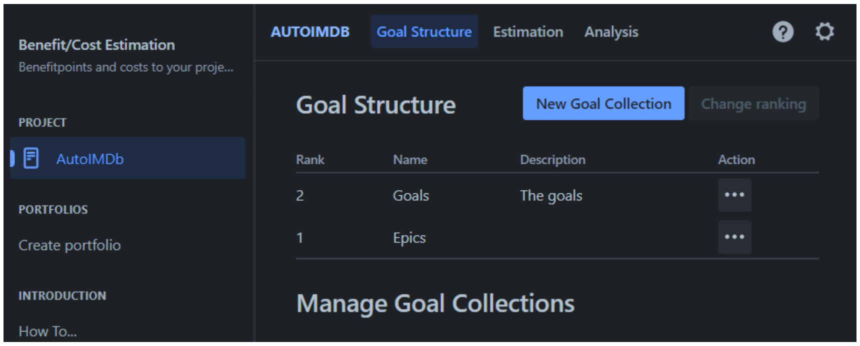
The user interface is divided into various pages, each developed to address benefits estimation and monitoring activities. Below, we will describe the main pages.

To display a lot of information on the different epics, we used the @atlaskit/dynamic-table component, which is a simple way of rendering a table and all of the epics, goals, and other table contents. This component is used on the Goal Structure page, the Estimation page, and the Analysis page. It is also highly adaptable as the style can be changed and allows for React/HTML DOM elements as children, meaning cells can contain more than simple text. We use coloured boxes and action bars in the tables. The component also has a built-in sorting for sorting columns. This is heavily utilized on all pages to allow users to sort by the preferred column.

### 5.1 Goal Structure Page

The *Goal Structure* page (Fig. 1) enables users to declare goal structures at various tiers. For example, one could have project output goals for the functionality of the system under development, goals for the effects that the project output is planned to have on business processes, organizational goals that business processes are intended to achieve, and societal goals that organizational goals should align with.

The goals on a tier are declared on a *Goal Collections* page (Fig. 2), by clicking the "New Goal Collection" button on the Goal Structure page. At the lowest tier, one imports epics from the Jira project timeline. Epics play the role of project output goals.



Goal Structure

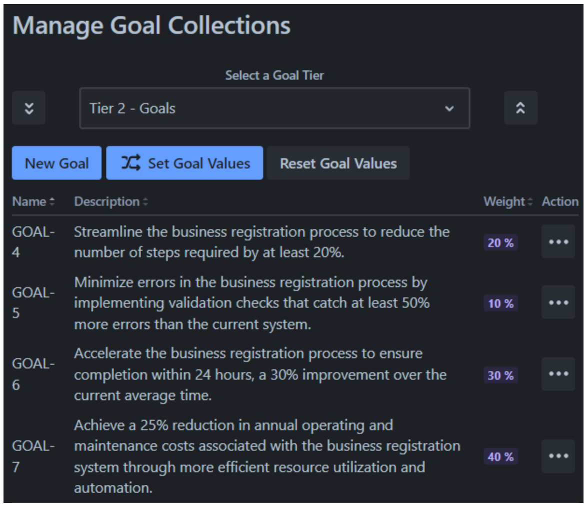
| Rank | Name | Description | Action |
|---|---|---|---|
| 2 | Goals | The goals | ••• |
| 1 | Epics | | ••• |

Fig. 1: The Goal Structure page

The Goal Structure page also enables the user to assign each goal a unique identifier, a description, and a weight. The weight can be set by clicking the “Set Values” button. On the lowest tier, the user can set and reset cost and time estimates for epics (Fig. 3) in relative denomination-free points normalized to 1–100. The cost and time are stored as their own type CostTime directly onto the Jira issue using fields. This means that the data will be retrieved on queries without requiring an extra call to the storage API.

### 5.2 Estimation

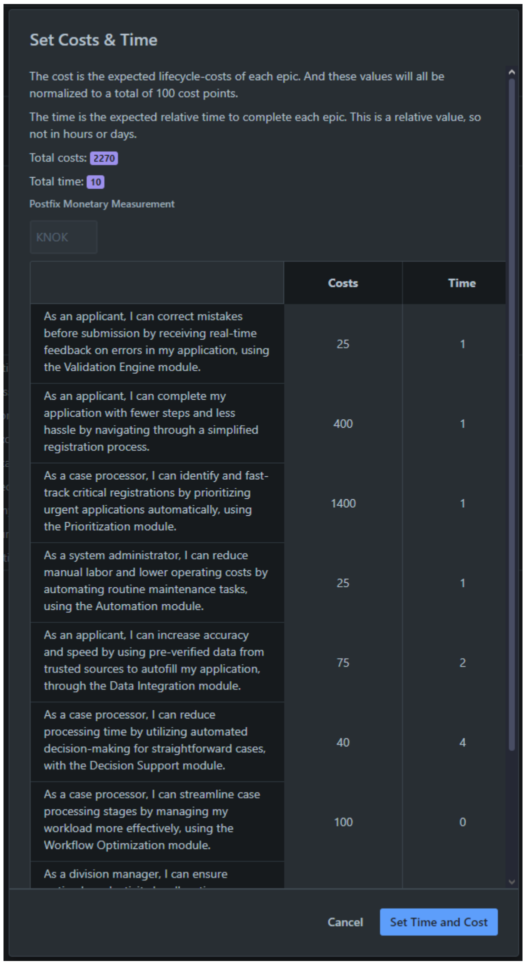
The *Estimation* page is where the benefit points are set. Points can be assigned for each goal on one tier according to the estimated contributions to goals on the tier above. This generates weights for goals and overrides any weights set directly. At the top-level tier,



Select a Goal Tier: Tier 2 - Goals

| Name | Description | Weight | Action |
|---|---|---|---|
| GOAL-4 | Streamline the business registration process to reduce the number of steps required by at least 20%. | 20 % | ••• |
| GOAL-5 | Minimize errors in the business registration process by implementing validation checks that catch at least 50% more errors than the current system. | 10 % | ••• |
| GOAL-6 | Accelerate the business registration process to ensure completion within 24 hours, a 30% improvement over the current average time. | 30 % | ••• |
| GOAL-7 | Achieve a 25% reduction in annual operating and maintenance costs associated with the business registration system through more efficient resource utilization and automation. | 40 % | ••• |

Fig. 2: The Goal Collections page



Set Costs & Time

The cost is the expected lifecycle-costs of each epic. And these values will all be normalized to a total of 100 cost points.

