STUDIA INFORMATICA 2015
Volume 36 Number 1 (119)

Piotr GAJOWNICZEK, Rafał ZAWIŚLAK
Lodz University of Technology, Institute of Automatic Control

# CROSS TEAM, MULTI TASK, MULTI FLOW KANBAN APPLICATION

**Summary**. The main goal of the paper is to present a new application based on Agile-Kanban methodology. The application allows not only to visualize the development stage but also includes the possibility of inside-system cross-team cooperation for multi task and multi flow project. In the last chapter some examples of the usage of the application, gathered during the tests have been shown and concluded.

**Keywords**: Agile-Kanban methodology, IT smart application, work optimization

# MIĘDZYZESPOŁOWA, WIELOZADANIOWA, WIELOWĄTKOWA APLIKACJA TYPU KANBAN

**Streszczenie**. Głównym celem artykułu jest przedstawienie nowej aplikacji bazującej na koncepcji metodologii Agile-Kanban. Aplikacja pozwala nie tylko na wizualizację aktualnego stanu zaawansowania nadzorowanych za jej pomocą projektów, ale również na śledzenie przepływu informacji pomiędzy współpracującymi zespołami, w ramach wielu realizowanych równolegle podprojektów (zadań). Autorzy przedstawili nowe podejście do znanej koncepcji, omówili również testy aplikacji i perspektywy jej dalszego rozwoju.

**Słowa kluczowe**: metodologia Agile-Kanban, inteligentna aplikacja, optymalizacja czasu pracy

## 1. Introduction

The Development of software that fulfills all of customer's expectations is a complicated and difficult task. In fact the process is so complex, that only less than 40% of software projects fully succeed [14]. The problem is more focused in large scale projects, where the suc-

cess rate is smaller by a factor of 4. Great majority of projects tend to run over the budget and schedule, while delivering less content [2, 3].

There are many elements that can influence the project execution, and influence its chances for success. One of the most important factors is project environment changing in time. Development of large scale projects requires extensive period of time. Nowadays, the technological advance is very rapid, especially in the IT related domains. The developers, in order to stay competitive, must adapt to the cutting-edge technologies. The change of technologies used in an ongoing project is rather hazardous. Another problem is frequently introduced by the customers. Over the time, they tend to change their expectations and requirements towards the software product.

In order to deal with those issues and achieve success, developers have to adapt to ongoing transformations of the environment. However, traditional plan-driven approaches to software development do not provide such flexibility. Most commonly used Waterfall model for software development is a sequential process. It consists of steps that follow each other. First phase of the development is to gather and document all of the requirements for the project. Only after that documentation is ready, the design and later implementation steps can proceed. However, if anything to requirements changes, the whole procedure must be repeated. With each change, the project execution is hindered and more vulnerable to failure.

Agile methodologies, based on principles formulated in Manifesto for Agile Software Development [15], might be used to deal with aforementioned problems. In this paper, Agile methodologies, and how they affect project execution, will be described in more detail. Also a new solution will be proposed, to deal with some drawbacks and problems related to the use of Agile.

## 2. Agile development

### 2.1. Reasoning

Agile development is a group of development methods based on incremental and iterative approach. This means that instead of one big life cycle, the project is divided into smaller parts – iterations. Each iteration is a separate development cycle, during which new functional parts are added to the software, resulting in a working product. Splitting the project into smaller parts is beneficial for several reasons. First of all, such approach allows developers to initially focus only on most important features. Thus, after some number of iterations, they are able to deliver a product, with all key functionalities, that the customer can use. Another advantage is, that the developers can adapt to changing environment between iterations.

One of core Agile principles is cooperation between the developers and customers. The goal of the project is not to deliver product, but to satisfy the customer. This is achieved by early, and frequent delivery of valuable software, as well as by taking into account the feedback given by the customer.

