Energy Efficient Software Development Process Evaluation for MacOS Devices

Shokhista Ergasheva(✉), Dragos Strugar, Artem Kruglov, and Giancarlo Succi

Innopolis University, Innopolis, Russia
s.ergasheva@innopolis.ru

Abstract. InnoMetrics is the system that aims at collecting software development process metrics in a non-invasive way to access and optimize the development process and its efficiency. This paper demonstrates the development and analysis of energy consumption of MacOS systems based on the software process measurement data. It represents the experience of the development of MacOS system collector and Transfer, in addition to the user interface and early analysis of energy consumption metrics calculations.

Keywords: Energy efficient software · MacOS software development · Software development process metrics

1 Introduction

The importance of energy efficiency of any system and devices becomes sufficient property of the present Information Technology life. Thus, measuring and what is more further analysis of energy efficiency and its prediction can be managed by monitoring the software development process. Before going deep into the software development process metrics let's define what is the measurement itself. The history of measurement can be divided into two generations. The first generation is the Personal Software Process (PSP) - self improvement process that helps developers to control, manage and improve the way they work. This method can be called as invasive as it requires the direct involvement of participants in the data collection process. Users of the-invasive method should create and print forms in which they log their effort, size and defect information. One obvious downside of this invasive approach is the high overhead cost it entails. The developers should often switch between development tasks and metrics collection tasks, which imposes a high cognitive burden to the developers while the second generation of measurements, non-invasive measurements, do not require manual intervention of the participants during metrics collection [4]. The main aim of the successful use of a non-invasive measurement system is to support the

This research project is carried out under the support of the Russian Science Foundation Grant № 19-19-00623.

V. Ivanov et al. (Eds.): OSS 2020, IFIP AICT 582, pp. 196–206, 2020.
https://doi.org/10.1007/978-3-030-47240-5_20

developers, the development process optimization and maintain the data privacy. Innometrics is one of the promising software development process data collection systems with automatic data collection and transfers to the server for further data analysis. It allows the developers and managers to be aware of the process strength and weaknesses based on the data from the developers performed. The crucial insights are visualized concerning the metrics and analysis. The benefits of such system can be counted as real-time process analysis on a daily basis, the system can be accustomed to different levels of company sizes and audits of the software development process and development itself can be monitored at the same time.

2 Metrics

The first thing in the way to data collection framework was to perform a Systematic Literature Review (SLR) of existing research papers, energy-related metrics and best practices to collect data during the software development process. The Systematic Literature Review surveyed more than 500 studies in the field of software metrics collection and analysis. As a result, at about 170 metrics were derived, and divided into 3 categories as code, product and process metrics. Based on the findings from SLR, we concluded that the mostly unexplored branch of software development metrics is process metrics [8]. Besides, all the tools observed during the research project consider static analysis of the code, nevertheless, the process of developing the software still has no much analysis. The main reason for this insufficient analysis of process measurements is the absence of the tool. Furthermore, the process cost increases, since usually, it requires the developers' participation.

The system of InnoMetrics is developed based on the monitoring of the software development process energy consumption and its efficiency, the developer's productivity. All direct and model-based measurements that involve usage of third-party hardware tools to get energy metrics from various components were considered out of scope of non-invasive software development process analysis approach. The energy consumption of software applications was thoroughly researched and concluded based on the research findings. Battery draining applications result in lower user experience and dissatisfied users [18]. Optimal battery usage (energy usage) is an important aspect that every client must consider.

Application energy consumption is dependent on a wide variety of system resources and conditions. Energy consumption depends on, but is not limited to, the processor, the device uses, memory architecture, the storage technologies used, the display technology used, the size of the display, the network interface that you are connected to, active sensors, and various conditions like the signal strength used for data transfer, user settings like screen brightness levels, and many more user and system settings [18].