The time is the expected relative time to complete each epic. This is a relative value, so not in hours or days.

Total costs: 2270

Total time: 10

Postfix Monetary Measurement

KNOK

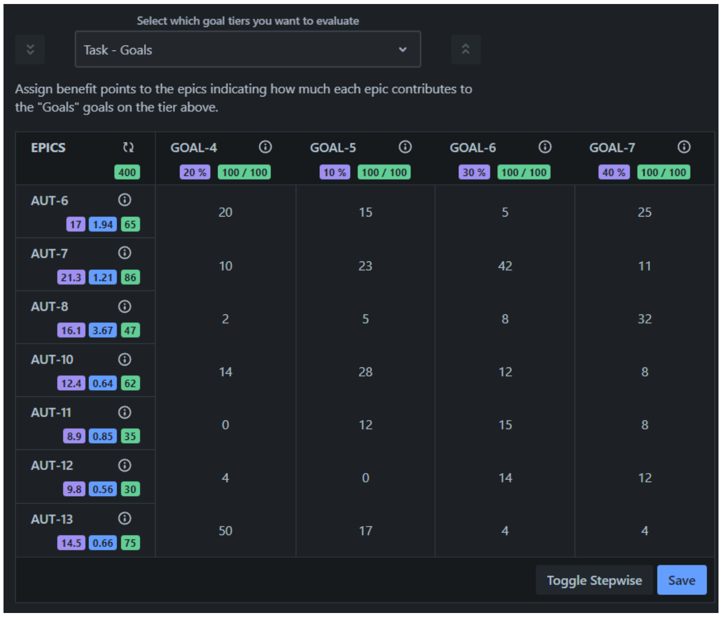
| | Costs | Time |
|---|---|---|
| As an applicant, I can correct mistakes before submission by receiving real-time feedback on errors in my application, using the Validation Engine module. | 25 | 1 |
| As an applicant, I can complete my application with fewer steps and less hassle by navigating through a simplified registration process. | 400 | 1 |
| As a case processor, I can identify and fast-track critical registrations by prioritizing urgent applications automatically, using the Prioritization module. | 1400 | 1 |
| As a system administrator, I can reduce manual labor and lower operating costs by automating routine maintenance tasks, using the Automation module. | 25 | 1 |
| As an applicant, I can increase accuracy and speed by using pre-verified data from trusted sources to autofill my application, through the Data Integration module. | 75 | 2 |
| As a case processor, I can reduce processing time by utilizing automated decision-making for straightforward cases, with the Decision Support module. | 40 | 4 |
| As a case processor, I can streamline case processing stages by managing my workload more effectively, using the Workflow Optimization module. | 100 | 0 |
| As a division manager, I can ensure | | |

Fig. 3: Set Epic Cost and Time

where there is no tier above to evaluate against, Weights *must* be set manually, unless the understanding is that the goal at the top-level tier are of equal worth.

For the lowest tier containing epics, setting benefit points is the way to give benefit estimates for the epics' effects on, e.g., business processes (Fig. 4). The numbers in green boxes displayed on the rows in Fig. 4 are the total benefit points assigned to each epic. The numbers in purple boxes on the rows show the weighted and normalized benefit points according to the weights set on the goals (numbers in purple boxes on the columns), and the numbers in blue boxes are the resulting benefit/cost indexes. All of these fields are updated in real-time to cater for re-estimation during project learning.

To facilitate the methodological point of reducing cognitive load by focusing on one goal at a time [7], it is possible to toggle between the table view in Fig. 4 and a single-



Select which goal tiers you want to evaluate

Task - Goals

Assign benefit points to the epics indicating how much each epic contributes to the "Goals" goals on the tier above.

| EPICS<br>400 | GOAL-4<br>20 % 100 / 100 | GOAL-5<br>10 % 100 / 100 | GOAL-6<br>30 % 100 / 100 | GOAL-7<br>40 % 100 / 100 |
|---|---|---|---|---|
| AUT-6<br>17 1.94 65 | 20 | 15 | 5 | 25 |
| AUT-7<br>21.3 1.21 86 | 10 | 23 | 42 | 11 |
| AUT-8<br>16.1 3.67 47 | 2 | 5 | 8 | 32 |
| AUT-10<br>12.4 0.64 62 | 14 | 28 | 12 | 8 |
| AUT-11<br>8.9 0.85 35 | 0 | 12 | 15 | 8 |
| AUT-12<br>9.8 0.56 30 | 4 | 0 | 14 | 12 |
| AUT-13<br>14.5 0.66 75 | 50 | 17 | 4 | 4 |

Fig. 4: Benefit Estimation Table for Epics. Epics in rows, goals in columns.

column view. This also gives space to display the description of epics and objectives, with full texts displayed by hovering.

