Agile Micro-Methodology for the Development of Software in Programming Marathons (Hackathons)

L Guillermo CANTO-SUSTAITA
Héctor G PEREZ-GONZALEZ
Alberto S NUÑEZ-VARELA
Francisco E MARTINEZ-PEREZ
Department of Computer Science, Universidad Autónoma de San Luis Potosí,
San Luis Potosí, S.L.P, México

ABSTRACT

This paper describes our experiences using and evaluating a methodology for the development of applications in extremely short periods of time, such as programming in Hackathons. A hackathon is by definition, a programming event that includes a component of competition, however, experience following some methodology in such extreme situations could contribute to the improvement of software processes when it requires extremely short production times. A semi-formal qualitative study of the use of the methodology was conducted, in order to evaluate the success of the process, with a view to improving future similar events. A real Hackathon event was used to evaluate mentioned methodology and gather experience about the same.

Keywords: Hackathon, Software Development Methodology, Agile, Scrum, Kanban.

1. INTRODUCTION

In the software development field, we find certain scenarios in which the time required to generate a (software) product represents one of the most important restrictions to achieve it. This refers to situations in which it is required in extremely short periods of time to add a new feature, or to make a correction or update. This situation forces the development team to breach the guidelines of a conventional methodology to solve the problem.
A specific example of such development work is presented at events known as programming marathons or Hackathons in which a self-managed development team [10] must obtain a product in a short period of time (72, 48 or even less hours).
These types of scenarios present a great challenge for conventional methodologies and even for well-known agile methodologies. Such is the case of Scrum [3], in which process iterations (Sprints) [4] with an average duration of 15 days are recommended. A hackathon is by definition, an event that includes a component of competition, however, experience following some methodology in such extreme situations could contribute to the improvement of software processes when it requires extremely short production times. This type of situation arises in normal software industry for example, in the generation of security patches or expedited corrections, the production of additional features resulting from changes in requirements or urgent business needs, or the inclusion of functionalities such as incremental improvement or solution of problems that affect users in real time.
The proposed research question is:

“What are the benefits of implementing an extremely fast development methodology for hackathons?”

The objective of this work is to propose and evaluate the use of a "Micro-methodology" for the development of applications in extremely short periods of time, such as programming Hackathons. Section 2 presents an overview of conventional and agile software process models. Section 3 describes the hackathons. Section 4 describes the proposed micro-methodology and its application in a particular case. Section 5 shows the results of applying the evaluation queries on the use of the micro-methodology. The article concludes with the discussion of the results, conclusions and future work.

2. SOFTWARE DEVELOPMENT METHODOLOGIES

A software development model or methodology (SDM) is the abstract representation of a software process seeing from a particular point of view [5]. In the same way an SDM commonly reflects the evolution of the process and the product (software) trough the time according the peculiarities of the model. Furthermore, an SDM defines a framework for the core activities of the project, the inputs and outputs, constraints and the role of each one of the persons involved in the process [1].
The principal models that have influenced the definition and design of the proposed methodology are: Rapid Application Development (RAD) and Scrum + Kanban (Scrumban).

Rapid Application Development (RAD)
The key feature of this model is the rapid delivery of parts or phases of the product to end users, in a way they can test and suggest improvement according to the real needs and experience; this kind of evaluation make the processes to be known as test driven development [2]. Prototypes and code-generation software, like Gradle or Visual studio, are strongly used in these models so that sometimes these methodologies are also called Prototyped and Software assisted processes. These tools provide templates and full frameworks that allow an efficient incremental development of the product.

Agile, Scrumb and Kanban
Agile methodologies are focused in getting a dynamic development life cycle, seeing the development process more as an iterative process than an incremental one. The agile basis consists of the collaboration of self-managed and multidisciplinary teams, involved in a shared short-term decision-making process [12]. The core of the methodologies is based on structured iterations that seek to deliver well-defined functionality.

Scrum is possibly the best-known agile methodology today. It is an iterative development model, with regular sprints every 2 to 4 weeks, with goals or features prioritized in a product stack

(backlog). At the end of each iteration, a partial delivery usable by the customer is produced [5].

