A Study on an Evolution of a Data Collection System for Knowledge Representation

Jari SOINI[1], Markku KUUSISTO, Petri RANTANEN, Mika SAARI and Pekka SILLBERG

Tampere University, Faculty of Information Technology and Communication Sciences, Pori, Finland

Abstract. In this article the focus is on software evolution, which is an important part of software engineering. In practice, software development does not stop when a system is delivered but continues throughout the lifetime of the system. After the system has been deployed, external pressure for change can generate new requirements for the existing software. This change aspect, which is a characteristic of software engineering, should be taken into consideration when developing and modeling new software systems. In this paper the theme was studied using experience gained from the piloting of a reference system developed in an earlier research project carried out by Tampere University of Technology. Software evaluation is examined from the point of view of system developers, administrators (maintenance), and end users based on a concrete long-term piloting period.

Keywords. Information systems, software evolution, data representation, visualization

1. Introduction

The prototype system that had been developed in an earlier research project is examined in this paper from the point of view of software evaluation. Software engineering (e.g. [1]) includes the functions of the development of software and its maintenance and operation, which jointly provide the necessary conditions for the development of systems, in other words for software evolution. The present study that has been carried out on the subject is based on the long-term (four-year) piloting of the system and on the need for changes to the software which arose during the pilot. Over the long piloting period, the system required certain changes to be made both to the service and to the application itself, and this process of change can be seen as the evolution of the prototype system. In addition, the factors which caused the need for changes to be made in the software are described. Moreover, the study includes the end users' experiences of using the services during the piloting period.

The developed service which is described in this article acts as a reference system for the examination of software evolution. This service, called the "Data Collection

[1] Corresponding Author, Tampere University, Pohjoisranta 11 A, P.O. Box 300, FI-28101 Pori, Finland; E-mail: jari.soini@tuni.fi.

System", was developed to answer a real need in a collaboration project between Tampere University of Technology (TUT) and the City of Pori [2]. The main aim of the project was to study potential new technologies for managing and controlling conditions in buildings in a smart way. In the case which is used as an example in this article, the research was aimed at discovering resources for improving the quality of measurement data as well as better and more illustrative tracking of usage information in real time. The solution created in this case was a modeled system for increasing the reliability of the collected data and a smart application to support data collection, recording, and analysis.

This article describes the evolution of the "Data Collection System" during the long piloting period in a real usage environment. The article also describes how the end users of the service viewed and experienced the system and its evolution during the long piloting period. Furthermore, we discuss the future research topics that arose from this study as well as key observations and conclusions based on the results gained.

There is a wide stream of research on the topic of software evolution which was the starting point for this study. [1] justifies the importance of the subject in software engineering and presents a model of the software evaluation process. Also, [3] have examined software maintenance and evaluation issues widely and proposed their model of the software evaluation life cycle. In addition, [4] have carried out research in this area and they divide software evolution into the evolution of frameworks which causes changes to software and the evolution of software due to changing requirements. They present ideas to accommodate "smooth" software evolution. Yamashita et al. [5] present a dataset for further studies of software evolution. The dataset contains data from four real-life systems developed by six developers having several control variables. Vogel-Heuser et al. [6] discuss the challenges in the evolution of automated production systems (aPS) at length. Haitzer et al. [7] present support for incremental changes in their article containing four case studies. Related to the example service used in this article, Aggrawal et al. [8] present a tool to evaluate the energy consumption changes induced by new versions of software. The study area of the original research [9] on manual data collection was also revised, but it was challenging to find newer reports on the subject. Most of the research found on the topic is being done on how to automate the process, but there could still be cases where study of manual processes or semi-automatic tools could prove to be useful (e.g. [10],[11]).

The structure of the paper is as follows: Section 2 describes the context of the study and the structure of the prototype system under examination. Section 3 presents the evolution of the prototype system. Next, Section 4 deals with user experiences during the system piloting. Section 5 consider alternatives for utilizing the data on the prototype system. Finally, the conclusions are presented in section 6.

2. Research Basis

2.1. System Evolution

The prototype system developed in a recent research project is examined in this paper from the point of view of system evolution. The concept of system evolution is based on the uncertainty of predicting the real world, in other words on the fact that it is not possible to understand changes in the environments of systems beforehand. In the case of "systems," evolution refers to the phenomenon in which the effect of the environmental factors of a system in some way changes the demands of the system and thus causes pressure for changes in its operation [12]. In other words, the systems should be capable of being changed (continuously) so that their ability to operate in their environment is preserved.

In addition, information systems have an evolutionary character. Software engineering (e.g. [1]) includes the functions of software development and its maintenance and operation, which jointly provide the necessary conditions for the development of systems, in other words system evolution. Within the sphere of software engineering, the evolution of systems refers to the development of the system during its lifecycle [13],[14],[15]. Software developers must always take the above-mentioned adaptability of the system into consideration when designing and developing the system. Section 3 shows the stages in the evolution of the prototype system developed in an earlier research project and the adaptability of the system to the need for change, which emerged during the long-term piloting.