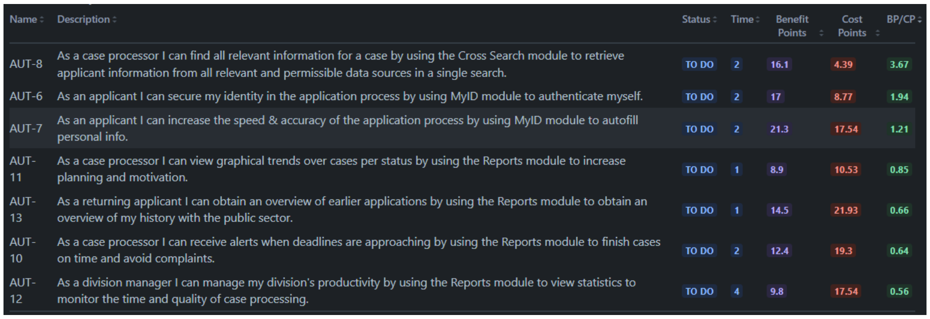
Benefits points can be assigned in various fashions;the default being to distribute 100 points among elements, with support for helping the user: The box containing the number of distributed points alters its colour dynamically to signify user actions: red if the total number of points points exceeds 100, yellow if the count is below 100.

### 5.3 Analysis

The analysis page has a table containing all epics with benefit points, cost points, time points, and benefit/cost indexes (Fig. 5). The table is automatically sorted by decreasing benefit/cost to show the suggested backlog order, where epics with the highest benefit/cost are at the front of the backlog.

The colours adapt to the selected style in the Atlassian/Jira colour template. The colour theme also changes based on what issue board the epic is placed in; for example, Jira's "To Do" issues will have a certain colour, and the issues "In Progress" will have a different colour; this helps in differentiating the state of the different epics in the graphs.

**Points View or Monetary Value View** The quantities on the analysis page can be viewed in two modes; in denomination-less points or monetary values. Instead of weights, it is possible to give monetary values to express the worth of goals, but to avoid inconsistencies, this is only possible for the top-level tier. When this is done, the benefit points are



| Name | Description | Status | Time | Benefit Points | Cost Points | BP/CP |
|---|---|---|---|---|---|---|
| AUT-8 | As a case processor I can find all relevant information for a case by using the Cross Search module to retrieve applicant information from all relevant and permissible data sources in a single search. | TO DO | 2 | 16.1 | 4.39 | 3.67 |
| AUT-6 | As an applicant I can secure my identity in the application process by using MyID module to authenticate myself. | TO DO | 2 | 17 | 8.77 | 1.94 |
| AUT-7 | As an applicant I can increase the speed & accuracy of the application process by using MyID module to autofill personal info. | TO DO | 2 | 21.3 | 17.54 | 1.21 |
| AUT-11 | As a case processor I can view graphical trends over cases per status by using the Reports module to increase planning and motivation. | TO DO | 1 | 8.9 | 10.53 | 0.85 |
| AUT-13 | As a returning applicant I can obtain an overview of earlier applications by using the Reports module to obtain an overview of my history with the public sector. | TO DO | 1 | 14.5 | 21.93 | 0.66 |
| AUT-10 | As a case processor I can receive alerts when deadlines are approaching by using the Reports module to finish cases on time and avoid complaints. | TO DO | 2 | 12.4 | 19.3 | 0.64 |
| AUT-12 | As a division manager I can manage my division's productivity by using the Reports module to view statistics to monitor the time and quality of case processing. | TO DO | 4 | 9.8 | 17.54 | 0.56 |

Fig. 5: Analysis Table

automatically converted to the corresponding monetary values. It is further possible to set the monetary value of cost points, and then the monetary value view is possible throughout the analysis page, and one can toggle between the two views.

**Sorting Criteria** We implemented four different ways of sorting the epics that give different criteria for optimizing benefits realization. Using benefit/cost gives a list of epics sorted by the benefit/cost ratio. However, we also included other ways to sort to see how this alters the suggested order of the backlog. Focusing only on benefit sorts the backlog disregarding cost. The third way of sorting is by using the ratio benefit/time to evaluate what would bring the most benefit in the shortest timespan. The last way of sorting is by (benefit/cost)/time; giving the highest benefit/cost per development time. Sorting follow a two-stage process. Initially, epics are sorted according to the chosen criterion. In the event of a tie, epics are sorted ascending on time.

**Graphs** Graphs provide ways of visualizing complex relationships; especially since the backlog can be sorted meaningfully according to various sorting criteria. The graphs have an information box located in the top right corner that contains a brief description of what the chart is showing. For optimal viewing experience, particularly for users on smaller screens, the graphs are placed in grids that adapt to screen size dynamically.

**Timeline Charts** A simple timeline chart (Fig.6) shows epics sorted according to the chosen criteria while reflecting the relative time estimates per epic. One can focus on smaller sections of the timeline by dragging over an area.

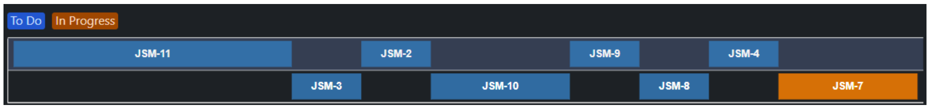


Fig. 6: Simple Timeline Chart

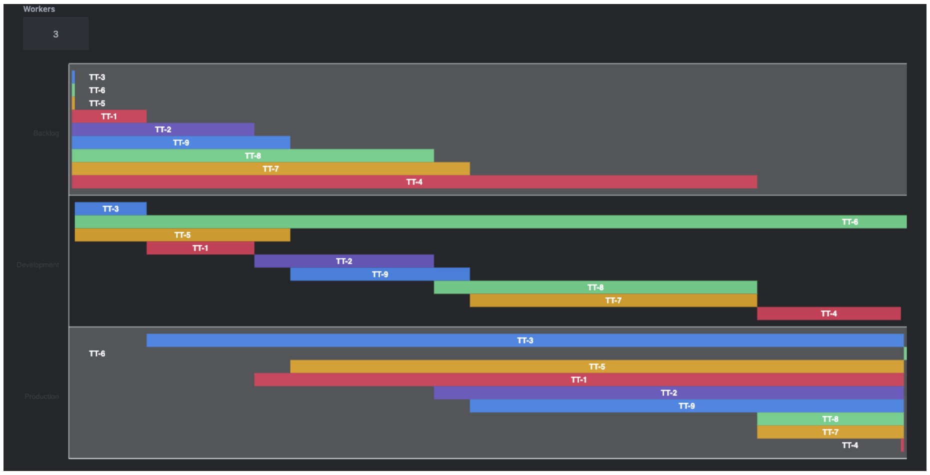


Fig. 7: Parallel Timeline Chart

Usually, there are multiple resources (people or teams) at play, so several epics can be attended to in parallel. Figure 7 shows a timeline chart with the following stages:

1. Backlog, where epics are placed before moved to development
2. Development, where epics are being implemented
3. Production, where epics are deployed and running and potentially generating benefits

The chart shows how each epic is moved from the backlog to development according to the chosen sorting, and how each epic spends its estimated time in development, and then how it is moved to production when done. There are nine resources, so development can proceed in parallel. There are various optimization variants; e.g., to maximize benefit generation in production, the order from the backlog might not be choosing the epic with the highest benefit/cost, but rather one that can be developed quickly.

**Burndown Chart** Following the timeline for the different epics, the burndown chart (Fig. 8, leftmost) displays the remaining benefit/cost over time and has the correct

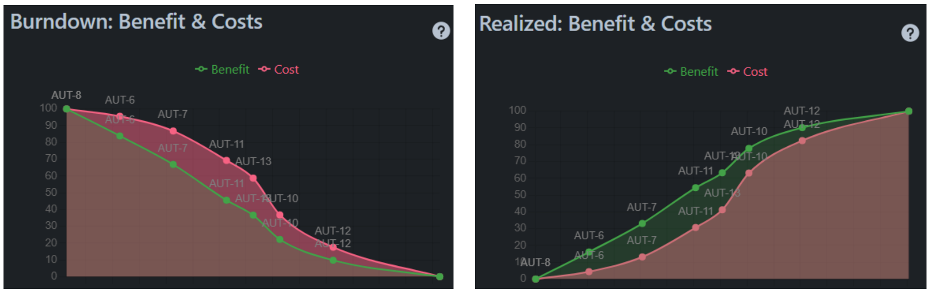


Fig. 8: Burndown Chart (left), Accumulated Estimated Benefits and Costs (right)

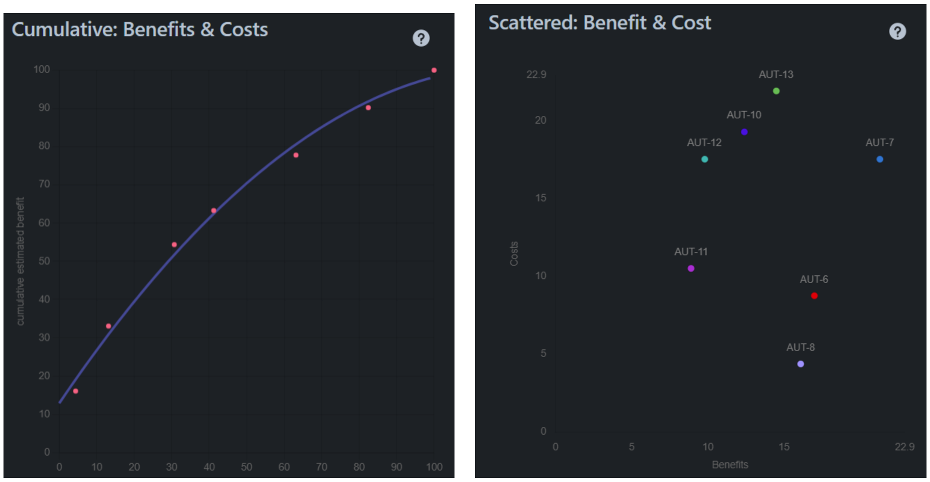


Fig. 9: Cumulative Benefit/Cost (left), Scatter Chart (right)

aspect ratio with respect to time. One gets the remaining benefit and cost at the given time if completed in that order.

**Realized Benefit and Cost Chart** Figure 8, rightmost, shows the accumulated estimated benefit and estimated cost of the completed epics. The points in the graph are spaced horizontally according to their development time. The chart inverts the burndown chart and shows two of many possible metrics of *Earned Business Value Management* [27].

**Realized Benefit/Cost** Figure 9 (leftmost) shows the accumulated benefit/cost index and indicates how the benefit evolves along with the cost and projects the estimated evolution of the benefit/cost index. Here, the chart tapers off, reflecting the order of developing epics with highest benefit/cost first.

**Scatter Chart** In addition to the burndown chart, a scatter chart is also generated with benefit and cost as the x- and y-axis values, respectively (Fig. 9 rightmost). This allows users to notice outliers and get a better overview of the different epics. Each point represents at least one epic, and hovering over a point gives more information about the epics at each point, should there be several. When the denominations are equal (i.e., monetary value on both benefit and cost), the scatter graph is quadratical to reflect the true linear relationship between the axes.

**Pie Charts** Finally, two pie charts provide a simple way of seeing the relative contribution of benefit and cost for each epic (fig. 10). The epics are labeled with their key, and additional information is given on hovering.