For precise energy consumption measurements, one needs specialized hardware [21]. While they provide the best method to accurately measure energy consumption on a particular device, such a methodology is not scalable in practice, especially if such measurements have to be made on multiple devices.

Even then, the measurements by themselves will not provide much insight into how the application contributes to the battery drainage, making it hard to focus on any application optimization efforts.

The InnoMetrics system aims at enabling users to estimate their application's energy consumption without the need for specialized hardware. Such estimation is made possible using a software power model that has been trained on a reference device representative of the low powered devices applications might run on [15,17,19,20,22]. Based on the findings of the research, metrics like following were investigated [10]:

– Software Energy Consumption (SEC) - the total energy consumed by the software;
– Unit Energy Consumption (UEC) - the energy consumed by a specific unit of the software;

Considering our profiling method and the tools available for us, the ability to attribute the energy consumption was possible only at the process level in coarse granularity. However, the hardware resource usage can fill the gap when it comes to accurately relating Energy Consumption (EC) to individual software elements hence enabling the computation of the UEC.

Profiling the performance requires a basic understanding of hardware components that has to be monitored through "performance counters", which is possible in Windows System. While interpreting performance data for further analysis, the context information has to be taken into account (e.g. hardware-specific details).

To evaluate the Unit Energy Consumption (UEC) the following hardware resources should be monitored:

– Hard disk: disk bytes/sec, disk read bytes/sec, disk write bytes/sec;
– Processor: percentage of processor usage;
– Memory: private bytes, working set, private working set;
– Network: bytes total/sec, bytes sent/sec, bytes received/sec;
– IO: IO data (bytes/sec), IO read (bytes/sec), IO write (bytes/sec).

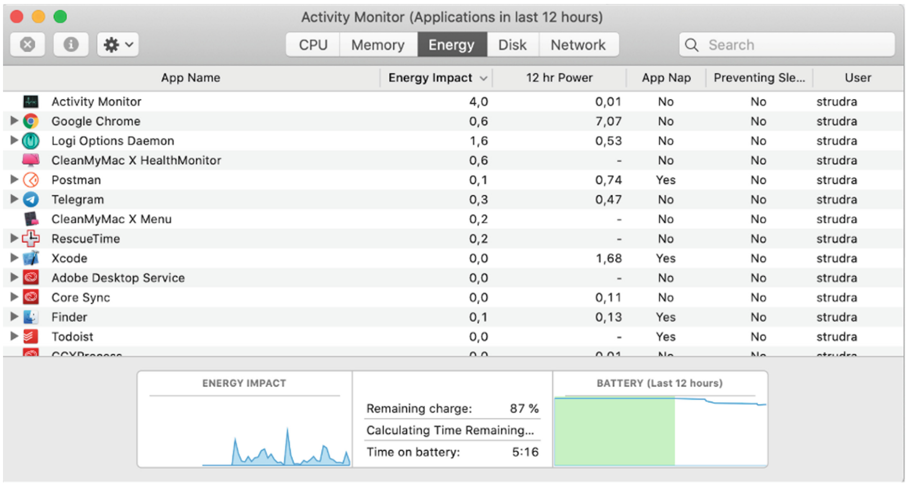
Attributing some weights to elements of the UEC or by some reliable assumption such as considering the power model to be linear in the nature for each individual component, the SEC Metric is computed.

Besides, the energy usage can also be appraised using Performance Counter, Performance Counter Category and related classes that are available with .NET Framework. To be specific, by analyzing the MSDN documentation to attain the goal of collecting energy related metrics, it was concluded that the information about CPU time, Total Processor Time per process, CPU usage, Memory usage, network usage that Performance Counter provides can be reliable.

The bottleneck in this situation is that it is difficult to match up constantly changing application process IDs and names. The energy consumption of the system depends on a variety of factors that are not limited to those which can be collected using the above mentioned performance classes.