In practice, software engineering is by nature based on feedback, a constant process, in other words it is only possible to adapt the developed system to take its environment better into account after release and implementation. In other words, feedback from a system and its operation is obtained only once the system has been adopted in a certain context. Feedback on the operation of the system and on its quality is always only received after the version has been released, when the end users have started using it [16]. In fact, the implementation of an information system is quite a complex process, which can be described as a technological, social, organizational, and people-centered operation [17]. Often the changeover from the predominant system (in this case pen and paper) to another ("measuring client") is connected to the introduction of the information system. Eason [18] classifies five different methods of implementing a change in practice. In our case, the so-called "big bang" method was used – in which the change takes place all at once during a certain agreed time. Section 4 presents the change of practice and involvement during the system evolution from the end user's point of view.

2.2. The Reference System

The reference system, which acts as the foundation of the evolution examination in this paper, was developed during an earlier research project ("Smart Analysis of Property Systems Data"), implemented by Tampere University of Technology (TUT). The research focused on studying potential new technologies for managing and controlling conditions in buildings in a smart way. One of the main results of this project was a system that helps property staff in their daily work related to data collection and monitoring. The objective in this case was to improve the quality of a specific measurement data collection process, eliminate human error in data collection, improve the traceability of the collected data, and facilitate the analysis of the said data. In 2014, a "measuring client" was developed as part of the project. The technology was mainly created to combat the quality issues of manually collected data. The measuring client was developed to replace data gathering with pen and paper. This collection had resulted in several errors in copying long numbers from one field to another and back to a system in another location.

Figure 1 (below) shows an overview of the system. The service is available over the Internet where both the *Management User Interface* and *Client* application can be connected. Basically, the management user interface is a web page accessible with a web browser. There, the system administrator can configure a particular object and interpret the results sent by the client. For example, the results can be viewed as raw data or plotted as a chart.

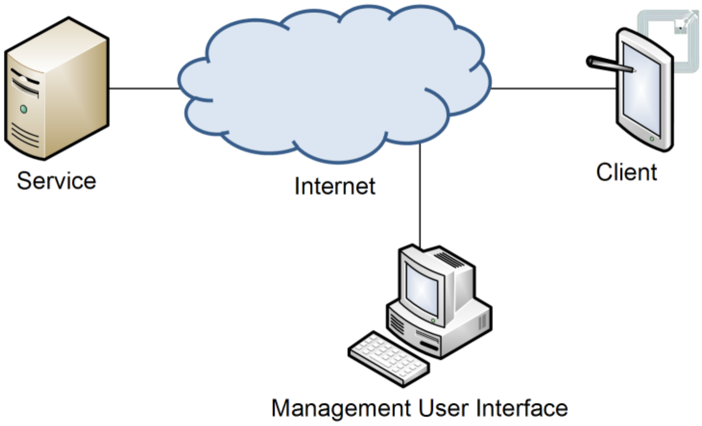

Figure 1. System overview [9].

The client, in Figure 1 (above), is separated in this case, but in principle there are no restrictions on which kind of device can use the management interface. In practice, it might be more convenient to use a desktop computer to configure the system, as it generally has larger sized displays. The client application in our case is programmed for Android devices. The system uses Near Field Communication (NFC) compatible [19] Radio Frequency Identification (RFID) tags [20],[21] for identifying the connection and Javascript Object Notation (JSON) and Representational State Transfer (REST) interfaces for data transfer. Each information collection cycle consists of three phases which are: identifying the object, inputting the data, and saving the data. The identification of the object was perhaps the most novel in the technological sense, as it was determined by reading a unique ID number from an RFID tag. The passive RFID tags that were chosen proved to be reliable, easy to use, and a feature appreciated by the end users – there were no complaints about this part of the system. The measuring client was developed to replace data gathering with pen and paper. This collection method had resulted in several errors when copying long series of numbers from one field to another and back to a system in another location. It was not financially viable to replace the old system with a fully automatic one, however. Lastly, in the third phase, technology was created to aid the human operator in the process of data gathering, which had previously been a tedious task to carry out. With the aid of a computer, the gathered data was immediately available in visual form. For the end user, the ability to see any abnormalities immediately was perhaps the most satisfying feature of the system – a visual proof that everything was going as expected. Figure 2 below presents the system in real use.