### 2.2. Kanban

As mentioned before, Agile is a group of development methodologies. Most popular examples would be Scrum, Kanban, eXtreme Programming or Test Driven Development [1,8]. Let's focus on Kanban in more detail, as it is very popular either as a development method itself, or as a supporting method used in conjunction with other ones, as for example Scrum. Kanban is a visual process management system, that aims at improving the development process through evolutionary change. The basis for Kanban is **visualization of workflow**. Most commonly it is realized by means of a board (magnetic or cork). The board is divided into a number of columns, each representing different step or phase of the development, like for example 'To do', 'Designing', 'Implementing', 'Testing'. A card (post-it note, sticker, pinned piece of paper) is used to represent a single work item – e.g. certain functionality the software must implement - and by placing it in particular column on the board, we indicate that it undergoes corresponding phase of the development. Thus everyone on the team has the possibility to observe and evaluate the progress of work.

Another Kanban practice is to **limit work in progress** [9]. We should not have too many tasks undergoing at the same stage of the development. We should focus on finishing work on ongoing tasks, rather than starting other ones. Only fully implemented functionality provides value to the product. Thus it is meaningful to focus on developing it as fast as possible (reduce the lead time - time required to complete the work on single task), and add value to the resulting software. As ***"Longer lead times seem to be associated with significantly poorer quality"*** [1], through limitation of work in progress, improvement of software quality can be achieved.

Visualization of the workflow makes the process of development explicit, what helps to comprehend it. For example it is easier to detect bottlenecks in the process. Once a bottleneck is detected, actions may be undertaken to eliminate it, thus improving the flow and optimizing the development process. This could also be a significant factor influencing the lead times, quality of work or just the satisfaction of the personnel. It also simplifies the adaptation to changes of the environment.

# 3. Drawbacks of agility and how to deal with them

## 3.1. Issues outline

Use of Agile methods resolves some of the problems that influence the success of software development projects, but it also imposes some limitations. One of them is that, due to globalization effect, developers working together on a project tend to be geographi-cally distributed. This is a major problem when using Kanban system for visualization, as it traditionally requires a physical board placed in one location. All developers, working from different area, are not able to use it.

Second problem is related to scalability of Agile. Kanban boards are useful when one team is concerned. For large scale projects however, large number of developers must be involved, whereas size of one team should not exceed 15 people. Again, divide and conquer approach is used, where multiple teams are created, and the "team of teams" is formed. In such a case, each team should have its own board for visualization, as presenting flow for the whole project will usually be unintelligible. Such approach leads to a problem with maintaining consistency between all of the boards.

## 3.2. Possible solutions

There is one simple solution, addressing the first of the aforementioned issues - instead of using a physical board, one can take advantage of equivalent electronic tool. Many such tools already exist on the market, both commercial and free, such as for example: *Trello, Jira, LeanKit Kanban* or *VersionOne*.

Some of them provide extensive functionality, however imposing also high complexity of the tool, which makes the application less user friendly. Other ones, usually those available free of charge, provide good functionality together with clear and user friendly interface. However in most cases, all of user data is stored on the producers' servers, which violates the information security standards in the majority of companies. **On the other hand, there are no solutions to the second problem.** All of those programs are designed to be used by Agile teams, **however they do not include the possibility of inside-system cross-team cooperation**.

# 4. New approach

New product has been developed, to fill the niche on the market. The web and data-base techniques as well as Java have been used [4, 5, 7, 10, 11]. Its main functionality is to (apart from serving as an electronic Kanban board) support connecting boards with each other, and allow transferring work items between them. Thereby it will not only organize the development processes execution inside one team, but throughout the whole life cycle of a larger scale project. It will also serve as a platform for work organization purposes across different levels of hierarchy in the organization.