2.1 MacOS Energy Metrics

In order to obtain energy-related data in MacOS devices, we explored some of the APIs that MacOS provides. The first and most obvious tool was Activity Monitor, an application that comes built-in every MacOS system. One of its aspects is the Energy consumption, shown in Fig. 1.



App Name	Energy Impact	12 hr Power	App Nap	Preventing Sle...	User
Activity Monitor	4,0	0,01	No	No	strudra
Google Chrome	0,6	7,07	No	No	strudra
Logi Options Daemon	1,6	0,53	No	No	strudra
CleanMyMac X HealthMonitor	0,6	-	No	No	strudra
Postman	0,1	0,74	Yes	No	strudra
Telegram	0,3	0,47	No	No	strudra
CleanMyMac X Menu	0,2	-	No	No	strudra
RescueTime	0,2	-	No	No	strudra
Xcode	0,0	1,68	Yes	No	strudra
Adobe Desktop Service	0,0	-	No	No	strudra
Core Sync	0,0	0,11	No	No	strudra
Finder	0,1	0,13	Yes	No	strudra
Todoist	0,0	-	Yes	No	strudra

Remaining charge:	87 %
Calculating Time Remaining...	
Time on battery:	5:16

Fig. 1. MacOS energy metrics

The second column, Energy Impact, attracted our attention immediately. We then wanted to figure out what these values (4.0, 0.6, 1.6, etc) represent. It has been then brought to our attention that the definition of Energy Impact is not precisely defined by Apple. According to Activity Monitor's documentation, the definition of Energy Impact says "Energy Impact: A relative measure of the current energy consumption of the app. Lower numbers are better" [3].

As other documentation says "The Energy tab of Activity Monitor displays the Energy Impact of each open app based on a number of factors including CPU usage, network traffic, disk activity and more. The higher the number, the more impact an app has on battery power" [2]. Both of these are vague, and we needed a concrete way of obtaining this metric's values.

One article on the Mozilla blog attempted to figure out a formula for this. They indicated that the result of MacOS's top command line tool which performs periodic measurements of all kinds of metrics; including ones relevant to energy consumption: CPU usage, wakeups and the power measure. The article we mentioned above suggests running the following command to get the above-mentioned information:

Yielding the results (trimmed):

```
top -stats pid, command, cpu, idlew, power -o power -d
```

Table 1. *Top* results

PID	Command	% CPU	IDLEW	Power
50300	Firefox	12.9	278	26.6
76256	Plugin-container	3.4	159	11.3
151	Coreaudiod	0.9	68	4.3
76505	Top	1.5	1	1.6
76354	Activity Monitor	1.0	0	1.0

They go on to suggest that POWER measure is calculated using a simple formula, and a specific configuration file (tuned for every machine's architecture). Using the findings from that article, we decided to use some of the most impacting metrics (Table 1):

– Battery percentage
– Battery status (is charging or not)
– RAM measurement (how much RAM does the active process use)
– vRAM measurement (how much vRAM does the active process occupy)
– % CPU utilized (per process)

All of these metrics were obtained using the macOS command line interface. E.g. to get the current battery percentage we used pmset -g batt, and for other measurements we used the ps -axm -o command with varying parameters (depending on the use case). It was also possible to use the top command, but as we are performing periodic checks anyway, top was not necessary. Incorporating these metrics and collection process to the InnometricsCollector was successful and results are available on GitHub:

https://github.com/InnopolisUniversity/innometrics-agent-os-mac/.

3 Collector Development

InnoMetrics system contains of the following components [14,23,26]:

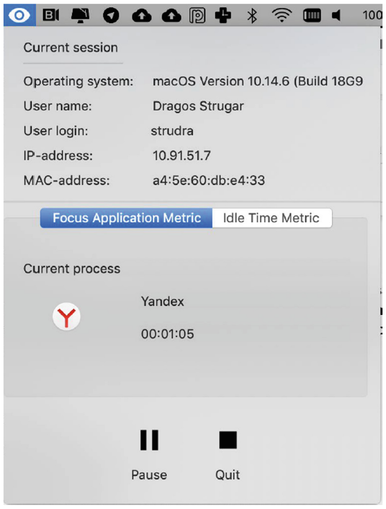
– Data Collectors - the separate for widespread Operating Systems (Windows, MacOS, Linux) data collecting frameworks;
– Server - which includes analytic module of obtained data referring quantitative and qualitative analysis;
– Dashboard - the component that responsible for visual representation of the analyzed development process data.