Kanban is a relatively new concept in the field of software engineering; it was originally applied in Lean Manufacturing at the Toyota production lines [6]. Kanban provides a flexible workflow for teams and an easy traceable progress control, limiting the work in progress (WIP) for each activity to a maximum number of tasks or items at any given time. It provides a clear display of the phases in the project using visual dashboards. It is common to find Kanban foundations applied in a variety of Project Management technologies which can be successfully integrated with Agile Development process [16].

Scrumban. Recently, proposals have emerged with the idea to integrate the similarities of different methodologies, as well as strategies to mitigate their failures. One example of this is the pair Scrum / Kanban which allow the two methods to be combined by implementing some of the Scrum practices and Kanban principles. Such a combination is known as Scrumban [15], which takes advantages of the inherited capability of scrum to be an iterative process divided in small tasks and the clear and solid workflow visualization of Kanban.

3. PROGRAMMING MARATHONS (HACKATHONS)

A hackathon, also known as a codefest, is a programming or coding event that brings together computer programmers and other interested people to improve or create a new software program. The word hackathon is a combination of the words hacker (intelligent programmer and problem solver) and marathon (event characterized by the resistance of its participants). The duration of these events usually varies between 72, 48 or even less hours [11]. Currently, the use of Hackathons has become popular for various purposes, whether for learning or teaching [7], to encourage the use of new technologies [9] or to promote new products or innovative ideas within a company [8]. According to the codeslaw, [14], a highly influential website in the software industry, the most important hackathons today are:

1) HackZurich,
2) TechCruch Disrupt and
3) hackNY.

HackZurich is the largest hackathon in Europe. This is an annual non-stop coding competition held in Switzerland over a period of 40 hours.
TechCruch Disrupt is a collection of events that take place in New York City, San Francisco and Berlin. It hosts a 24-hour hackathon, which gives coders the opportunity to create new products, interact with industry leaders, and connect with investors.
hackNY is a New York City 24-hour hackathon for student programmers from all universities, settings, and skill levels. It takes place at New York University, or at Columbia University.

As observed, an especially important objective of hackathons in this context is to become a link between academia and the industry so that the latter finds new employees to fill their coder needs. Because of this, it is important for students to have the skills and abilities to best accomplish the intended goal.
According to related literature, there is no specific methodology established for software development within the context of a hackathon.

4. AGILE MICRO-METHODOLOGY FOR THE DEVELOPMENT OF SOFTWARE IN HACKATHONS

This work proposes a short-term development methodology that contributes to the realization of quality software. The methodology is mainly based on the principles of Agile models, such as Scrumban. We decided to take as foundations the agile values: *people over process* and *software over documentation*; and the principles: *software as measure of progress* and *leverage self-organized teams* [12].

In addition to this, other principles like prototype usage from ot RAD model are used to enrich the methodology. Additionally, the generation of simple and frequent design documents is proposed as modeling exercises.

The proposed methodology is based on the following principles:

1) Generation of concept or initial high-level design and establishment of vision and design document.
2) Definition of high-level initial tasks organized in iterations; this is a micro-backlog (similar to Scrum).
3) Use of task tables (Kanban methodology)
4) Use of frequent prototypes throughout the hackathon to encourage rapid proof of concepts and usability.

As shown in Fig. 1, in order to achieve these principles, the methodology proposes three main hackathon phases:

1) Design
2) Creation and execution and
3) Closure.

Every software system should ideally be the solution to a problem. In a regular development process, the customer is who explains this problem. However, in the case of a hackathon, the precision of the problem explanation reaches exceptionally low dimensions. This is because the organizers regularly raise a topic and participants are asked to identify a possible problem within the scope of that topic and propose a solution. In this situation, developer skills such as creativity, ingenious and imagination take on great relevance. The three phases of the proposed methodology focus on the search for the enhancement of these skills.