## 4.1. Tool features

The basic functionality of the application is to allow management of a single Kanban board. Users have the possibility to manage columns on a given board, including creation, deleting, renaming or defining work in progress limits on columns. System indicates limit status with color adjustments - yellow color indicates that the declared limit has been met, while red shows it has been exceeded. User that exceeds the limit is also prompted to specify the reason. Other feature concerns work item management - creation, movement, editing but also reporting work hours. Users are able to estimate the amount of time required to finish the task, and report time spent working on it. Those values can be later used to prepare statistical analysis of the flow, or can be exported and used to prepare work time reports. As the system is targeted to be used across different levels of hierarchy, on each level the work item is of different scope – from "epics" describing operation of whole subsystems, through "user stories" depicting single functionality, to atomic development tasks. Managing tasks also allows splitting one task into smaller parts. Users are able to define links connecting columns of different boards to realize the flow of work between those boards, and assure data consistency.

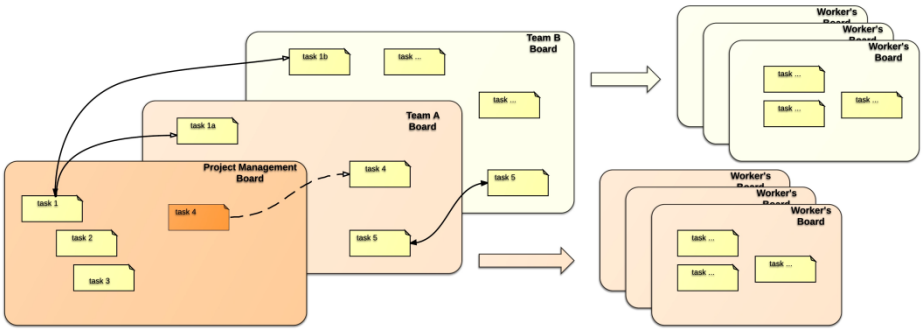


Fig. 1. Tasks can be split, shared or delegated to the different boards
Rys. 1. Zadania mogą być rozdzielane, kopiowane bądź przesuwane do różnych tablic

Those links can be setup to delegate tasks to other boards, and report work progress on the delegated tasks. As it can be seen in figure 1 tasks can be split, shared or delegated to the different boards.

To ensure data security several measures are implemented to the system. First of all, users need to authenticate using password. The login process can be realized within the application itself, or with the use of some external system. What is more, users have defined permissions to boards. There are four levels of permissions: administrator, modifier, observer and none. Users with no access permissions to a board cannot see its content. Observers can see all of the columns and tasks on the board, but are not able to perform any actions on them. Modifiers can manage tasks and administrators are able to manage columns and links, as well as assign permissions for other users. Permissions are assigned in two ways: user specific or default.

Last but not least, the tool provides some statistical analysis of the flow for each table. It calculates column-wise indicators expressing how much time users report working on tasks and average time work items spend in the column. Value Added Ratio is calculated, as a quotient of those two values, and can be used as a efficiency rating. For each board also average lead time, and cycle time (interval of time between finishing tasks). Additionally system generates plots: cumulative flow diagram, depicting distribution of tasks in columns versus time, or burn down chart, showing remaining work versus time.

### 4.2. Example

To better explain the features of the application, two simple examples will be considered. The first one will concern the workflow visualization, whereas the second one will present what information can we extract from statistics.

**Flow**

The flow assumes that there are several teams of developers working on the project, each consisting of a number of employees. The hierarchy of boards, which will be used to visualize the workflow, will be as presented in figure 2.

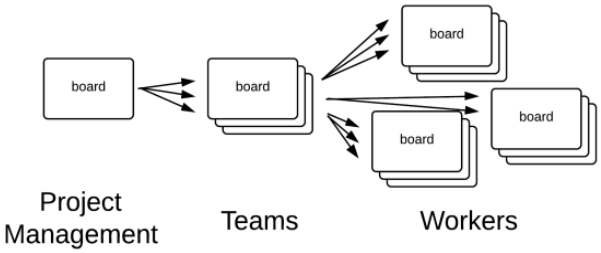


Fig. 2. Board hierarchy
Rys. 2. Hierarchia tablic

Figure 3 presents exemplary flow during the development of some system. The first column contains consecutive steps that are taken during the development process. Second column represents the corresponding actions that has to be undertaken, in order to visualize these steps. Third column shows what additional operations does the system perform during the user activity.