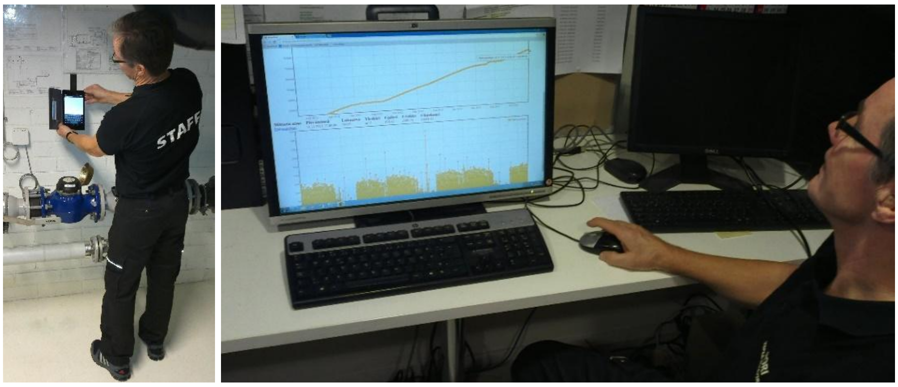

Figure 2. System in real use.

The measuring client was seen as the point of largest impact by the City of Pori [2] at the time of its creation. The system acts as a concrete example of attempts to obtain more reliable and thus better quality measurement data, and it also improves the visual presentation of collected data for analysis. The visualization aspect alone is very important if a large amount of data is collected and recorded. There is evidence that visualization can add significant value when trying to understand the raw data available. Data visualizations allow the user to gain insight into the data, and also help in finding recurring patterns and outliers among the collected data. It is also easy to adapt the “data collection” system in question for other operating environments.

The following sections 3 and 4 describe the evolution of the prototype system from its initial implementation (in 2014) until the end of the long-term piloting (in 2018). Furthermore, we relate the end users' experiences of the change in the way of collecting data, experiences of using the new system and observations of the piloting of the system, i.e., their involvement in the evolution stage.

3. Evolution of the Prototype System

Externally, the look and feel of the application as well as the feature set provided for the property maintenance staff remained virtually identical throughout the four-year period. From the researchers’ side, only minimal resources were available for the maintenance of the prototype system. Initially, the idea was that after a shorter testing period the system would be deployed at other premises, perhaps maintained by a software company operating in the area, but ultimately, this change never happened. Nevertheless, there was a constant requirement and possibility that the system could be moved at some point of time during the longer time period. Behind the scenes, this required certain changes to be made to both the service and to the application itself. This process of change can be seen as the evolution of the prototype system, and is explained in more depth in the following subsections.

3.1. Service

The service was designed to work as a purpose-built data collection service. This also meant that the potential user numbers for the system were quite limited. The initial service was deployed on a basic computer used as a web server on university premises.

In Figure 3, it is shown how the service was gradually deployed for more and more containerized and host machine agnostic environments. The current version is also located at the university, but has been running on a Linux-based virtual machine, and lately on networked Docker images, making moving and re-deploying the service slightly easier. Future implementation, be it microservices, serverless, or any other cloud computing solution, has not been considered as the service will be transferred to a third party service provider. The maintenance operations will also be transferred at the same time.

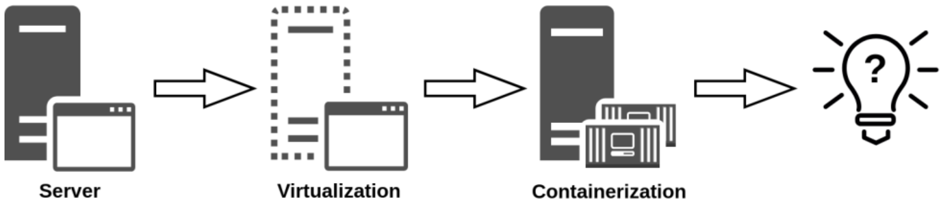

Figure 3. Deployment targets during the lifespan of the service.

From the maintenance point of view, the service has been a low demand target. During the four-year period, there were only a few times that the research team were contacted due to problems in the system:

- Once the Android tablet was not able to connect to the Internet, but the reason was due to poor contact with the SIM card, and reseating the card was enough to rectify the issue.
- We were asked whether the user interface could be modified to display data from a shorter time window initially (i.e., one or two years) with an option to zoom out to see the whole data set.
- The battery of the tablet had to be changed due to low capacity because of old age.
- During the fourth year of the pilot, the service suffered an expected downtime of about two weeks which was not communicated to the end users. This led the users to cease collection of the data for a while. It occurred during summer time, which is a typical low activity season in Finnish companies and institutions.

The service shared some of its code-base with other projects, and the problems detected and fixed there were propagated into the service. Most of the allocated work hours originated from making the maintenance of the service less cumbersome. Investments were made in the backbone of the system which allowed the continuation of the service with minimal further input. Nevertheless, maintenance tasks such as occasional sanity checks of the service (e.g., whether the user can log in and view the data), correction of obvious input errors, and preparation of and verification of database backups, were still conducted manually.