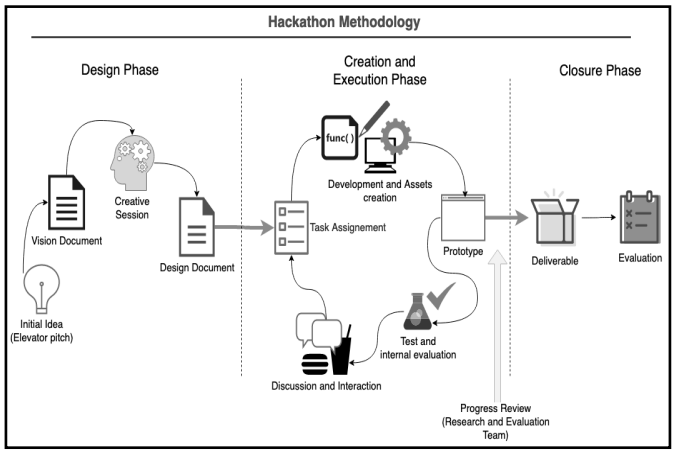

Figure.1 Hackathon Software Development Micro-methodology

Design phase
The design phase begins with the instrumentation of a scope-shaping technique known as Elevator Pitch. [13]. Each team of participants (from 3 to 5 members) presents the identified problem and its proposed solution. This exposition is done to 2 or 3 evaluators inside a real elevator that goes up several floors (7-10) without stopping on intermediate floors. The exposure period is approximately half a minute. The evaluators take note and do not ask the participants questions.
Once this step is completed, the participants are taken to the hackathon headquarters to generate two highly relevant deliverables: The Vision and the Design document. The teams must spend the first hours to establish some of the points presented in the vision document. It is in this way that we seek to help the participants to concretize ideas and to define the style and concrete objectives of the team. Within the first segment (12 to 24 hours), the team seeks to materialize their ideas by carrying out design work and capturing on paper and computer the first sketches of planning and design.

Creation and Execution Phase
The team displays their tasks on a visual board, they are assigned to each member and the times are established for each task. The team constantly updates its task board and if necessary, add more or delete some. A review by the evaluation and approximate research team is made at each completion of iterations. Every 4-5 hours for a 32-hour hackathon. The team is free to display artifacts, prototypes, diagrams, or any progress generated.

Closing phase
The closing phase consists of the delivery of results and evaluation. This stage also shows very different characteristics from the delivery phase of an industrial software product. How the results are presented can be as important as the results themselves. Participants' communication skills take on special importance.
Delivery of physical and digital prototypes is important. Upon completion of the hackathon the team is responsible for delivering final board and digital artifacts.
To carry out the evaluation and consequent decision of the winners of the hackathon, they are considered with different weightings, the exposure in the elevator pitch, the vision and design documents, the planning and execution boards, the physical and paper prototypes and the verbal exposure of results.
The use of the proposed micro-methodology in a particular case is described below.

Application of the Methodology
The Autonomous University of San Luis Potosí, Mexico has the Computer Engineering academic program classified as the best Computer Engineering degree in the country according to the National Assessment Center (CENEVAL).
This institution has the experience of having organized three hackathons in which two of them have applied the proposed methodology.
The most recent hackathon "HackSLP" (Dec 06-07, 2019), with more than 150 students and professionals, implemented this methodology with the results presented in the following section. Below are the artifacts that were considered core deliverables in that edition.

1) Vision and Idea Document - Idea using the key words. The document is presented in the first 6 hours and is saved as historical. It was used as an evaluation factor. Content:
• Team vision
• Initial requirements of the contest (keywords)
• "Elevator pitch" paragraph
• Complete idea description
• General Objectives of the Project / Idea

2) Design Document- First sketch of the Idea using the key words. A transcript of the elevator speech dynamics is included. Content:
- Overview of the project
- Category of the software or problem to solve
- Brief Functionality Description
- Nuclear functionalities or features
- General rules and constraints
- Main interactions
- Technologies and platforms to use
- Development plan, initial tasks (high level)
- Team roles

3) Initial Backlog - It is not a document as such. It can be an initial board or high-level tasks presented in a design document.
- High-level tasks according to the needs and planning of the team.
- Classification of tasks into categories (buckets) according to roles or activities.