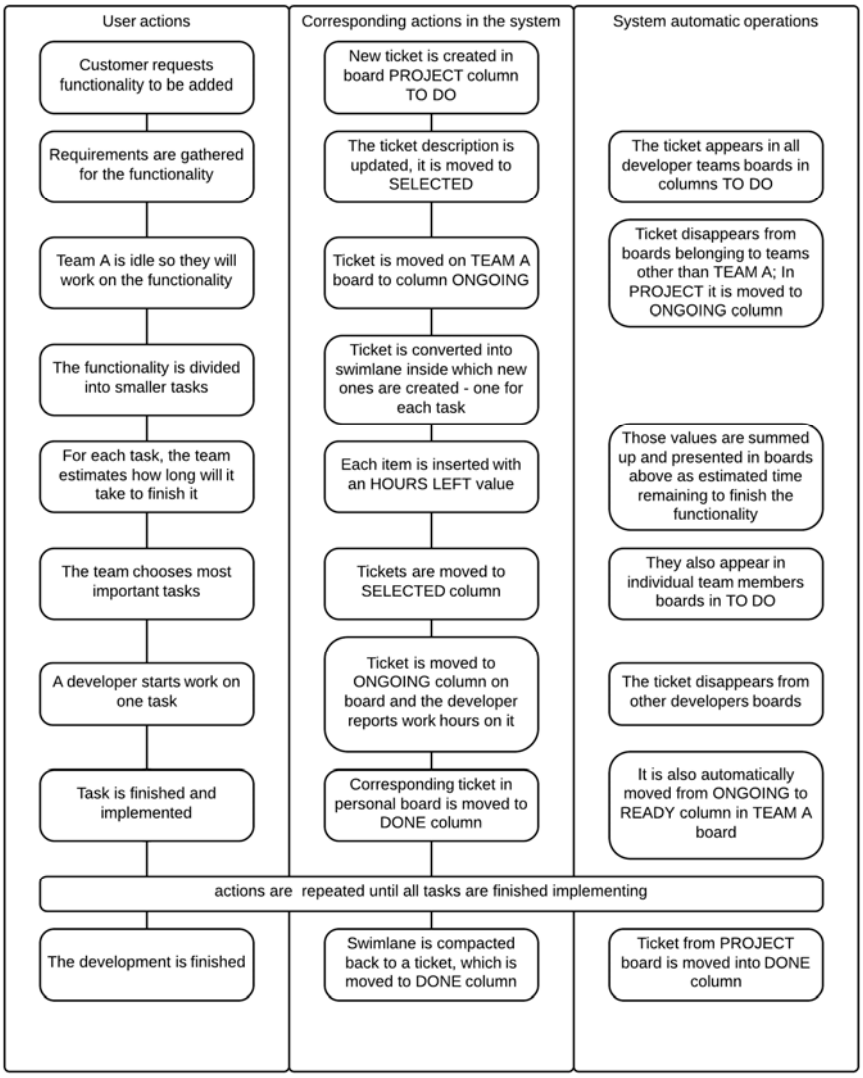


Fig. 3. Exemplary flow
Rys. 3. Przykładowy przebieg pracy

The boards presented in figures 4-7 depict an exemplary setup, consisting of four sample boards which are as following: project board, sample team board and two boards for employees. The project board is used for visualization of work on the level of whole functionalities.


Fig. 4. Project board
Rys. 4. Tablica projektu

The sample team board – where the team divides the functionality it is working on, into smaller tasks, and estimates how much time is required to finish each of them.