3.2. Client

During the development of the client application (in 2013), the newest Android version available was version 4 with version 5 following the next year. Especially at that time, the version updates for consumer devices were anything but guaranteed and often newer versions were not made available for older devices. Both our development devices and the “production” devices at the swimming pool used various versions of

Android 4.0 – Android 4.4 and were later updated to Android 5.0. The production devices were never updated to more recent Android versions, because the newer versions did not provide any significant improvements for our specific use case.

This also meant that very few code changes were necessary. The backward compatibility on Android devices is, in general, reasonably good, and there were very few problems running or testing the application on the newer Android devices (and versions) available at the university. A minor obstacle was the deprecation and eventual removal of the Apache HttpClient library. Porting the application code to use some other library or different Android APIs would not have been too difficult, but it was still much easier to use an Apache library Android port [22]. Similarly, Apache Commons Math libraries and Google's Gson libraries were updated to more recent versions, but these changes did not cause any issues. During 2014, Java version 8 was released and many of the language's features (e.g., try-with-resources) were later added to the Android SDK, but these could not be used in our implementation, because they were not supported on the older devices.

Perhaps most surprisingly, the biggest issue turned out to be Google's decision to abandon the older Eclipse-based Android SDK in favor of the IntelliJ-based Android Studio. In our case the original codebase had been lying dormant for a couple of years and the plan was to import the project directly to Android Studio version 3. However, in practice, this plan turned out to be impossible to achieve. The project could not be compiled in any reasonable way, much less deployed or debugged on the newer IDE. The directory structure, configuration files, build files of Maven and Gradle required so many changes that it was ultimately easier to create an entirely new Android project and to simply copy-paste all the required files to the newer project format, and to manually update the configuration and build files with whatever modifications were necessary.

4. User Experiences

The system development, and also maintenance after termination of the project in March 2015, was the responsibility of the SEIntS (Software Engineering and Intelligent Systems) research group at Tampere University of Technology (TUT), Pori unit. Naturally, we were interested in how the users of the service viewed our system. As mentioned before, the service had remained unchanged from the users' point-of-view, but there had been changes in the lower level implementation. We wished to know if the users had noticed any variations in the quality of service. Furthermore, we also tried to find out how – and how *extensively* – the service was used in practice, and whether or not the users felt that parts of the service could be done in a different way or did not work as well as they should.

In order to obtain feedback from the swimming pool employees, a *semi-structured interview* (e.g., [23]), which is a special case of the interview situation in which the dialogue is mainly free-form, was organized. According to [24], the interview as a material gathering method is more suitable in situations where the interviewees are allowed to talk about their experiences freely. For this reason a semi-structured interview was chosen as the type of interview used in this study.

For practical reasons, only the personnel currently working at the swimming pool were interviewed and not all the staff who might have used the application in the past. According to the swimming pool management, only full-time employees used the application, and the persons interviewed had used the application for the whole four-year testing period, and thus, should have had a clear idea of the features of the application. Additionally, the person responsible for the management and usage of collected data was interviewed to get an idea of where and how the data was actually used. The interviewees did not know any particular details about the evolution of the service and they were not made aware of the changes during the interview. The idea was to find out whether the users themselves would report any issues that could be traced back to the various behind-the-scenes modifications performed on the system.

Ultimately, the property maintenance staff had very few issues to report, and especially, there were no major problems that could be directly traced to any particular evolutionary modification. The Android application was used on a daily basis, i.e., five days a week, from Monday to Friday, and generally in the mornings. The management web page was not accessed quite as regularly (usually once per week). In any case, quality-of-service related issues would have been encountered in daily use if there had been any. The main findings of the user interviews are summarized in the subsections below.

4.1. Management Perspective

It was found that the data was not analyzed very deeply or used in further decision making. The primary purpose for collecting data was to keep a historical record of the energy usage. In the case of problems (leaking pipes, instrument failures, etc.), the data could be processed to find out when the problems had first occurred.

We also asked about the possibility of releasing the entire data set with accurate daily statistics as free and open data, but currently no release was planned. There had not been discussions about whether or not the data could be useful for third-party organizations or companies. Furthermore, there were no concrete ideas about feasible and practical use cases where open data could be offered or utilized. On the other hand, the collected data was considered important for the singular use case in which it was employed, and although losing the data would not have been disastrous for the operation of the swimming pool, it would at the very least be "very inconvenient."

Furthermore, the swimming pool management was interested in buying a similar service from a software company in the event that the university could not continue hosting the service in the future. There had been no complaints about the use and functionalities of the application or the management web page, and in general, the attitudes toward the service were very positive.