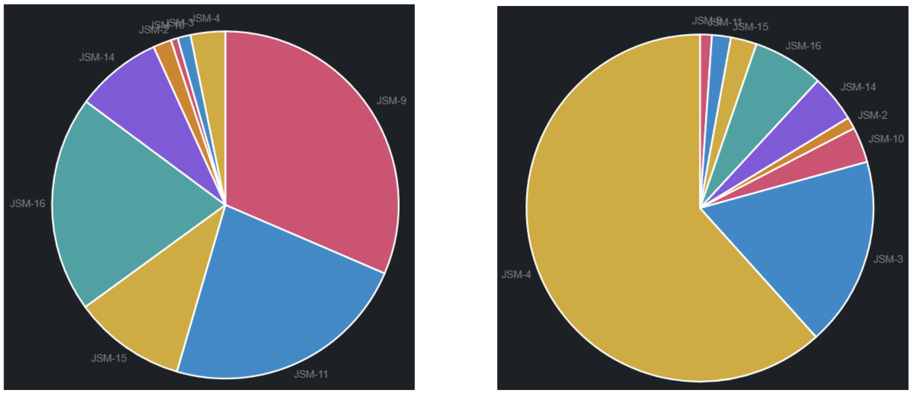


Fig. 10: Benefits Pie Chart (left) and Costs Pie Chart (right)

### 5.4 Introduction Page

Benefit/cost analysis management is not widely known and utilized in today's practices, and we added an introduction page to give users a quickstart to this part of benefits management. The Introduction page contains a simple introduction to the methodology of BCAM and the steps involved. The page is meant as a guide or Q&A, where users can educate themselves on the relevant concepts and important topics. The page does not include any features other than text explanations.

### 5.5 Preview Page

The first time a Jira project is opened, a Project Initialization page prompts for two steps: *Select issue type* – Select what types of issues to target as issues (epics, issues, tasks etc.), and *Select category/label* – Select what status the issue has, e.g. To Do, In Progress or Done (loaded from existing labels).

After this two-step process has been completed, the project is ready to go, and the user may enter the pages described above. We found the Project Initialization page very confusing in terms of the type of issue to target, since the selection of the target is based on the different pages in Jira, for example, from the timeline or the board. To make this process easier, we added a *Preview* page of which issues are targeted. This is done by sending a request for a preview that then returns all items for the issue type without filtering for categories such as To Do, In Progress, or Done. This page is also shared with a Settings page, in case the issue type needs to be changed later.

### 5.6 Onboarding

We also added onboarding using the @atlaskit/onboarding component which is built on the same components as the rest of Jira, so that the design matches the rest of Jira both colour-wise and design-wise. The onboarding process is used to explain what can be done on each of the pages, explain functionality, and ease the user into the usage of the

BCAM tool. An API call is made to determine if the onboarding has been completed for a given user, and onboarding will start accordingly. The user can reset the status with a restart onboarding button. The onboarding functionality is as follows:

1. Project Page – Welcomes the user to using BCAM.
2. Goal Structure Page – Shows how to open the goal structure page and explains what can be done on the page.
3. Estimation Page – Shows how to open the estimation page and gives a brief introduction into what can be done on the page.
4. Analysis Page – Shows how to open the analysis page and briefly explains what can be done and viewed on the page.
5. Create Goal Collection – Shows what button to use to create a goal collection.
6. Manage Goals – Highlights the place where goals worth can be set and explains briefly what it is used for.
7. Set Cost and Time – Highlights and explains how to set the estimated cost and time of an epic on the Goal Structure page.
8. Introduction – Informs the user that more information about how to use the application can be found on the Introduction page.
9. Settings – Shows how to open settings and change the type of issue to target.
10. Restart Onboarding – A way to restart the onboarding.

## 6 Additions and Improvements

We have made a range of improvements to the existing prototype described in [18,19]. The main improvements are the pages described above for Onboarding, Introduction and Analysis described; as well as the following:

- **Adding Cost and Time** – This is one of the most important additions for the tool and allows for the calculation of benefit/cost and time considerations. The graphs use the cost and time to determine the sorting order and the time aspect.
- **Optional Stepwise Estimation** – Adding the ability to set benefit points one goal at a time instead of having to view the entire table of goals and epics helps focus on one relation at a time, while being able to see the description of the epics and the goal-relation that is currently being estimated. This should enhance ease of use.
- **Merging Goal Structure and Goal Tier** – One of the feedbacks on the first prototype [19] was that the goal structure page and the goal tier page (now only goal structure) should be merged to avoid having to switch between pages unnecessarily, this change has saved both loading times while also keeping a clear separation between other tabs in the navigation pane. The merge also made the application more efficient and simple which are two of our design principles.
- **Confirmations** – We developed a more responsive UI when clicking buttons; some buttons such as Reset Values for the goals did not have a warning before executing, which could cause the user to inadvertently reset all the worth points for the goals. Accidentally resetting work one has done is discouraging, so we added a pane warning about the action that is about to be completed and that the action is irreversible.

- **Code Cleanup** – An improvement made to improve efficiency was utilizing *barrel files*. A barrel file forwards the export of a variable, a function, or another type and makes it easier and more concise to import many things that earlier required the knowledge of each file. Some files were also not formatted properly, so we reformatted to increase readability for future developers.
- **Reduced Loading** – Optimizing the code was prioritized as the time spent waiting for pages to load is time that could be spent using the application for its intended purpose. We optimized by placing whatever can be placed directly on the issue as a custom field and reduced calls when opening new tabs. We thus changed the earlier approach of using routes for page functionalities. The ability to go back in the page history and to share a link directly to a Set Values or Reset Values is however lost, but should be easy enough to find by navigating through the application. Reducing loading times supports the design principle efficiency.
- **Improved Security** – Safety is important, and we looked for unnecessary parameters passed from the client side to the server. We found that the project ID was one such parameter passed quite often, and already existing in the the request provided by the Forge architecture. Using the built-in context gave us the ability to ensure the project the query came from, was the one being searched instead of allowing the client side to tamper with the data sent to retrieve other projects' information. Changing this also shed light on the fact that the context provides a lot of useful information one could use in the future for issues such as server-side user restrictions.
- **Improved Project Initialization** – The process of setting up the project has been improved by simplifying the process of selecting the correct issue type.