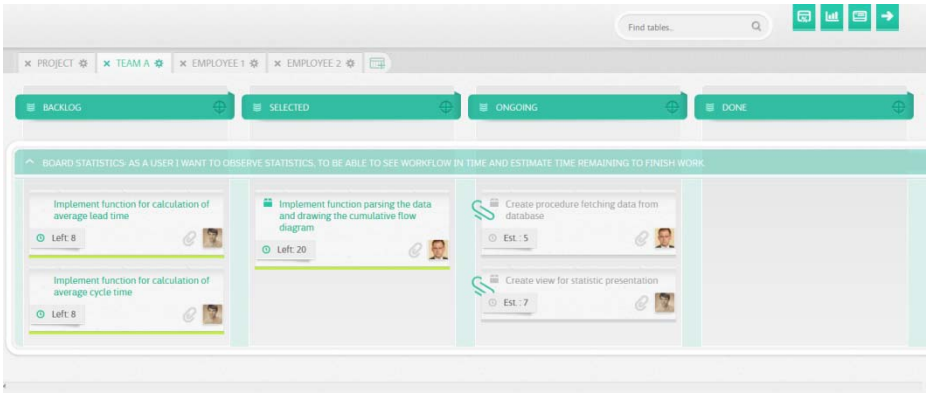


Fig. 5. Team board
Rys. 5. Tablica zespołu

These boards are connected with each other, so that e.g. when task in *PROJECT* board is moved onto the SELECTED column it is copied into the BACKLOG of the *TEAM A*. Additionally, when in *TEAM A* board, the task will be moved to the ONGOING column, it will also be moved in the *PROJECT* table, to indicate the state change. One can also benefit from the links in a following way. In the given example, the column SELECTED from *TEAM A* board is connected to column TO DO from both employee boards. This means, that each task that is being selected will be copied onto each of employees boards - so that when one of them logs in to the system, he will notice what tasks are waiting in the queue. However, when any of the employees moves the task away from the TO DO column, the task will disappear from every other employee board - to avoid issues, when two persons work separately on one task.

The last two boards are for employees – where employees manage the tasks they are working on.

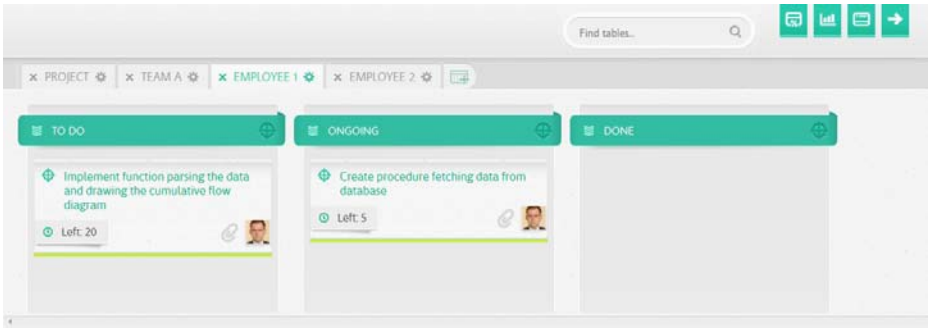


Fig. 6. Sample employee 1 board
Rys. 6. Przykładowa tablica pracownika nr 1

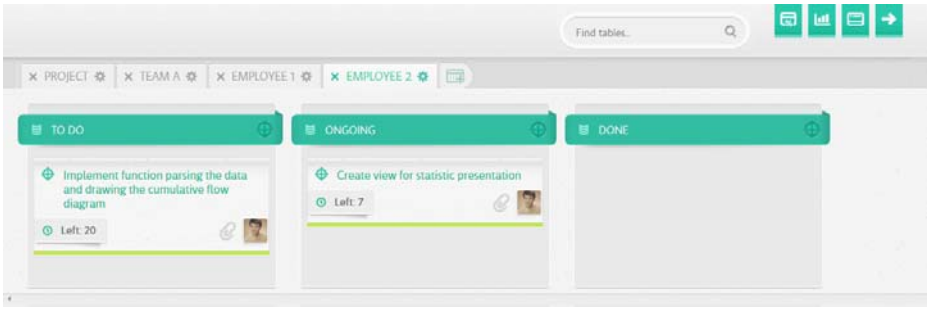


Fig. 7. Sample employee 2 board
Rys. 7. Przykładowa tablica pracownika nr 2

Another feature that can be observed, is that the estimated times, are propagated to other boards. When the estimation for a task changes, it will also automatically update the estimation for the functionality. Thus, the values on each board are up to date and can be used for planning purposes.