4.2. User Perspective

Using the system was discovered to be less "prototype testing", and for the employees it had become more of an everyday work routine. Feedback given on use of the system was very positive, and none of the employees were willing to change back to the pen-and-paper or Excel spreadsheet-based approaches. The web page can also be used to generate Excel spreadsheets, but this feature was not used. Perhaps there was no real need for Excel files after alternative means were provided. In any case, such a feature was not requested by the staff during the four-year testing period. Another feature that was very little used was the graphical diagram view of the Android application that could be used to quickly view similar figures as those available on the management web page (and illustrated in Figures 4 and 5, in Section 5). All in all, the simplicity of the application was much appreciated and it was seen as a handy and reliable tool for collecting data. Viewing and browsing the data was considered more user friendly on a larger screen (desktop PC), and implementing excessive data analysis tools on the mobile device would not, at least in this case, be worth the extra effort.

There had been only a few problems with accessing the web page used for checking the energy consumption statistics, some problems with the automatically calculated threshold values used for checking valid input values in the Android application (threshold values were set too low/high in the initial setup), and a broken battery on an Android device, amongst other minor issues, but in general, the system had worked very well and stably. There had not been any perceived differences in the quality of the service provided by the system, and the usability and performance of the Android application and management web page were fluent and stable during the four-year piloting period.

5. Future Studies

The application has been used to collect daily district heating, water, and electricity consumption statistics for the past five years. An example graph of the entire data – in this case, district heating – is shown in Figure 4 and a more zoomed-in version showing the previous year can be seen in Figure 5. Similar figures are also available for water and electricity and accessible by using the management web page, as described in more depth in an earlier publication [9].

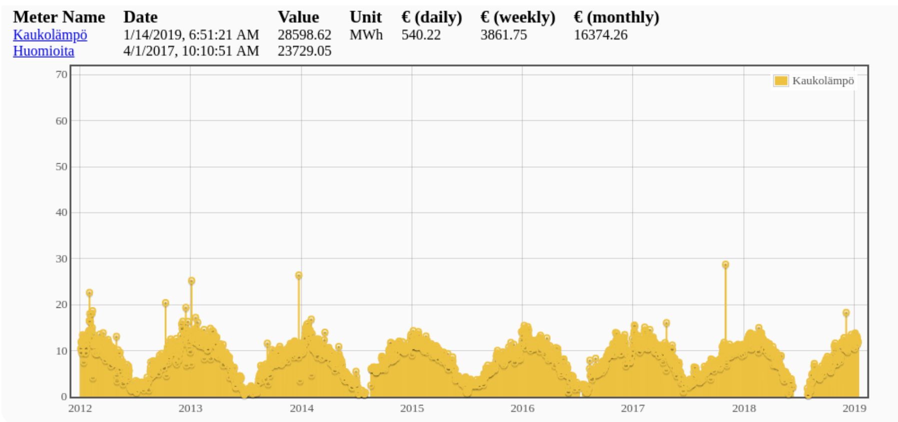

Figure 4. Annual consumption of district heating over a period lasting seven years.

The collected data also includes values from earlier years (since 2012), which had been collected by the staff using a simple pen-and-paper approach and later converted to Excel spreadsheets, which was used to import the data to our system. The data was collected both by pen and paper and by using our application during a short transition period in 2014, and for that reason, there is no visible gap in the data showing the move from manual data collection to using our application.

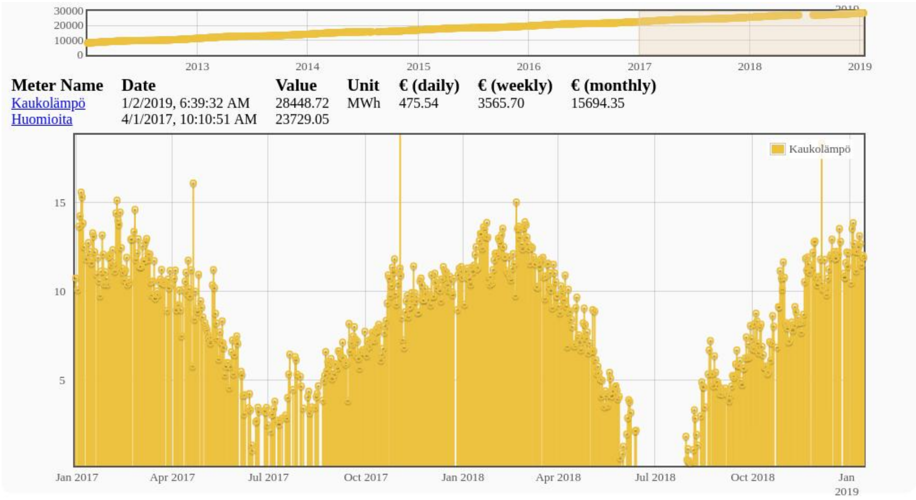

Figure 5. Graph zoomed in closer to show a seasonal variance.