The scope of the paper do not consider describing the data collectors for all operating systems, nevertheless MacOS collector will be described thoroughly. MacOS collector includes two separate applications using a service oriented approach [25]:

– InnoMetrics Collector
– InnoMetrics Transfer

InnoMetrics Collector is a daemon process, it runs in the background. The only interaction the user has with the application is that the collector sits in the background and collects data. It displays the user information (Fig. 2):

– User Name;
– System version and specifications;
– IP address;
– Mac address of the device;
– as well as the currently open application.



Current session

Operating system: macOS Version 10.14.6 (Build 18G9
User name: Dragos Strugar
User login: strudra
IP-address: 10.91.51.7
MAC-address: a4:5e:60:db:e4:33

Current process

Yandex
00:01:05

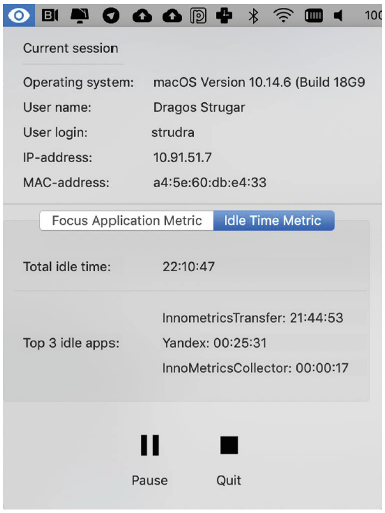
Fig. 2. MacOS Data Collector: Current Process

The Idle Time tab (Fig. 3) shows the total idle time and the top 3 application that have been idle for the longest. In addition, the user can stop the process of collection, as well as they can quit the app.

3.1 Metrics Being Collected

The data being collected on a per-process basis consists the following types:

– Metric type (app focus/idle);
– Application Name;
– Bundle Domain of the app (e.g.com.apple.finder);
– Bundle Path (where the app is installed, e.g. Applications/Yandex)
– Timestamp of the opening of the app/process;
– Timestamp of the quitting of the app/process;
– Duration of the process open (timestamp end-timestamp start);



Current session

Operating system:	macOS Version 10.14.6 (Build 18G9
User name:	Dragos Strugar
User login:	strudra
IP-address:	10.91.51.7
MAC-address:	a4:5e:60:db:e4:33

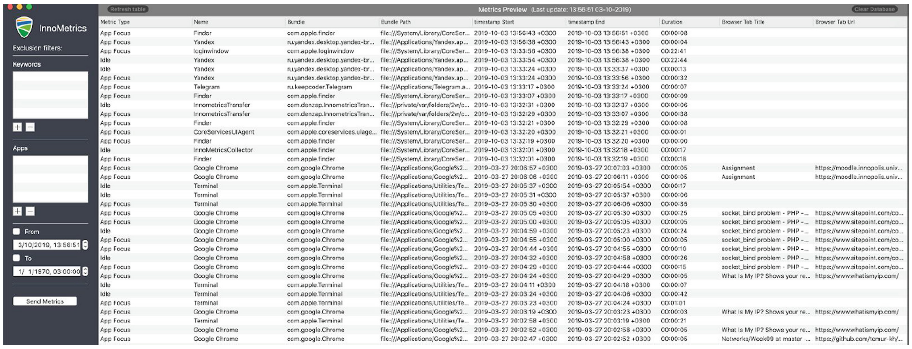
Total idle time:	22:10:47
Top 3 idle apps:	InnometricsTransfer: 21:44:53 Yandex: 00:25:31 InnoMetricsCollector: 00:00:17

Fig. 3. MacOS Data Collector: Idle Process

– Tab Name (for browser);
– Tab URL (for browser).

