A Dashboard for Visualizing Software Engineering Processes based on ESSENCE

Sebastian Brandt, Michael Striewe, Fabian Beck and Michael Goedicke
paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen
Email: firstName.lastName@paluno.uni-due.de

***Abstract*—While traditional project planning approaches focus on precise scheduling of tasks, the ESSENCE standard proposes a higher-level approach that focuses on monitoring. Hence, a new kind of process visualization that picks up ideas of Kanban boards and physical cards is sketched in the standard. This tool paper presents a dashboard application refining, extending, and implementing these ideas based on five use cases posed by two industry partners. It demonstrates that a high degree of support for project management can be achieved by using a relatively small set of visualization means.**

I. Introduction

Software engineering projects are diverse in their nature. Some projects only live for a short period of time, while other projects last for years and involve distributed teams. Some projects are agile, handling new requirements during the whole project, while others follow a straight path from a predefined specification to its exact implementation. A common concern in all projects is to manage the way of working among the people involved in the project, to monitor project progress, and to be able to keep track of all tasks.

The ESSENCE standard [1] intends to tackle these process related aspects of software engineering projects. It defines both a language for software engineering process descriptions as well as a so-called kernel of key elements (named "alphas" and "activity spaces") that are supposed to be relevant in any software engineering project. Each alpha defines a set of states with checklists, which allow to track project progress. An alpha state is considered achieved when all items on its checklist are done and all preceding states of the alpha have also been achieved. Simple process descriptions can be created by forming groups of states from several alphas and thus defining project phases or milestones. A sample visualization of a process in which the alpha states are grouped into four phases is shown in Figure 1. More details can be added to a process description by assigning "work products" to alpha states (e.g., to represent the alpha "Requirements" by a work product named "Use Case Documentation") or "activities" to activity spaces. The ESSENCE standard is not limited to a fixed set of alphas, but also allows to create own ones if needed. These additional alphas may be independent from the existing ones in the kernel or can be added as so-called sub-alphas, which means that they represent a particular aspect of their parent alpha in more detail (e.g., "System Architecture" may be a sub-alpha to "Software System").

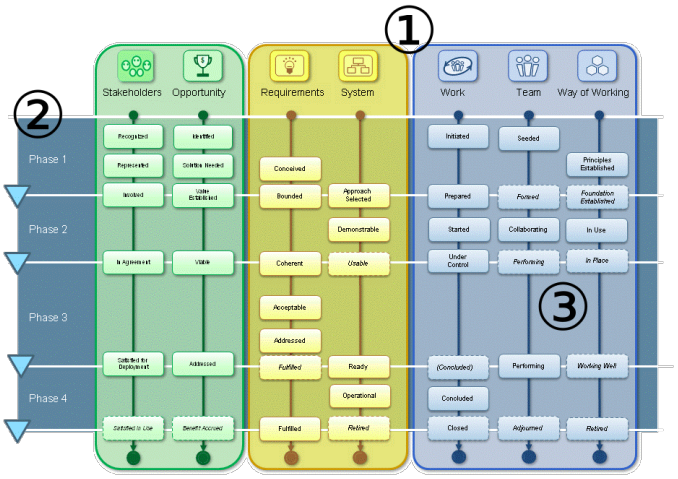


Fig. 1. Sample visualization of an ESSENCE process description based on phases (see Appendix E.2.5 of the ESSENCE standard [1]). The layout uses seven columns for the alphas (1) and groups them into three areas. It uses four rows of different height for the phases (2). States for each alpha run from the top to the bottom and are aligned with the row for the phase in which they should be fulfilled (3). States may occur twice, when they are optional for one phase (indicated by a dashed border) and mandatory for the next phase.

The standard also introduces the notion of physical or virtual state cards. Each card represents and displays an alpha state and the related checklist. A set of cards can be used to visualize the grouping of states in a process description or to tick off checklist items and thus forming subsets of states, where one set contains all completed states, one partially completed states, and the remaining untouched states.

While these ideas offer a natural and intuitive way to visualize and interact with a process, it is impractical to deal with several sets of physical cards when managing several projects in parallel. Moreover, it is impractical to prepare physical cards in different sizes to be equipped both for personal use on travel as well as for collaborative use in a conference room.