In both the figures above (4 and 5), certain seasonal variances can be clearly seen. The upper part of Figure 5 shows a cumulative overview of the meter value in question and can also be used to select start and end points (i.e., date range for values) visible in the lower parts of Figure 4 and Figure 5. The zoomed-in views both highlight variances from the expected values. The indoor swimming pool is closed in the summer and maintenance is usually scheduled for the summer months, causing the "lower" values visible in both figures. The collected data is primarily used for validation purposes if or when problems are detected later on, even though issues could be detected earlier by utilizing automated means. For example, several "spikes" or abnormal measurements can be easily identified in both figures. A simple link for generating and downloading an Excel file containing the abnormal values detected by the system was later added to the web page. The link ("notifications") is visible in both figures; however, this feature however was seldom used.

Furthermore, the swimming pool has access to other statistics that could be combined with the data collected by our system. For example, the number of customers (user history) or energy prices could be used to optimize the overall energy consumption. Energy consumption optimization is currently a significant topic, and in our case, there already exists comprehensive data that could be used to develop algorithms or methods for lowering energy costs and usage. The data collected by the application can also be downloaded for free from GitHub [25] both as an SQL database dump and as CSV files. The GitHub repository also contains source codes for the web service, for the web page, and for the Android application.

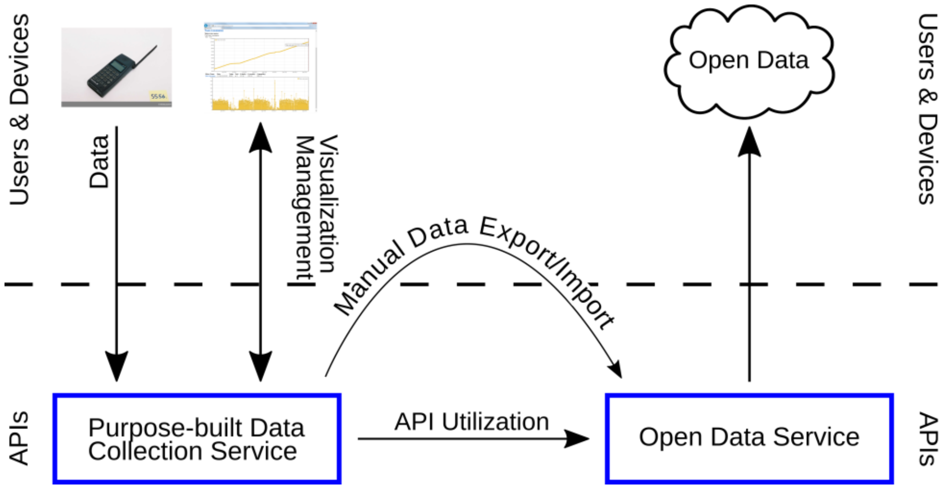

Figure 6. Data utilization plan.

There are plans to utilize the data in open data services currently run or to be established by the City of Pori. Thus, in the future, the data could be more easily retrieved and viewed by other researchers or other parties interested in energy consumption statistics. This is illustrated on a very high level in Figure 6. The current usage is limited to the "Purpose-built Data Collection Service" visible on the left side of the figure. The REST APIs originally designed to supply data for the JavaScript and HTML5 based management web page could be used to share the data to other services, but in practice, the data is shared manually (e.g., by exporting Excel spreadsheets).

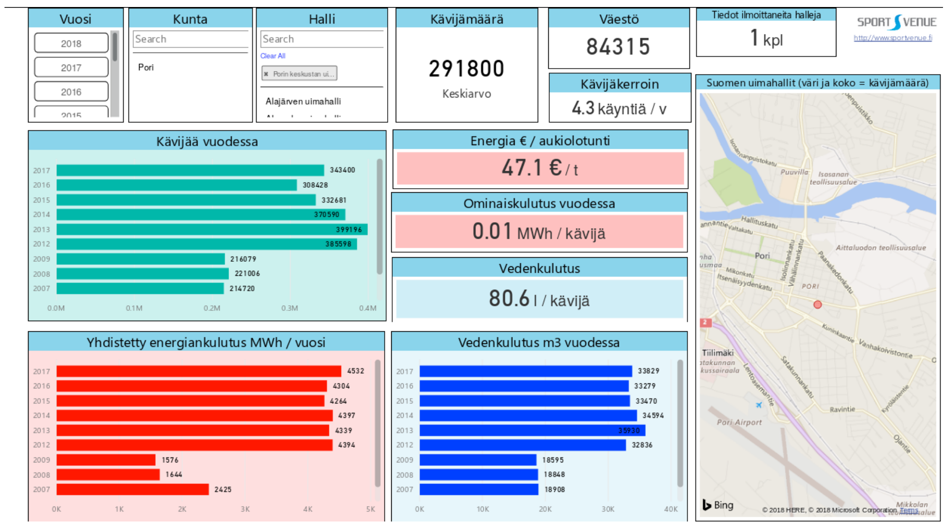

Figure 7. Data usage statistics on LIPAS service [26].