## 7 Study

The research questions in Sect. 3 are posed at the general level of tool support for benefits management practices; the intention being to address the research questions through several studies over time. The present study addresses the research questions concretely in terms of the tool support (BCAM) implemented in the Jira extension. In the following section, we operationalize the research questions into concrete hypotheses. We state the three main hypotheses; each of them were refined into three to seven sub-hypotheses, reflecting individual features of the BCAM application, but for space reasons, we present high-level results; full details can be found in [28].

**Hypothesis 1** The BCAM application is easily understandable, has valuable (new) features and acceptable loading times.

This hypothesis addresses RQ1, as it tests accessibility and user-friendliness, which are critical factors in facilitating adoption among software developers.

**Hypothesis 2** The integration of BCAM to current project management practices is natural and will be a valuable addition to their project management toolkit.

This hypothesis addresses RQ2, as it tests the flow and receptiveness to BCAM concepts.

**Hypothesis 3** The BCAM Analysis page graphs provide a meaningful visualization of the order of epics in the backlog.

This hypothesis addresses RQ3, since it tests the effectiveness of graphical representations in conveying critical information about BCDAS concepts.

### 7.1 Evaluation

The evaluation of the BCAM tool was conducted in workshops using unsupervised usability testing. The testing and collection of results were completed using a secure online survey tool owned by the University of Oslo. We planned three main testing groups; *Experts* who are familiar with benefit/cost management and project management and expected to be able to provide insights into how they envision the application to meet their needs; *Intermediates* familiar with project management in Jira or other similar frameworks, but not benefit/cost management, who should be able to compare similar applications that they have used and determine if the new features are useful; and *Rookies* consisting of people largely unfamiliar with project and benefits management, meant for assessing the introduction, onboarding, and the conceptual information provided. For our evaluation, we were able to summon three, one, and six individuals, respectively, for the three groups.

Participants were instructed to navigate to the Introduction page, where they were asked to familiarize themselves with BCAM concepts. Then, they were asked to navigate to the Jira project page to configure a project. After this step, participants were asked to complete the onboarding, which automatically starts when the project is configured.

The participants were then asked to create a goal collection and then add four goals that were provided in the questionnaire and to set a value for each of the four goals; specifically, by using the monetary value option. The next step was to set the benefit points for epics on the Estimation page (Fig. 4); where both views, as described in Sect. 5.2 should be tried. Participants were also asked to assign cost and time points for each epic on the Goal Structure page (Fig. 3). Finally, participants were to use the Analysis page, trying different ways of sorting, and using the different graphs to view the suggested backlog of epics based on the values they filled out.

The test involved a series of specific tasks during the steps outlined above. These tasks were designed to evaluate various aspects of the BCAM functionality tool. After completing each task, the participants were asked to answer the questions in Table 2. At the end, participants were asked to provide additional feedback.

| Topic | Question | Scale |
|---|---|---|
| Understanding the Task | How clear was it to you what needed to be done for this task? | 1 - Not clear at all<br>5 - Very clear |
| Ease of Completion | How easy was it to complete this task? | 1 - Very difficult<br>5 - Very easy |
| Helpfulness of Feature | How helpful do you find this feature in the context of the application? | 1 - Not helpful at all<br>5 - Extremely helpful |
| Additional Guidance Needed | Did you feel the need for additional instructions or guidance to complete this task? | Yes/No |
| Suggestions for Improvement | Please provide any suggestions or comments on how this task could be made more understandable or easier to complete. | * |
| Time Spent | Approx. how much time was spent completing the task? | Minutes |

Table 2: Post-task Questions

## 8 Results

*Hypothesis 1*: From Table 3, most of the tasks received high scores on the question: “How easy was it to complete this task?”, with the creation of goal collections and the creation of goals rating highest. The onboarding also seems to be quite easily understandable. The less satisfying aspects were project configuration, the assignment of benfit points for goals, the estimation matrix, and the single column estimation, with the preview feature on the configuration page the least satisfying.

A few of the participants’ comments when asked which view for benefits estimation (Fig. 4 versus single-column) they preferred, and why, were (translated from Norwegian):

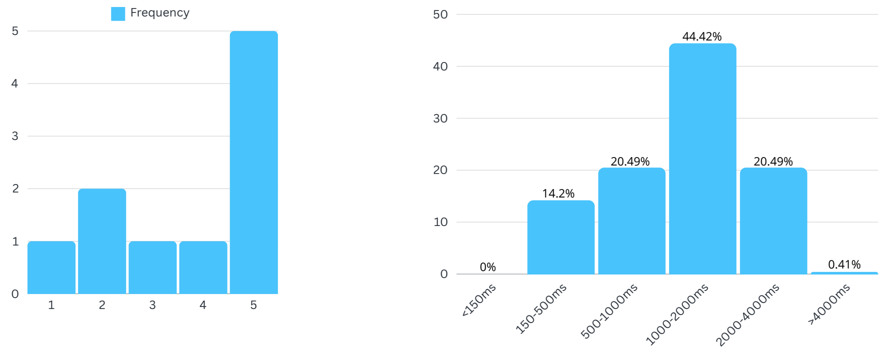
- “I think the single column tab really has potential. A nice feature could be to visualize the difference in benefit contribution by using a bar graph in which you can pull up or down” -*Expert*
- “Full relational view, since it gave me a better overview.” -*Rookie*
- “I guess this is quite context dependent, and could argue that all of them are useful in their own regard” -*Intermediate*
- “I found more useful the full relation view. I was a bit confused in the beginning, because I couldn’t see the info of all the epics and goals (without clicking on the "i" button), but I prefer it because I can have the whole overview and it is easier to see all the estimates with the colours and so on” -*Expert*

The preview feature turned out to be a point of confusion in terms of what to choose from the type and status of the issue. One participant stated “Not sure what the preview feature was”, another participant stated “Did not know which selected issue type to pick, nor did I even think about the preview. Also, did not know which issue status to pick either.”. Another participant noted that they prefer to use the Jira Query Language, which has functionality to filter by issue type and status.