This paper elaborates on a dashboard application[1], which is designed to overcome these limitations and add additional features that allow for more use cases than physical cards. The application is intended to be used by project managers

[1]The client-only version of the dashboard application can be accessed under http://www.s3.uni-duisburg-essen.de/pub/essence/demo/Wobbleboard.html.

primarily. The dashboard has been developed in cooperation with the IT departments of MUNICHRE and another insurance company, which both aim to shift all their software engineering projects to processes defined in ESSENCE. The second industry partners has already started to use the dashboard application in their projects.

This paper discusses related work for process visualization, ESSENCE use cases and tool support in Section II. Section III reflects on the requirements for a suitable visualization of process descriptions, the development of prototypes on the way to the final outcome and presents the main visualization features of the application. Section IV concludes the paper and names future work.

II. RELATED WORK AND BACKGROUND

Our dashboard solution can be considered as a process visualization specialized for the ESSENCE standard. We discuss the state of the art in process visualization with a specific focus on software engineering applications. Further, we provide background on ESSENCE with regard to use cases and available tools. This shows how our approach relates to existing visualization approaches and how it complements the current tool support for ESSENCE.

A. Process Visualization

Graphical project planning has a long history. One of the early and still popular type of diagrams are Gantt charts [2], [3], time-by-task diagrams that show planned task durations as bars on a horizontal timeline. PERT (Program Evaluation and Review Technique) network charts [4] show activities as nodes in a graph with dependencies and highlight critical path that potentially cause delays. In contrast to these planning tools, our focus is not on a precise scheduling of tasks, but on monitoring the current state of development for a standardized meta process model.

We use an interactive dashboard as a basis for the monitoring. Existing dashboard solutions for software development focus on team activity [5], current work items and developers [6], or software quality metrics [7]. In context of agile software development, Kanban boards [8]—an overview of work in progress often implemented with sticky notes on a whiteboard—also represent a visualization of the development process. Our solution, which looks at the process from a high level of abstraction, differs from these dashboards focusing on the low-level team activities and tasks as well as code characteristics.

B. ESSENCE Use Cases

During the early stages of the work towards the standardization of ESSENCE, the work was guided by a set of ten use cases [9]. While some of them are relevant to software engineering project managers in their daily duties, some target specific needs of methodologist or other actors. Hence, from these use cases only one ("Plan Based On Method") is fully considered by our visualization, while others are only partially reflected by the needs expressed by our industry partners.

More recently, research has been published on how to use ESSENCE for state-based monitoring of projects [10] and particularly for reflection meetings [11]. As ESSENCE is proven to be useful in these cases, these are also the cases we aim at with our visualization.

Another use case of ESSENCE is to teach and learn software engineering [12], [13], which is also included in the set of ten use cases mentioned above. While this use case may also benefit from visualizations, it is out of scope for this paper. However, learning is not excluded in general, as there are also cases reported in which ESSENCE is used in student projects [14].

C. ESSENCE Tool Support

Simple support for progress tracking based on ESSENCE is implemented in the tool SEMATACC [15]. It allows selecting the current state for each kernel alpha and visualizes the project progress using a spider plot and a bar chart. However, it does not allow for more fine-grained tracking based on single checkpoints and also offers no means to describe processes.

Commercial tool support is provided by IJI with their tools PRACTICE WORKBENCH[2] and ALPHA STATE EXPLORER APP[3]. The former offers support in designing and documenting processes, but not in tracking actual project progress. The latter allows defining processes in terms of milestones and lifecycles as well as tracking progress based on these processes. Although its target audience is somewhat similar to the application presented in this paper, it has a very different design scope by offering less information and focusing on small displays. It is also limited to a fixed kernel and not intended to allow multiple users working on the same process description.

III. DASHBOARD APPLICATION

The limitations of physical cards mentioned in Section I defined both the need for a virtualized solution and some requirements that a new application should fulfill. Our industry partners provided input mostly in form of (sometimes contradicting) suggestions for concrete features fulfilling their specific needs. Hence, we had to work iteratively in order to elicit more generalized and harmonized requirements.

A. Initial Requirements

From the various needs of our industry partners, we could derive five major use cases describing the initial requirements for the dashboard. The first use case is to enable the user to get an overview of a process and to learn a new process (Use Case 1). The user should only see the top alphas and the alpha states grouped into the phases of the process. This way, the user can see the progressions of the different alphas throughout the process. To learn a process, a user should be able to access additional information about an alpha state, like its checkpoints or linked progression of a work product. The user should not have to switch the view to get this information.