In addition to the data dumps available at GitHub, a small subset of the collected data is already available in the LIPAS service [26] developed by the University of Jyväskylä. The LIPAS service is a Finnish national database of sport facilities, routes for outdoor activities, and recreational areas. In the case of the indoor swimming pool of the City of

Pori, the page lists some basic information about the facility as well as yearly averages for water, heating, and electricity consumption (illustrated in Figure 7). In Finland, energy consumption data must be collected for swimming pools to receive public funding, but there is no exact information on how much of this data is collected automatically or how extensively the data is used for further analysis. In our case, the data is manually exported and imported into the LIPAS service even though direct API access would also be available. In any case, the service has similar statistics for other indoor swimming pools, and also has functionality that allows the comparison of different faculties with each other. Furthermore, the property maintenance staff reported that the yearly averages of the energy statistics are provided for VTT (Technical Research Centre of Finland) [27].

In February 2019, this long piloting period operated by TUT will come to an end and the maintenance of the system will be transferred to a local software company. This also marks the end of our research on the evolution and user experiences of the system. Nevertheless, there are still ample opportunities for further research based on the collected data.

6. Conclusions

The application developed fulfilled its task and was seen as an important improvement on the quality of the work performed by the property maintenance staff. A minor complaint received from the staff was about our lack of notification about short maintenance breaks or known network downtimes. The Android application had a built-in backup Sqlite database, which was used to store the input data when the network service was unavailable, so no data was lost, but not being able to access the web page was considered slightly problematic. During the four-year period, the downtimes were very short, perhaps a day or two per year on average. In fact, the lack of issues allowed the system to run without interference for months at a time, if not entirely forgotten, then at least out of mind - from the research team's perspective at least. The routine-like work with the prototype system may increase expectations about the stability and availability of the system, and for us this was somewhat challenging, because the resources available for system maintenance were not comparable to what could be provided by a commercial company, and the original system developers (university researchers) working during the piloting period on other research projects had to do extra work to fix minor issues.

The results of the study show that the original prototype system was very successful, being continuously used for four years and being incorporated into daily working routines. The system had evolved over the years as described earlier in the article. However, this evolution was not apparent to the common user of the software. The user experience remained the same throughout the period, and all maintenance breaks to change the background software were made in a way that did not affect the end user. After the initial development cycle the system was maintained with a minimal effort from university resources. This was made possible due to the successful prototype developed.

To date, the data produced by the system has been used by a fairly small group of people within the swimming hall. The small amount of data usage begs the question of how it could be used or utilized more widely. Lately, some of the data has been published in the LIPAS service, but most of the data remains hidden. In future research it would be interesting if there were some framework for indicating systems that are run continuously but are not used. The authors speculate that there may be a lot of these kind of systems in existence in different organizations. There are several reasons why this kind of situation could arise, for example the system works autonomously and is forgotten about.

From the evolution point of view, the system managed to survive the implemented modifications well. The success of the process can largely be accredited to the proper hardware and software choices made during the development of the application. This has enabled the service to run without major changes for the whole four-year testing period. Nevertheless, it is challenging to give any unambiguous and solid advice for any technology selection process. In 2013, NFC was a promising technology and the Android operating system was strong (if not quite as dominant as in 2019), but technologies can change quickly and, when operating with minimal resources, any major hardware or API changes can become problematic.

Another point that enabled the pilot to run successfully was the successful definition of roles for everyone participating in our pilot test case. The property maintenance staff were strictly responsible for the maintenance of the devices integral to the management of the swimming pool itself, but had no role other than providing the basic requirements for the design and management of the data collection service. In fact, the research team would not even have had the competence to manage the swimming pool hardware. In our case, the manual data collection (i.e., users typing the values into the client device) also created a "gap" in the system integration, allowing the various systems to remain very loosely coupled. This is not always possible or desired, but in our case, it would have enabled us to make bigger changes to the system if required. It should also make the maintenance transition to a commercial company more fluent.

Finally, when we started the research for the original prototype in 2013, the research on manual data collection seemed to be very limited. Now, almost six years later, the situation has not noticeably improved with the research focus turning more and more to fully automatic or to so-called "smart systems". The future might be intelligent, but there are still a huge number of systems that have a "manual" component. These systems could be used to generate data for more advanced services if bridge and integration technologies continue to be studied and developed.

To briefly sum up the results of the study, it can be stated that in the reference system which was the subject of the study, very few requirements for change arose after the implementation of the system and they had had only minor significance regarding the use of the software, and on the other hand, the modifications made facilitated the maintenance of the system. The planning of the system and technology choices that were made in the development phase were very successful and the software evolution itself was well managed and fairly easy in this case.

References

[1] I. Sommerville, Software engineering (10th ed.), International Computer Science. Pearson Education Ltd. Essex, England, 2016.