Several participants noted that the use of the application was difficult due to the lack of feedback when interacting with the application. One participant noted: “failed to load because I misclicked something. There were no suggestions about what went wrong”. These feedback issues can be solved by using a a flag component for showing notifications for a limited period of time.

| Task | Mean | Median |
|---|---|---|
| Task 1: Configuring Project | 3.9 | 4.5 |
| Task 1: Preview Feature | 2.6 | 3 |
| Task 2: Onboarding | 4.6 | 5 |
| Task 4: Create Goal Collection | 4.8 | 5 |
| Task 4: Create Goals | 4.5 | 5 |
| Task 5: Assigning Goal Points | 4.4 | 5 |
| Task 6: Estimation Matrix | 4.2 | 4.5 |
| Task 6: Estimation Single Relation | 4.0 | 4 |

Table 3: Ease of Task Evaluation



| Mean | Median |
|---|---|
| 3.7 | 4.5 |

| P50 (Median) | P90 | P95 |
|---|---|---|
| 1346 ms | 3064 ms | 3549 ms |

Fig. 11: Loading Times Satisfaction (left), Actual Invocation Times (right)

Figure 11 summarizes the satisfaction rating for the application's loading time and actual innvocation times. The ratings show variation depending, perhaps, on participants' patience and unstable Atlassian servers. Atlassian provides several metrics, such as invocation time and the invocation success rate. We experienced a success rate of 97.65% on a total of 510 invocations. Note that certain pages require a chain of requests entailing waits for multiple invocations. The maximum chain length in the application is three, which with he median invocation time of 1346 ms gives a loading time of approximately four seconds. Several participants stated that "better" and "faster" loading would have improved their experience.

*Hypothesis 2*: Figure 12 (leftmost bar chart) summarizes the responses to the value of the features themselves (rather than the ease of tasks) and the overall value of the BCAM functionality (rightmost bar chart), as well as the perceived seamless integration into existing, and probable use in future, projects (two bar charts at center). The chart indicates a reluctance to include the application in any current projects, while most participants would consider it for future projects. An intermediate participant notably

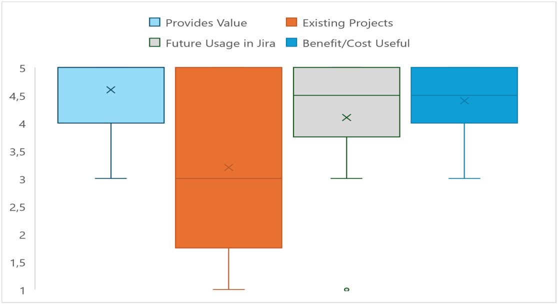


Fig. 12: Responses for Application Features

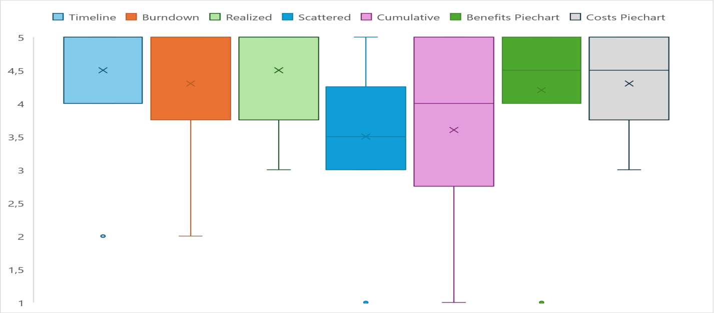


Fig. 13: Box Plots for Charts Feedback

remarked “My perception is that the extension seems natural to Jira, and I don’t know why these features are not already a part of Jira.”

Average self-reported times for completing onboarding and introduction were, respectively, 2.4 and 2.8 minutes. The participants completed the test unsupervised, so there was no training or information beyond the application itself. It is not clear whether the short “training” time was due to low necessity or insufficient quality of the pages.

*Hypothesis 3*: Figure 13 shows boxplots for responses to the usefulness of the charts. Scores are predominantly high. The burndown and cumulative benefit/cost charts are the ones with the largest variances. Five participants, including all experts, found the burndown chart very useful. The lower ratings came from inexperienced participants who did not find the graph particularly useful, suggesting that those without experience in benefit management may struggle to envision practical applications of the graph. However, they did not consider the graph entirely useless. Even with limited experience, these participants could still imagine uses for the graph, indicating some understanding of its purpose. The novice group struggled with interpreting the cumulative benefit/cost graph: “I do not know what to use the cumulative graph for”, indicating a gap in understanding. This suggests a need for educational efforts to bridge knowledge gaps.

## 9 Threats to Validity

There are several validity issues that demand further studies addressing the same research questions. In addition to reliability threats due to the small number of test subjects, we briefly mention threats to internal and external validity [29].

**Internal Validity** Internal validity in the context of usability testing refers to the bias participants have towards a certain response or behaviour [30].

*History* refers to unrelated events that influence the results [31]. Since there is a limited number of experts familiar with relevant concepts, we had to reuse one of the participants who had used the application once before. This could have affected the test results since the participant has pre-existing experience with the application. We fixed points of confusion, such as merging “Goal Tiers” and “Goal Structure” onto one page. Participant might be inclined to give higher scores prompted by such improvements.

*Response bias* pertains to the evaluations of graphs or other visual components being influenced by spurious factors; primarily, aesthetic appeal and previous exposure. Participants may find a certain graph visually appealing, which could confound a more favourable evaluation of functional value. In addition, participants might associate the designs in the application, with other effective tools or applications they might have encountered earlier, which could change their perception of the functional value.

**External Validity** External validity in the context of usability testing refers to errors that capture behaviors or situations not characteristic of our target audience [30].