4) Kanban board. It can be a paper document or digital representation; in each session it is updated (iterations). Includes at least the following lanes.
- to do
- doing
- Completed

5) Paper prototypes and design diagrams
• Any draft, sketch, in any medium (Physical / Digital)

Evaluation
Below is the list of essential physical deliverables
- Completeness level
- Usability reports (by experts)
- Functionality - no runtime / logic / broken mechanics errors
- Documents presented (best development process)
- User rating or evaluation (real tests on prototypes) (tests with end users)

The hackathon was executed using the 3 phases mentioned previously. All participants attended to a methodology introduction talk and received a methodology kit (documents templates, tasks boards examples, information about Kanban board soft, etc.)
At the end of the first day (Dec 6), the Vision and Design document were delivered by all teams. These are the only documents required for the entire event since the value of "people/software over documentation" is fundamental.
From the first hours of the second day and until the end of the event, the progress of the teams was reviewed using prototypes Kanban board of each one of them.

5. EVALUATION AND RESULTS

In this particular hackathon, only 40% of the teams delivered a complete product with the requested characteristics (keywords and base requirements); Only 20% of the teams delivered a product that met the evaluation parameters of the expert panel.
Some of the characteristic details of the winning team's working model were the following:

- Clear definition of Vision and Design Document
- Clear and systematic task management using the Kanban Flow software.
- Prototype creation and evaluation

We got feedback from the teams, which deliver the expected product, asking about the overall performance in the event and the phase and artifact they found more valuable; all teams agreed on the impact of the Task Management and the early creation of a Design document.

Supporting this idea, we found an interesting fact: All participating teams delivered the design documents in the estimated time and with the expected content.

In order to acquire deeper feedback and sustainable information, we designed the following set of questions, which were shared as a survey with all the participants (Table 1):

Table 1. Questions (translated from Spanish) designed for artifacts and phase evaluation.

QUESTION	PHASE TO EVALUATE
Being 4 maximum value, how useful do you think it was the "Design Day"	Design Phase
With your words, tell us what was do you think about the "Design Day"	Design Phase
Being 4 maximum value, how clear do you think the "Design Document" was	Design documents
Being 4 maximum value, how useful do you think it was to create a "Design Document"	Design documents
Being 4 maximum value, how useful do you think it was to have "task assignment and management" in the process	Task and iteration management
Being 4 maximum value, how easy do you think it was to manage tasks with "Kanban Flow" software	Task and iteration management
Being 4 maximum value, how would you grade the "24hrs of Development" phase	Execution Phase
With your words, tell us what was do you think about the "24hrs of Development" phase	Execution Phase
Please share with us which do you think was your team main limitation during the challenge.	General Performance

Of the surveyed participants, 83% think that the Design day is really useful. The majority of the participants referred the "Design Day" as something needed and a useful for the general development. This aligns with the feedback provided by the winner teams previously mentioned.
In the other hand, 58% considered that having a task management was very useful, 42% considered it useful. Similarly, the impact of the tools used during the process is considered important.
The result is interesting since 99 % considered the KanbanFlow software as practical, 0.8% considered the software as impractical.
Additionally, 43% of the users believe that the more exhausting phase of the event was the 24 hrs. development (programming) period.
In the same way, 42% of the participants mentioned the main limitation for the team was lack of technical knowledge.

Finally, we were able to interview 12 participants who were part of the 2 previous hackathons hosted by the UASLP. We asked for their inputs about most notorious differences and improvements with respect the previous events. 75% of the interviewed participants agreed in the following points:

- Having a well-structured design day helped them to get more advantage of the execution phase.
- Early definition of tasks and having a task-board truly increased team performance.
- As an improvement deeper introduction talk should be implemented to share benefits of the methodology at the beginning of the event and not just provide a brief chat.