[2] City of Pori, https://www.pori.fi/en (Retrieved 7/1/2019 World Wide Web)

[3] K. H. Bennett and V. T. Rajlich, "Software maintenance and evolution: a roadmap", In Proceedings of the Conference on the Future of Software Engineering, ACM, (2000), 73-87.

[4] Q. Gao, J. Li, Y. Xiong, D. Hao, X. Xiao, K. Taneja, L. Zhang, and T. Xie, "High-confidence software evolution", Science China Information Sciences. Vol. 59 (2016)
[5] S. A. Yamashita, S. Abtahizadeh, F. Khomh and Y. Guéhéneuc, "Software Evolution and Quality Data from Controlled, Multiple, Industrial Case Studies," 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), Buenos Aires, (2017), 507-510.
[6] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: Challenges and research directions", Journal of Systems and Software, Vol. 110, (2015), 54-84.
[7] T. Haitzer, E. Navarro and U. Zdun, "Reconciling software architecture and source code in support of software evolution", Journal of Systems and Software, Vol. 123, (2017), 119-144.
[8] K. Aggarwal, A. Hindle and E. Stroulia, "GreenAdvisor: A tool for analyzing the impact of software evolution on energy consumption,"IEEE International Conference on Software Maintenance and Evolution (ICSME2015), Bremen, Germany, (2015), 311-320.
[9] Soini, J., Sillberg, P. and Rantanen, P., "Prototype System for Improving Manually Collected Data Quality", In Z. Budimac and T. G. Grbac (Eds.), Proceedings of the 3rd Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications, SQAMIA 2014, September 19-22, (2014), Lovran, Croatia (pp. 99-106). (Ceur workshop proceedings; Vol. 1266). M. Jeusfeld c/o Redaktion Sun SITE.
[10] Z. Pengjun and M. McDonad, "An Investigation on the Manual Traffic Count Accuracy", Procedia - Social and Behavioral Sciences, Volume 43, (2012), 226-231.
[11] D. J. Findley, C. M. Cunningham and J. E. Hummer, "Comparison of mobile and manual data collection for roadway components", Transportation Research Part C: Emerging Technologies, Volume 19, Issue 3, (2011), 521-540.
[12] M. Lehman and J. F.Ramil, "Software evolution-Background, theory, practice", Information Processing Letters, Vol. 88, Issues 1–2, (2003), 33-44.
[13] C. Kemerer, and S. Slaughter, "An Empirical Approach to Studying Software Evolution", IEEE Transactions on Software Engineering, Vol. 25, Issue 4, (1999), 493-509.
[14] K. Bennett, "Software evaluation: past, present and future", Information and Software Technology, Vol. 38, No. 11, (1996), 673-680.
[15] D. E. Perry, "Dimensions of software evaluation", Proceedings International Conference on Software Maintenance, IEEE, (1994), 296-303.
[16] J. Humble and J. Molesky, "Why Enterprises Must Adopt DevOps to Enable Continuous Delivery", Cutter Business Technology Journal, Vol. 24, No. 8 (2011)
[17] S. Smithson and R. Hirshheim, "Analysing information systems evaluation: another look at an old problem", European Journal of Information Systems, Vol 7, No. 3, (1998), 158–174.
[18] K. Eason, Information Technology and Organisational Change. Taylor & Francis Ltd, London, UK, 1988.
[19] International Organization for Standardization. ISO/IEC 18092:2013, Information technology -- Telecommunications and information exchange between systems -- Near Field Communication -- Interface and Protocol (NFCIP-1).
[20] K. Finkenzeller, RFID Handbook: Radio-frequency identification fundamentals and applications. Wiley and Sons, Ltd, 1999.
[21] J. Nummela, Studies towards Utilizing Passive UHF RFID Technology in Paper Reel Supply Chains. Doctoral dissertation. Faculty of Computing and Electrical Engineering. Tampere University of Technology, Finland, 2010.
[22] The Apache Software Foundation, HttpClient for Android, https://hc.apache.org/httpcomponents-client-4.5.x/android-port.html (Retrieved 7/1/2019 World Wide Web)
[23] S. Hirsjärvi and H. Hurme, Tutkimushaastattelu: teemahaastattelun teoria ja käytäntö. Helsinki, Yliopistopaino (2001), ISBN 951-570-458-8
[24] U. Schultze and M. Avital, "Designing interviews to generate rich data for information systems research", Information and Organization, Vol. 21 No. 1, (2011), 1-16.
[25] GitHub, Tampere University Project Page, https://github.com/otula/kiiaudata/tree/master/datasets (Retrieved 7/1/2019 World Wide Web)
[26] Lipas.fi , https://www.jyu.fi/sport/en/cooperation/lipas (Retrieved 7/1/2019 World Wide Web)
[27] VTT, Technical Research Centre of Finland, https://www.vttresearch.com/about-us (Retrieved 7/1/2019 World Wide Web)