[2]https://www.ivarjacobson.com/esswork-practice-workbench

[3]https://www.ivarjacobson.com/alpha-state-explorer-app

Instead, the user should be able to filter the view to include or exclude specific types of information.

In addition, the dashboard should act as a process reference (Use Case 2). The user should be able to easily and quickly access all the information about the process needed to fulfill the daily activities. The information should either be directly included in the dashboard or easily accessed directly from the dashboard, e.g., by a link. Referenced information could be a detailed description of a practice used in the process or a detailed description of a work product containing guidelines on how to create this work product.

Besides viewing and learning the process, users should also be able to document the project progress on the dashboard (Use Case 3) and to determine and assess the current project status using information provided by the dashboard (Use Case 4). The dashboard should enforce a common way to describe progress in a project by including the checklists for each state and by marking checkpoints that are fulfilled as checked. This way, determining and assessing the current state of a project can be done using the rules defined in the ESSENCE standard. Thus, the current state of the project is defined by the current states of the included alphas.

It was explicitly stated by the industry partners that no automated detection and visual representation of potential "pain points" in a project are required. It was assumed by project managers that automated warnings could draw too much attention to problems and thus conceal project success in other places. Moreover, it was assumed that automated warnings could mislead project teams to be dishonest in tracking their progress in order to avoid triggering any warnings.

Finally, the dashboard should support the team in planning the next steps of their project (Use Case 5). Based on the assessed status of the project, the team can plan the next steps by defining tasks based on the checkpoints needed to achieve the next alpha states and the necessary progressions of the work products from the current alpha state to the next one.

In addition to the use cases discussed above, there were also some more requirements regarding the handling of the application. Most importantly, the application should not be limited to a single process description, but should be able to work with any process description that follows the general style of grouping alphas into phases. In particular, it should be able to work with an arbitrary number of additional alphas, sub-alphas, and even sub-alphas of sub-alphas. Moreover, it should be able to work with an arbitrary number of phases and an arbitrary number of states per alpha grouped into one phase.

B. A Brief History of Prototypes

Before we started to develop the dashboard application, our industry partners created four major prototypes, which were only partially interactive and did not fulfill all use cases. However, they allowed to identify key benefits or drawbacks of particular means of visualization or interaction and are hence discussed here briefly.

The first prototype consisted of an extensive deck of POWERPOINT slides and covered only Use Case 1 and Use Case 2. A lot of hyperlinks were placed on the slides to allow for quick navigation between overview slides and detail slides. The prototype was used in various training presentations but turned out to be too hard to change and update, as new information needed to be entered several times on different slides. Consistent styling of all slides was an additional issue in this prototype. Consequently, a second prototype was created covering the same use cases and being based on HTML and JavaScript. This prototype contained all the information in one view with consistent styling and offered options to filter the information provided. The main drawback of this prototype was, that the information was included directly in the HTML document and there were still some redundancies for different kind of display options. To overcome these shortcomings, the next prototype was designed based on the idea to enter all information only once into a central JSON document and to use templating to generate the view dynamically based on the process definition provided. The prototype was fully implemented and parts of its source code have been reused in the dashboard application.