After collecting the metrics, the system saves the current session with detailed information (user name, login name, IP address, mac address, etc) only if it has not been changed. Then it saves the data to a local database, allowing the other app (MacOS InnoMetrics Transfer App) to send the data to the server.

Regarding the Innometrics Transfer app, shown in the Fig. 4, it is a separate application. However, it is not a daemon, it is a GUI application which allows users to see all the metrics that have been collected in addition to the functionality of sending such data to the back end.

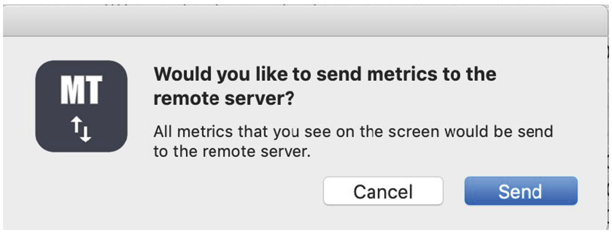


Metrics Preview (Last update: 13:56:51 03-10-2019)

Metric Type	Name	Bundle	Bundle Path	timestamp Start	timestamp End	Duration	Browser Tab Title	Browser Tab Url
App Focus	Finder	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:56:43 +0300	2019-10-03 13:56:51 +0300	00:00:08		
App Focus	Yandex	ru.yandex.desktop.yandex-br...	file:///Applications/Yandex.ap...	2019-10-03 13:56:38 +0300	2019-10-03 13:56:43 +0300	00:00:04		
App Focus	loginwindow	com.apple.loginwindow	file:///System/Library/CoreSer...	2019-10-03 13:33:56 +0300	2019-10-03 13:56:38 +0300	00:22:41		
Idle	Yandex	ru.yandex.desktop.yandex-br...	file:///Applications/Yandex.ap...	2019-10-03 13:33:54 +0300	2019-10-03 13:56:38 +0300	00:22:44		
Idle	Yandex	ru.yandex.desktop.yandex-br...	file:///Applications/Yandex.ap...	2019-10-03 13:33:24 +0300	2019-10-03 13:33:37 +0300	00:00:13		
App Focus	Yandex	ru.yandex.desktop.yandex-br...	file:///Applications/Yandex.ap...	2019-10-03 13:33:24 +0300	2019-10-03 13:33:56 +0300	00:00:32		
App Focus	Telegram	ru.keepcoder.Telegram	file:///Applications/Telegram.a...	2019-10-03 13:33:17 +0300	2019-10-03 13:33:24 +0300	00:00:07		
App Focus	Finder	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:33:07 +0300	2019-10-03 13:33:17 +0300	00:00:09		
Idle	InnometricsTransfer	com.denzap.InnometricsTran...	file:///private/var/folders/2w/c...	2019-10-03 13:32:31 +0300	2019-10-03 13:32:37 +0300	00:00:06		
App Focus	InnometricsTransfer	com.denzap.InnometricsTran...	file:///private/var/folders/2w/c...	2019-10-03 13:32:29 +0300	2019-10-03 13:33:07 +0300	00:00:38		
App Focus	Finder	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:32:21 +0300	2019-10-03 13:32:29 +0300	00:00:08		
App Focus	CoreServicesUIAgent	com.apple.coreservices.uiage...	file:///System/Library/CoreSer...	2019-10-03 13:32:20 +0300	2019-10-03 13:32:21 +0300	00:00:01		
App Focus	Finder	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:32:19 +0300	2019-10-03 13:32:20 +0300	00:00:00		
Idle	InnoMetricsCollector	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:32:01 +0300	2019-10-03 13:32:18 +0300	00:00:17		
App Focus	Finder	com.apple.finder	file:///System/Library/CoreSer...	2019-10-03 13:32:01 +0300	2019-10-03 13:32:19 +0300	00:00:18		
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:06:57 +0300	2019-03-27 20:07:03 +0300	00:00:05	Assignment	https://moodle.innopolis.univ...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:06:06 +0300	2019-03-27 20:06:11 +0300	00:00:05	Assignment	https://moodle.innopolis.univ...
Idle	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:05:37 +0300	2019-03-27 20:05:54 +0300	00:00:17		
Idle	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:05:31 +0300	2019-03-27 20:05:37 +0300	00:00:06		
App Focus	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:05:30 +0300	2019-03-27 20:06:05 +0300	00:00:35		