6. CONCLUSIONS AND FUTURE WORK

Our research question is: "What are the benefits of implementing an extremely fast development methodology for hackathons?" According to the study carried out, we can conclude with the following list of possible benefits:

- Development of micro-iterable addressable products is beneficial to reduce surprises.
- Explicit dedication of a number of hours to specific tasks adds visibility to the process.
- The focus in the design stage increases the probability of success of the project and the tranquility of the stakeholders.
- The use of software tools for works coordination improves the process.
- Capability of process control when a very small time of deliver is a requirement.

An interesting subject to develop in the future, would be the deeper analysis of the Kanban boards and task management, seeking to strengthen the results already presented and to find a relationship between the deliverable cycle and task management. In the same way, it would be very informative to study the relationship between the dispersion of ideas and the progression of the deliverable.

Finally, this work could be a foundational piece for a bigger and more focalized work using future hackathon events in which the scalability of the methodology could be evaluated.

9. REFERENCES

[1] Peters, J. F., & Pedrycz, W. **Software engineering: An engineering approach.** New York: John Wiley. 2000.

[2] Martin, James "**Rapid Application Development**" MacMillan Publishing Co., ed. 1990.

[3] K Schwaber, J Sutherland - **The scrum guide**, Scrum Alliance, 2011

[4] Schwaber K. **SCRUM Development Process.** In:Business Object Design and Implementation.Springer, London 1997

[5] Barrientos Acosta, Juan Manuel "**Aprendizaje de la Ingeniería de Software utilizando Simuladores basados en Sistemas Multiagente**". UASLP 2010.

[6] Ahmad, M.O., Markkula, J., Oivo, M. **Kanban in software development: A systematic literature review**. In 39th EUROMICRO Conference on Software Engineeringand Advanced Applications (SEAA), pp. 9–16. IEEE Press 2013

[7] C. Guerrero, M. del Mar Leza, Y. González and A. Jaume-i-Capó, "**Analysis of the results of a hackathon in the context of service-learning involving students and professionals"**

International Symposium on Computers in Education (SIIE), Salamanca, 2016, pp. 1-6. 2016

[8] S. Saravi et al., "**A Systems Engineering Hackathon – A Methodology Involving Multiple Stakeholders to Progr ess Conceptual Design of a Complex Engineered Produc**t, " in IEEE Access, vol. 6, pp. 38399-38410, 2018.

[9] Serrano-Laguna Á., Rotaru DC., Calvo-Morata A., Torrente J., Fernández-Manjón B. **Creating Interactive Content in Android Devices: The Mokap Hackaton.** In:User Development. IS-EUD 2015. Lecture Notes in Computer Science, vol 9083. Springer, Cham 2015

[10] M. Komssi, D. Pichlis, M. Raatikainen, K. Kindström and J. Järvinen, "**What are Hackathons for?,"** in IEEE Software, vol. 32, no. 5, pp. 60-67, Sept.-Oct. 2015.

[11] Arnab Nandi and Meris Mandernach. **Hackathons as an Informal Learning Platform.** In Proceedings of the 47th ACM Technical Symposium on Computing Science Educati on (SIGCSE '16). ACM, New York, NY, USA, 346-351. 2016

[12] **Agile Manifesto**, official site https://agilemanifesto.org/ , last seen July 2020

[13] R. F. Ciriello, A. Richter and G. Schwabe, "**When Prototyping Meets Storytelling: Practices and Malpractices in Innovating Software Firms**," IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP), Buenos Aires, 2017, pp. 163-172. 2017

[14] Codeslaw Official Site, "**The 5 Best Hackathons Programmers Should Attend**" https://codeslaw.com/insight/the-5-best-hackathons-programmers-should-attend-943185

[15] Banijamali A., Dawadi R., Ahmad M.O., Similä J., Oivo M., Liukkunen K. **Empirical Investigation of Scrumban in Global Software Development**. MODELSWARD 2016. Communications in Computer and Information Science, vol 692. Springer 2017

[16] D. Parsons, R. Thorn, M. Inkila and K. MacCallum, "**Using Trello to Support Agile and Lean Learning with Scrum and Kanban in Teacher Professional Development,"** 2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), Wollongong, NSW, 2018, pp. 720-724.