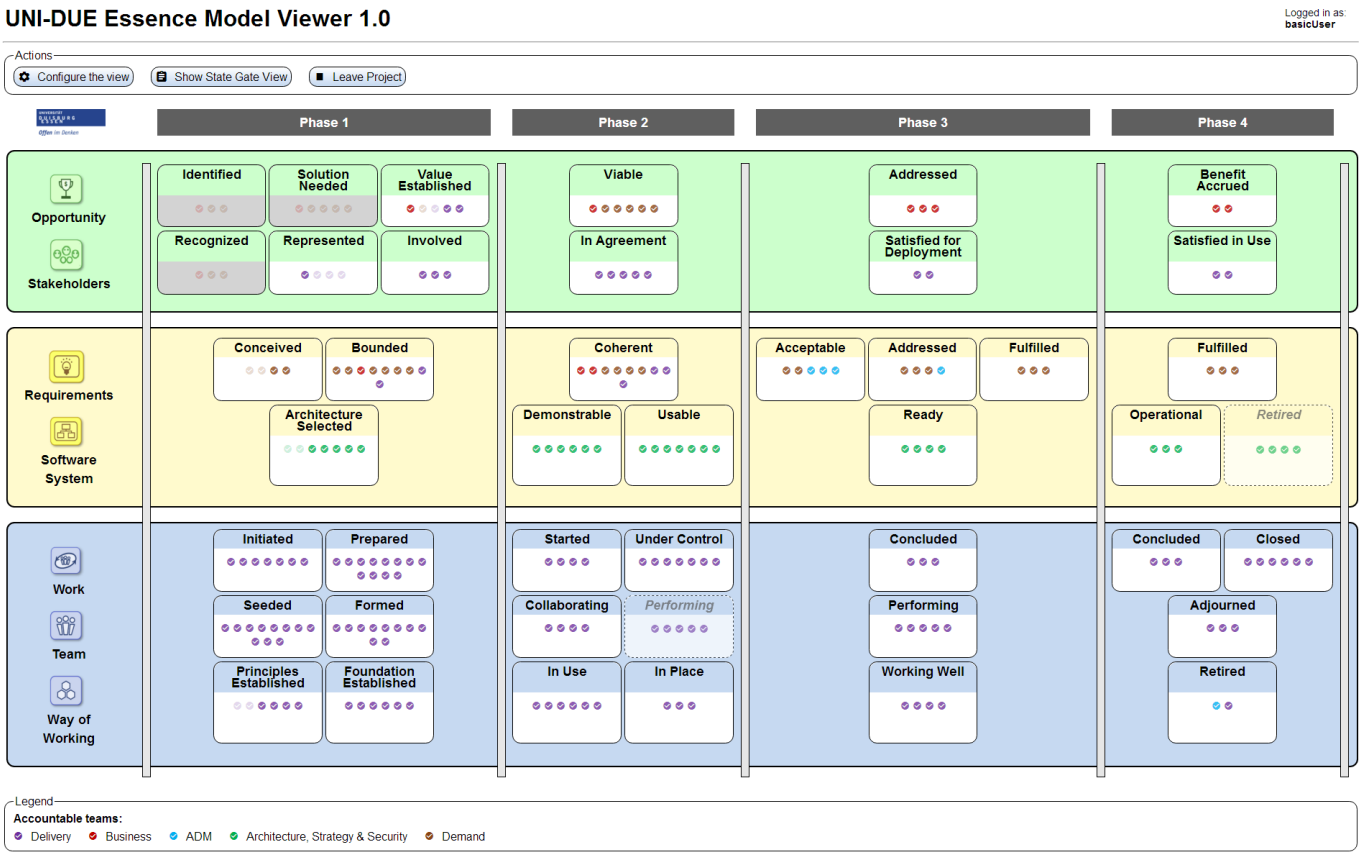
Independent of these three prototypes, a fourth one has emerged from the need to cover Use Cases 3 to 5. It consists of an EXCEL sheet that contains a process description and uses color formatting of cells to keep track of the project progress. As the Excel sheet is not write protected, it also allows to customize the process for a particular project or keep track of any additional information. However, the latter turned out to be a major drawback, as different versions of the EXCEL file easily diverged from each other and hence many different and inconsistent process descriptions were in use.

C. The Interactive Dashboard

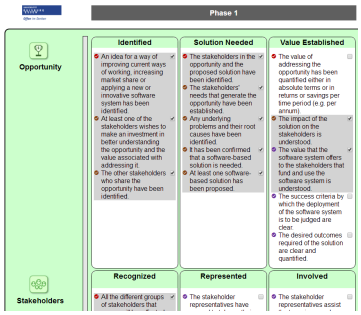
The interactive dashboard has been developed as a client-server application using HTML, CSS, and JavaScript for the browser front end, Java EE for the server back end, and RESTful webservices for the communication between front and back end. Process definitions are loaded from JSON files stored on the server and the application is able to handle several process descriptions in parallel for different projects. So far, the application does not support customization of process descriptions directly in the client, but this is planned as a future feature.

Once a new project has been created and a process description has been selected for it, users can start to interact with the main view shown in Figure 2(a). It shows the process definition organized in rows for each alpha and columns for each phase. Boxes with rounded corners represent the alpha states and thus reflect on the notion of cards. If a process description defines states to be optional for a particular phase, they are shown with a dashed border and slightly opaque (see two states in Figure 2(a)).