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:05:05 +0300	2019-03-27 20:05:30 +0300	00:00:25	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:05:00 +0300	2019-03-27 20:05:05 +0300	00:00:05	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
Idle	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:59 +0300	2019-03-27 20:05:23 +0300	00:00:24	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:55 +0300	2019-03-27 20:05:00 +0300	00:00:05	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:44 +0300	2019-03-27 20:04:55 +0300	00:00:10	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
Idle	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:32 +0300	2019-03-27 20:04:58 +0300	00:00:26	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:29 +0300	2019-03-27 20:04:44 +0300	00:00:15	socket_bind problem - PHP -...	https://www.sitepoint.com/co...
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:04:24 +0300	2019-03-27 20:04:29 +0300	00:00:05	What Is My IP? Shows your re...	https://www.whatismyip.com/
Idle	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:04:11 +0300	2019-03-27 20:04:18 +0300	00:00:07		
Idle	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:03:24 +0300	2019-03-27 20:04:06 +0300	00:00:42		
App Focus	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:03:23 +0300	2019-03-27 20:04:24 +0300	00:01:01		
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:03:19 +0300	2019-03-27 20:03:23 +0300	00:00:03	What Is My IP? Shows your re...	https://www.whatismyip.com/
App Focus	Terminal	com.apple.Terminal	file:///Applications/Utilities/Te...	2019-03-27 20:02:58 +0300	2019-03-27 20:03:19 +0300	00:00:21		
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:02:52 +0300	2019-03-27 20:02:58 +0300	00:00:05	What Is My IP? Shows your re...	https://www.whatismyip.com/
App Focus	Google Chrome	com.google.Chrome	file:///Applications/Google%2...	2019-03-27 20:02:47 +0300	2019-03-27 20:02:52 +0300	00:00:05	Networks/Week09 at master ·...	https://github.com/temur-kh/...

Fig. 4. InnoMetrics Transfer

It consists of two main areas: the left bar and the process metric list on the right. We have the feature to filter the processes based on keywords, applications, by date. Also, the user can select multiple metrics and hit the Backspace key to delete the metrics before sending them to the server (privacy feature). The right panel presents the data obtained from the collector (connects to a local database where collector was sending data). Finally, user can send the metrics to the server. After selecting the metrics user wants to send to the server, and hitting the "Send Metrics" button, user gets presented with a pop up (Fig. 5).


Would you like to send metrics to the remote server?

All metrics that you see on the screen would be send to the remote server.

Fig. 5. Popup for sending data to the server

Merging InnoMetrics Collector and InnoMetrics Transfer. As mentioned above, there is a dependency between these two applications. One first needs to collect data using the collector and then transfer the data using transfer. The UX would be better if everything was done in the same application, not requiring users to use both of the apps. That is why it was decided to merge the two applications. As a result of this decision, no longer do the users have to click on the submit button to send the data to the server. The data is automatically and periodically being sent as they are using the computer, without any distractions.