This flow example shows only small part of the development process. More complex examples might include additional boards for non-development teams or actors, such as for instance tester teams, business development or legal departments. In such a case, the board setup would be much more complex, with larger number of links and columns to model the real-life dependencies and communication patterns.

### 4.3. Statistics

In Fig. 8 statistics for a real-life board example are shown. One can analyze the visible data and extract important information. In the cumulative flow diagram one can see, that the orange region is increasingly growing with time. This region corresponds to *Approved* column. Tickets inside this column, has been successfully implemented and accepted, however they were not moved to Done - which means that they were not uploaded to the server. This indicates and area where the development process could possibly improve - deliver finished functionalities more frequently.

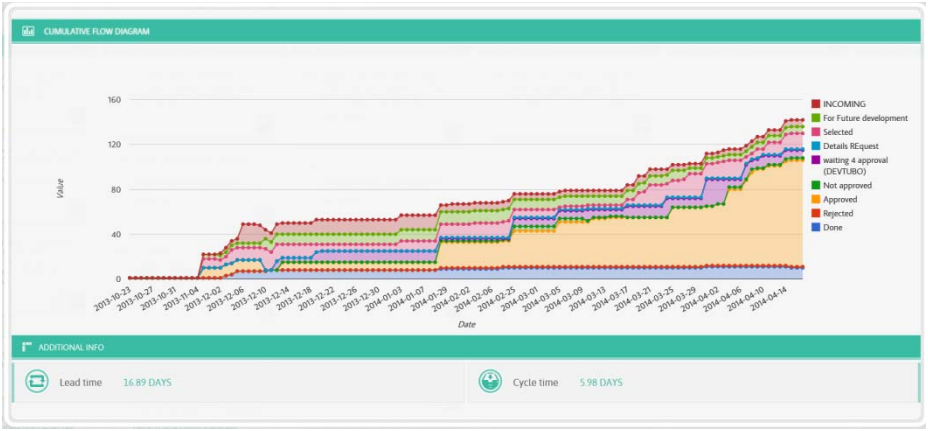


Fig. 8. Statistics for exemplary board
Rys. 8. Zestawienie statystyczne dla przykładowej tablicy

Apart from the diagram we can see also average lead and cycle time. These values can be used for development planning. The value for the lead time is approximately 17 days, which means that, on average, implementing new feature takes 17 days. This information could be used in a situation, when the customer needs to know, how much time will it take to implement some new top priority function. The other value informs us, how often the developers are able to deliver new parts to the system. In our case this is 6 days. Basing on this information, we can for instance plan meetings with customers every month, so that we should be able to present them 5 newly implemented features.

## 5. Conclusions

To conclude, new tool for visualization of workflow has been developed. The purpose of the application is to allow cooperation of multiple teams, distributed geographically, on single project with the use of agile methodologies. It also aids the organization and planning actions related to the development process.

The tool is being used in the Ericpol company, in real life scenarios. Over 200 of the company's employees registered to the system during its 5 month long uptime and it continues to gain active users. The system has achieved its main goal – it enables the cross-team cooperation, as well as task flow coordination on multiple levels of hierarchy in the development process. The application provides simple interface, while ensuring high utility. Placing the application logic on the database side, provides many possibilities for introduction of new interfaces to the system.

The project is under constant development, as new functionalities are implemented to the software. In the future, the project could be extended by integrating the application with

e-mail messaging systems, instant messengers or desktop gadgets, so that user will be notified when they receive new tasks.

**Acknowledgements:** We would like to thank the Ericpol company, for providing the necessary resources and opportunity to work on this topic. We would also like to thank our colleagues from Ericpol, with whom we are working on the project for their willingness to share their knowledge and experiences and great work atmosphere.