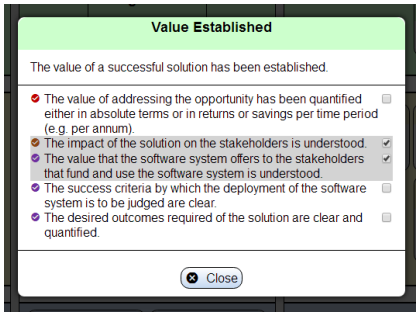
Besides the arrangement of states into phases, the main view also provides information on the checkpoints for each state via colored icons. Intensly colored icons indicate check-



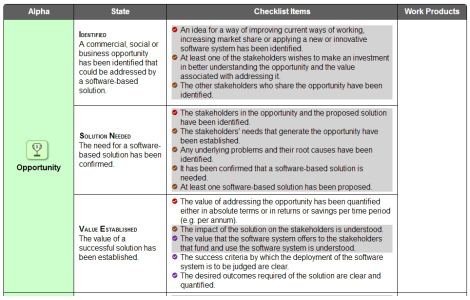
(a) Main process dashboard. The view shows summarized information for all states with completed states in gray, lightly colored icons for fulfilled checkpoints and intensely colored icons for unchecked checkpoints. Optional states are shown in a semi-transparent style with a dashed border.



(b) Upper left corner of the dashboard from Figure 2(a) with additional information activated. Checkpoint texts are now directly visible for each state.



(c) Information overlay that can be triggered from the view in Figure 2(a) by clicking on the state headers. It provides state descriptions and checkpoints details.



(d) Top part of a summary view for one process phase. The right column shows work products if they are defined in the process. If so, they also appear in all other views.

Fig. 2. Screenshots from the dashboard application.

points that are not yet ticked off, while lightly colored icons represent fulfilled checkpoints. If all checkpoints of a state are fulfilled, the state is colored gray (as shown for some states in the top left corner of Figure 2(a)). Thus, the main view supports both Use Case 1 (by providing an overview on the process) and Use Case 4 (by indicating progress). In order to support Use Case 1 independent of an actual project, the dashboard includes an identical view for plain process descriptions. However, the process view cannot be used to assess progress. Notably, one industry partner mentioned that the main view already shows too much information for a project and suggested to hide the icons completely.

The main view offers two alternatives to tick of checkpoints: First, it allows activating checkpoint information, so that each state card shows checkpoint texts and a checkbox that can be ticked off (see Figure 2(b)). Second, users can click on the titles of the state cards in the main view to open an information overlay (see Figure 2(c)). This overlay contains

checkpoint texts, checkboxes, and additional textual information on the respective alpha state. Both ways of interaction directly support Use Case 3 (documenting project progress) and Use Case 5 (planning next steps). Since the additional textual information on the alpha state shown in the overlay can also be activated for the main view, both ways of information presentation also support Use Case 2 (using the dashboard as a process reference).

As the main view always shows all phases and only allows to activate additional information for all or none of the states, one industry partner suggested to introduce an additional summary view focusing on one process phase. Figure 2(d) shows the upper part of this view for the first phase. It provides all available information and also uses gray coloring for fulfilled checkpoints. The main focus of this view is to support Use Case 4 and Use Case 5, helping the user to assess the current state of a project and to plan the next steps. Use Case 3, the documentation of the project progress, is currently not supported in this view, but will be added as an optional feature in a future version of the dashboard.

Both the main view and the summary view support an optional feature of colored checklist icons (which is in use in the figures). The coloring of the icons is used to define accountabilities of different teams or departments for a set of checkpoints. A legend at the bottom of the main view explains which color is mapped to which team or department. The main view includes a filter which makes it possible to only show those checkpoints a given team is accountable for. Using accountabilities is optional, because one industry partner stated, that accountabilities will not yield any advantage but may foster silo thinking, therefore degrade collaboration and hence have negative effects on the teams' performance.

The main view as shown in Figure 2(a) fits into the full screen browser mode for screens with a resolution of 1920 x 1200 pixels. However, this only applies to process descriptions using not more than seven alphas. Any additional alpha would add an additional row to the dashboard and hence exceed the vertical screen size. Notably, this was considered much more acceptable for the users than the original layout used in the ESSENCE standard (see Figure 1) in which additional alphas consume horizontal space. Nevertheless, the width of the dashboard may also be not sufficient for comfortable reading of contents when additional information are activated. Users can thus configure the view to be wider than the horizontal screen size. This has also been used for taking the screenshot in Figure 2(b), in which the state cards are wider than in Figure 2(a).