Auto Update Functionality. Desktop applications in general (deployed outside of the Apple App Store) suffer from the inability to update frequently [1]. That is why it was decided to implement the auto update functionality. For this we used the open source framework for auto update of MacOS applications called Sparkle (https://github.com/sparkle-project/Sparkle) [13]. It's a widely known and used framework that allowed to easily update our apps. As the two applications have been merged, there was no need to add the auto update functionality to the Innometrics Transfer. After implementing this functionality, the process of quickly iterating on versions became much quicker and it allows to become more agile when responding to user feedback [5,6,11,16,24]. Let us stress that the use of an open source application appears particularly suited as it enhances the reliability and the adoption of the system, as evidenced in several existing works [7,9,12].

4 Conclusion

To sum up, the system of InnoMetrics was developed for MacOS have been described thoroughly throughout the paper. The energy-related metrics are easily derived from the device with the help of Activity Monitor service in the MacOS. All kinds of metrics were derived and analysed based on the research studies. MacOS framework consists of two basic components: MacOS Collector and Transfer. The details of collecting data and its types, properties and their transfer is integrated in the system. Furthermore, the User Interface developed as a separate system - InnoMetrics Dashboard- for visual representation of the collected data and analysis, was presented. In this report, we have presented: the overall information how to get energy metrics in terms of MacOS systems, a deeper view of how the data collector and transfer operate with the back-end, displayed some results of development phase.

Further improvements of the system will be concentrated on the testing of the system with the functioning industrial companies, adding the external agents for deeper analysis of the software development process like Trello, GitLab, Github, and others. Besides, future work will include exploration of data analysis patterns and best practices for data analysis referring to the data records collected from the Data Collector.

References

1. Mozilla blog: what does the osx energy impact actually measure? https://blog.mozilla.org/nnethercote/2015/08/26/what-does-the-os-x-activitymonitors-energy-impact-actually-measure/. Accessed 29 Jan 2020
2. About mac notebook batteries (2020). https://support.apple.com/en-au/HT204054. Accessed 29 Jan 2020
3. How to use activity monitor on your mac (2020). https://support.apple.com/en-au/HT201464. Accessed 21 Feb 2020
4. Bykov, A., et al.: A new architecture and implementation strategy for non-invasive software measurement systems. In: Proceedings of the 33rd Annual ACM Symposium on Applied Computing - SAC 2018. ACM Press (2018). https://doi.org/10.1145/3167132.3167327
5. Coman, I.D., Robillard, P.N., Sillitti, A., Succi, G.: Cooperation, collaboration and pair-programming: field studies on backup behavior. J. Syst. Softw. **91**, 124–134 (2014). https://doi.org/10.1016/j.jss.2013.12.037
6. Corral, L., Sillitti, A., Succi, G.: Software assurance practices for mobile applications. Computing **97**(10), 1001–1022 (2015). https://doi.org/10.1007/s00607-014-0395-8
7. Di Bella, E., Sillitti, A., Succi, G.: A multivariate classification of open source developers. Inf. Sci. **221**, 72–83 (2013)
8. Ergasheva, S., Khomyakov, I., Kruglov, A., Succil, G.: Metrics of energy consumption in software systems: a systematic literature review. IOP Conf. Ser. Earth Environ. Sci. **431**, 012051 (2020). https://doi.org/10.1088/1755-1315/431/1/012051