*Situation effects* refers to factors such as setting, time of day, location, etc. [32]. At least two of the participants stated that they were tired after a long day. However, we did try to mitigate this risk by allowing participants to complete at their own pace and in their own time, while also indicating the expected time necessary.

*Selection Bias* was prominent, since several participants lacked knowledge or experience in project management. We have, however, tried to mitigate this risk by differentiating the responses according to groups. We also sought to use the selection to our advantage to test different aspects of usability.

## 10 Final Words

The Jira extension received enthusiasm, which is promising, given the low uptake of benefits management among practitioners. The present study seems to support the thesis that techniques integrated into project management tools are pivotal for the uptake of practices that are agreed to be advantageous, but that, nevertheless, are not employed.

## Acknowledgements

The authors are grateful to the participants who spent effort evaluating the application.

## References

1. M. Jørgensen, “A survey of the characteristics of projects with success in delivering client benefits,” *Information and Software Technology*, vol. 78, pp. 83–94, 2016.
2. M. Donkersloot and M. Gaikema, “An international research on IT-projects, to leverage success of Dutch IT-projects.”
3. S. S. Tanilkan and J. E. Hannay, “Benefit considerations in project decisions,” in *Proc. Int’l Conf. Product-Focused Software Process Improvement (PROFES 2022)*, 2022, pp. 217–234.
4. P. Svejvig and B. R. Schlichter, “The long road to benefits management: Toward an integrative management model,” *Project Management Journal*, vol. 51, no. 4, 2020.
5. R. Breese, S. Jenner, C. E. M. Serra, and J. Thorp, “Benefits management: Lost or found in translation,” *Int’l J. Project Management*, vol. 33, no. 7, pp. 1438–1451, 2015.
6. J. Ward and E. Daniel, *Benefits Management: How to increase the business value of your IT projects*, 2nd ed. John Wiley and Sons, 2012.
7. J. E. Hannay, *Benefit/Cost-Driven Software Development with Benefit Points and Size Points*, ser. Simula Springer Briefs. Springer, 2021.

8. C. Larman and B. Vodde, *Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum*. Addison Wesley, 2010.
9. D. Leffingwell, *Agile Software Requirements: Lean Requirements Practices for Teams, Programs and the Enterprise*. Addison Wesley, 2011.
10. D. Reinertsen, *Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Publishing, 2009.
11. J. Lystad, "Det er ingen skam å snu – erfaringer fra Mattilsynet og NAV," Presentation given at Conf. of the Agency for Public Management and eGovernment (Difi), Dec. 6, 2017.
12. S. Olaussen, Ø. Tendal, S. Johansen, V. Sem, S. Bråthen, H. Bremnes, E. Grubbmo, and A. D. Ræder, "KSP-rapport nr. 1 for Modernisering av IKT i NAV – Rapport til Finansdepartementet og Arbeids- og sosialdepartementet, Versjon: 1.0," 2015.
13. C. E. M. Serra and M. Kunc, "Benefits realisation management and its influence on project success and on the execution of business strategies," vol. 33, no. 1, pp. 53–66.
14. M. Cohn, *Agile Estimating and Planning*. Prentice Hall, 2005.
15. C. Larman and B. Vodde, *Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum*. Addison Wesley, 2008.
16. R. G. Greenwood, "Management by objectives: As developed by Peter Drucker, assisted by Harold Smiddy," *The Academy of Management Review*, vol. 6, no. 2, 1981.
17. J. Highsmith, L. Luu, and D. Robinson, *EDGE: Value-Driven Digital Transformation*. Addison-Wesley Professional, 2019.
18. S. Krøglid and J. E. Hannay, "Managing information on goals and benefits during development initiatives," in *Human Interface and the Management of Information*. Springer, 2024, pp. 319–337.
19. S. Krøglid, "Building and evaluating a web-based tool for software benefits estimation and management," 2023, Master thesis, University of Oslo, Dept. of Informatics.
20. M. Denne and J. Cleland-Huang, *Software by Numbers: Low-Risk, High-Return Development*. Prentice Hall, 2003.
21. E. Ries, *The Lean Startup*. Board Book, 2011.
22. H. H. Rydén, S. Hofmann, and G. Verne, "The self-serving citizen as a co-producer in the digital public service delivery," in *Electronic Government*. Springer, 2023, pp. 48–63.
23. I. Heggertveit, I. Lindgren, C. Østergaard Madsen, and S. Hofmann, "Administrative burden in digital self-service: An empirical study about citizens in need of financial assistance," in *Proc. 14th IFIP WG 8.5 Int'l Conf. Electronic Participation (ePart)*, 2022, pp. 173–187.
24. A. R. Hevner, S. T. March, and J. Park, "Design science in information systems research," *MIS Quarterly*, pp. 76–106, Mar. 2004.
25. H. A. Simon, *The Sciences of the Artificial*, 3rd ed. MIT Press, 1996.
26. M. Tully, "Investigating the role of innovation attributes in the adoption, rejection, and discontinued use of open source software for development."
27. J. E. Hannay, H. C. Benestad, and K. Strand, "Earned business value management—see that you deliver value to your customer," *IEEE Software*, vol. 34, no. 4, pp. 58–70, 2017.
28. S. Lyngar and M. Storvoll, "A benefit/cost analysis tool for better benefits management," 2024, Master thesis, University of Oslo, Dept. of Informatics.
29. W. M. K. Trochim, *The Research Methods Knowledge Base*, 2nd ed. Atomic Dog Publishing, 2001.
30. R. Budiu. Internal vs. external validity of UX studies. [Online]. Available: https://www.nngroup.com/articles/internal-vs-external-validity/
31. P. Bhandari. Internal validity in research: Definition, threats, & examples. [Online]. Available: https://www.scribbr.com/methodology/internal-validity/
32. ——. External validity: Definition, types, threats & examples. [Online]. Available: https://www.scribbr.com/methodology/external-validity/