IV. Conclusions and Future Work

With the development of the interactive dashboard, we were able to create an application that supports all five use cases and avoids most drawbacks of the prototypes. The notion of physical cards is preserved in the layout, both for the main view and the overlay. A small set of simple visualization features such as coloring and opacity turned out to be sufficient to include the most important information into the view.

Future work includes two main aspects: First, there are still some features to be implemented that are considered useful in the context of our requirements. This includes in particular features to toggle the visibility of checkpoints, states, or alphas, and thus to allow for more customization of a given process for a particular project. Customizing the process by drag-and-drop actions on alpha state cards would also ease planning and help transfering an important piece of user experience from the physical world into the the interactive application. Second, we plan to run user studies with our industry partners to evaluate the usefulness and actual usage scenarios of the dashboard application in practice. These studies will be organized according to the five use cases and focus on the question whether the particular features of the dashboard application indeed meet the users' needs.

References

[1] *Essence - Kernel and Language for Software Engineering Methods*, OMG Std., Rev. 1.1, Dec 2015. [Online]. Available: http://www.omg.org/spec/Essence/1.1

[2] H. L. Gantt, *Work, Wages, and Profits*, 2nd ed. Engineering Magazine Company, 1913.

[3] W. Clark, W. N. Polakov, and F. W. Trabold, *The Gantt Chart: A working tool of management*. Ronald Press Company, 1922.

[4] A. A. Mota, L. T. M. Mota, and A. Morelato, "Visualization of power system restoration plans using CPM/PERT graphs," *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1322–1329, 2007.

[5] J. T. Biehl, M. Czerwinski, G. Smith, and G. G. Robertson, "FASTDash: a visual dashboard for fostering awareness in software teams," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI. ACM, 2007, pp. 1313–1322.

[6] M. R. Jakobsen, R. Fernandez, M. Czerwinski, K. Inkpen, O. Kulyk, and G. G. Robertson, "WIPDash: Work item and people dashboard for software development teams," in *IFIP Conference on Human-Computer Interaction*, ser. INTERACT. Springer, 2009, pp. 791–804.

[7] F. Deissenboeck, E. Juergens, B. Hummel, S. Wagner, B. M. y Parareda, and M. Pizka, "Tool support for continuous quality control," *IEEE Software*, vol. 25, no. 5, pp. 60–67, 2008.

[8] M. Ikonen, E. Pirinen, F. Fagerholm, P. Kettunen, and P. Abrahamsson, "On the impact of Kanban on software project work: An empirical case study investigation," in *Proceedings of the 16th IEEE International Conference on Engineering of Complex Computer Systems*, ser. ICECCS. IEEE, 2011, pp. 305–314.

[9] N. Ignaciuk, "Analysis of the Completeness and Quality of the Essence specification," Master's thesis, University of Duisburg-Essen, 2014.

[10] C. Péraire and T. Sedano, "State-based Monitoring and Goal-driven Project Steering: Field Study of the SEMAT Essence Framework," in *Companion Proceedings of the 36th International Conference on Software Engineering*, ser. ICSE Companion 2014. ACM, 2014, pp. 325–334.

[11] ——, "ESSENCE Reflection Meetings: Field Study," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE '14. ACM, 2014, pp. 25:1–25:4.

[12] M. Kajko-Mattsson, "Tackling the incompleteness of software engineering education with the ESSENCE kernel," in *Proceedings of the 2015 International Conference on Software and System Process, ICSSP 2015*, 2015, pp. 179–180.

[13] J. Pieper, "Discovering the essence of Software Engineering an integrated game-based approach based on the SEMAT Essence specification," in *2015 IEEE Global Engineering Education Conference (EDUCON)*, 2015, pp. 939–947.

[14] P. Ng and S. Huang, "Essence: A framework to help bridge the gap between software engineering education and industry needs," in *26th International Conference on Software Engineering Education and Training, CSEE&T 2013*, 2013, pp. 304–308.

[15] D. Graziotin and P. Abrahamsson, "A Web-based modeling tool for the SEMAT Essence theory of software engineering," *Journal of Open Research Software*, vol. 1, no. 1, 2013.