9. Fitzgerald, B., Kesan, J.P., Russo, B., Shaikh, M., Succi, G.: Adopting Open Source Software: A Practical Guide. The MIT Press, Cambridge (2011)
10. Jagroep, E., Procaccianti, G., van der Werf, J.M., Brinkkemper, S., Blom, L., van Vliet, R.: Energy efficiency on the product roadmap: an empirical study across releases of a software product. J. Softw. Evol. Process **29**(2), e1852 (2017). https://doi.org/10.1002/smr.1852
11. Janes, A., Succi, G.: Lean Software Development in Action. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-00503-9
12. Kovács, G.L., Drozdik, S., Zuliani, P., Succi, G.: Open source software for the public administration. In: Proceedings of the 6th International Workshop on Computer Science and Information Technologies, October 2004
13. Mathur, A.: A human-centered approach to improving the user experience of software updates (2016)
14. Maurer, F., Succi, G., Holz, H., Kötting, B., Goldmann, S., Dellen, B.: Software process support over the internet. In: Proceedings of the 21st International Conference on Software Engineering, ICSE 1999, pp. 642–645. ACM, May 1999. https://doi.org/10.1145/302405.302913. http://doi.acm.org/10.1145/302405.302913
15. Musílek, P., Pedrycz, W., Sun, N., Succi, G.: On the sensitivity of COCOMO II software cost estimation model. In: Proceedings of the 8th International Symposium on Software Metrics, METRICS 2002, pp. 13–20. IEEE Computer Society, June 2002. https://doi.org/10.1109/METRIC.2002.1011321. http://dl.acm.org/citation.cfm?id=823457.824044
16. Pedrycz, W., Russo, B., Succi, G.: A model of job satisfaction for collaborative development processes. J. Syst. Softw. **84**(5), 739–752 (2011). https://doi.org/10.1016/j.jss.2010.12.018
17. Pedrycz, W., Russo, B., Succi, G.: Knowledge transfer in system modeling and its realization through an optimal allocation of information granularity. Appl. Soft Comput. **12**(8), 1985–1995 (2012). https://doi.org/10.1016/j.asoc.2012.02.004
18. Rahmati, A., Qian, A., Zhong, L.: Understanding human-battery interaction on mobile phones. In: Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI 2007. ACM Press (2007). https://doi.org/10.1145/1377999.1378017
19. Ronchetti, M., Succi, G., Pedrycz, W., Russo, B.: Early estimation of software size in object-oriented environments a case study in a CMM level 3 software firm. Inf. Sci. **176**(5), 475–489 (2006). https://doi.org/10.1016/j.ins.2004.08.012
20. Rossi, B., Russo, B., Succi, G.: Modelling failures occurrences of open source software with reliability growth. In: Ågerfalk, P., Boldyreff, C., González-Barahona, J.M., Madey, G.R., Noll, J. (eds.) OSS 2010. IAICT, vol. 319, pp. 268–280. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13244-5_21
21. Saborido, R., Arnaoudova, V.V., Beltrame, G., Khomh, F., Antoniol, G.: On the impact of sampling frequency on software energy measurements (2015). https://doi.org/10.7287/peerj.preprints.1219v2
22. Scotto, M., Sillitti, A., Succi, G., Vernazza, T.: A relational approach to software metrics. In: Proceedings of the 2004 ACM Symposium on Applied Computing, SAC 2004, pp. 1536–1540. ACM (2004). https://doi.org/10.1145/967900.968207. http://doi.acm.org/10.1145/967900.968207
23. Sillitti, A., Janes, A., Succi, G., Vernazza, T.: Measures for mobile users: an architecture. J. Syst. Archit. **50**(7), 393–405 (2004). https://doi.org/10.1016/j.sysarc.2003.09.005

24. Sillitti, A., Succi, G., Vlasenko, J.: Understanding the impact of pair programming on developers attention: a case study on a large industrial experimentation. In: Proceedings of the 34th International Conference on Software Engineering, ICSE 2012, pp. 1094–1101. IEEE Press, Piscataway, June 2012. https://doi.org/10.1109/ICSE.2012.6227110. http://dl.acm.org/citation.cfm?id=2337223.2337366
25. Sillitti, A., Vernazza, T., Succi, G.: Service oriented programming: a new paradigm of software reuse. In: Gacek, C. (ed.) ICSR 2002. LNCS, vol. 2319, pp. 269–280. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46020-9_19
26. Vernazza, T., Granatella, G., Succi, G., Benedicenti, L., Mintchev, M.: Defining metrics for software components. In: Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics, vol. XI, pp. 16–23, July 2000