## BIBLIOGRAPHY

1. Anderson J. D.: Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press, 2010.
2. Bloch M., Blumberg S., Laartz J.: Delivering large-scale IT projects on time, on budget, and on value. McKinsey&Company, 2012.
3. Boeg J.: Priming Kanban: A 10 step guide to optimizing flow in your software delivery system. Trifork, 2nd Edition, 2012.
4. Brittain J., Darwin I.: Tomcat: The Definitive Guide O'ReillyMedia. 2nd Edition, 2007.
5. Cohen F.: Java Testing and Design: From Unit Testing to Automated Web Tests. Pearson Education, 2004.
6. Gajowniczek P.: Tool for visualization of multilevel task flow. Technical University of Lodz, Bachelor thesis, 2014.
7. Hernandez M.: Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design. 3rd Edition. Addison Wesley, 2013.
8. Kniberg H., Skarin M.: Kanban and Scrum – making the most of both. C4Media, 2010.
9. Oostvogels N.: Kanban for skeptics: Clear answers to Kanban in software development. Leanpub, 2012.
10. Schwartz B., Zaitsev P., Tkachenko V.: High Performance MySQL. 3rd Edition. O'ReillyMedia, 2013.
11. Tacy A., Hanson R., Essington J., Tokke A.: GWT in Action. 2nd Edition. Manning Publications, 2013.
12. Tymiński J., Zawiślak R.: Application of numerical procedures package in preparation of the optimal offer of companies participating in public action. Methods and tools of modern organization management, Vol. 4, 2008, p. 607÷618.
13. http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf.
14. http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf.
15. http://agilemanifesto.org.

**Omówienie**

W ostatniej dekadzie obserwuje się znaczny wzrost ilości oprogramowania wspomagającego organizację pracy zespołów informatycznych. Tworzenie szybko i efektywnie oprogramowania, które spełniałoby wzrastające wciąż wymagania klientów, nadal nie jest jednak prostym zadaniem. Głównym celem artykułu jest przedstawienie nowej aplikacji bazującej na koncepcji metodologii Agile-Kanban. Stworzona przez autorów aplikacja pozwala nie tylko na wizualizację aktualnego stanu zaawansowania nadzorowanych za jej pomocą projektów, ale również na śledzenie przepływu informacji pomiędzy wieloma współpracującymi zespołami, w ramach wielu realizowanych równolegle podprojektów (zadań). Wizualizacja poszczególnych etapów pracy wraz z ich przewidywanym czasem ukończenia (czasem dotychczasowej realizacji i ewentualnymi opóźnieniami) pozwala na lepsze planowanie i ewentualne przesuwanie zadań pomiędzy współpracującymi grupami. Autorzy przedstawili nowe podejście do znanej koncepcji metodologii Kanban. Stworzona aplikacja pozwala bowiem na równoległe realizowanie prac przez wiele zespołów, uwzględniając ich pełną autonomię, ale również wzajemne interakcje zachodzące pomiędzy wspólnymi fragmentami zadań. Autorzy omówili również testy aplikacji i perspektywy jej dalszego rozwoju. Rozdziały 1-3 stanowią obszerne wprowadzenie do tematyki artykułu. Rozdział 4 zatytułowany „Nowe podejście" przedstawia koncepcję i realizację samej aplikacji. Na podstawie przykładowego diagramu przepływu, przedstawionego na rysunku 3, zawierającego opis prostego problemu informatycznego autorzy utworzyli odpowiednią strukturę tablic, umożliwiających obserwację rozwoju prezentowanego zadania. Rysunki 4-7 są kolejnymi zrzutami ekranu ilustrującymi wygląd aplikacji w zależności od poziomu uprawnień użytkownika. Artykuł kończy podsumowanie oraz prezentacja przykładowego zestawienia statystycznego generowanego automatycznie przez aplikację.

**Addresses**

Piotr GAJOWNICZEK: Lodz University of Technology, Institute of Automatic Control, Stefanowskiego 18, 90-924 Lodz, Poland, piotrgajow@gmail.com
Rafał ZAWIŚLAK: Lodz University of Technology, Institute of Automatic Control, Stefanowskiego 18, 90-924 Lodz, Poland, rafal.zawislak@p.lodz.pl.