Utilization of the Software Life Cycle Model using Process-Centric Development and User-Centric Tool (PCD.UCT) model: A Review

K.G.H. Piumali
Faculty of Computing
General Sir John Kotelawala Defence University
Rathmalana, Sri Lanka
36-se-0011@kdu.ac.lk

***Abstract*— The major attention has shifted from processes to users in system development methodologies since the waterfall development approach. "Predictive cum Adaptive Systems Development Methodology for HydroGIS Tool Development" research study by Pradeep and Wijesekara (2017) introduces an automated hybrid development approach that builds and insightful Process-Centric Development as well as the User-Centric Tool (PCD.UCT) model. As a result, the study article shows how the two models use PCD to implement software life cycle models. UCT is evaluating methodologies for the development of a HydroGIS (Hydrological Geographic Information System) application that will accurately automate complex hydrology processes in a GIS context while meeting user needs.So, this paper provides an overview of the Process-Centric Development and User-Centric Tool (PCD.UCT) model that utilizes the Software Life Cycle Model**

Keywords—HydroGIStool,Hybrid,Predictive,Automate, Accurate,User-friendliness

I. Introduction

Mr.Pradeep and Mr.Wijesekara (2017) used UcT.PcD, a hybrid development methodology that combines waterfall, prototype, and repetitive development methodologies. The paper focuses on the process of hydrology model automation, which requires hydrologists to devote a significant amount of time and effort to the creation of a hydrology model. The major goal of this research was to combine and evaluate appropriate approaches in order to develop a HydroGIS (Hydrological Geographic Information System) tool[1]. During the tool's development, two improvements are done at the same time: (1) engineering process automation and user-friendliness. This program's key goals are the accuracy of automating hydrological tasks and meeting user needs. This development technique is then transformed into a software project management tool. As a result, a well-known and well-proven software developer[5] has emerged.Process Centric Development to User Centric Tool (PcD.UcT) has been introduced .So,the following sub areas will provide a broad knowledge on utilizing the life cycle models using PcD. UcT.

A. *Process-Centric Tool Development*

Pradeep and Wijesekara (2012) used a prototype development process to assess the size of the detention storage facility in Moratuwa. Wijesinghe and Wijesekera (Wijesinghe and Wijesekera,2010) D Combining the User-Centered Tool with the Process-Centered Algorithm It was determined to terminate prototyping once the user friendliness reached an acceptable level, 70 percent, and there was a need to integrate the process for additional evaluations. However, the process automation will continue until the recoded set of results is 100% accurate. Once the process accuracy criteria was met, the user interfaces were merged with the process codes, which were written in the same languages and on the same platforms. During the integration, user modification suggestions that received 70% satisfaction were implemented and moved on to the final summative evaluation. Maintaining the Integrity of the Specifications

B. *User-Centric Tool (PCD.UCT) model*

According to the User Requirements Prototype Automation, The tool was put through three different types of user evaluations when it came to user requirement automation. Following the requirement analysis, the tool's basic functions were discovered, and a prototype was constructed The prototypes were evaluated using a questionnaire as they were being developed. This software adequacy questionnaire evaluates the usability of the tool's first features. Then, based on customer feedback, a second prototype was developed and tested twice more until everyone was pleased. After then, the calculating components were integrated with the working prototype.

C. *Integrating the User Centric Tool with Process Centric Algorithm*

According to Pradeep and Wijesekara [1], it was determined to suspend prototype modification after the user friendliness reached an acceptable level, 70 percent, and the necessity to integrate the process for further evaluations. However, the process automation will continue until the recoded set of results is 100% accurate. Once the process accuracy criteria was met, the user interfaces were merged with the process codes, which were written in the same languages and on the same platforms.

D. *Engineering Applications*

Engineering is a well-established profession with a wealth of experience. The engineering process is characterized by extensive planning, designing, and drafting. Because the costs of failure are so high, the engineers devote a significant amount of time and money to the early stages. When the software engineering profession first emerged in the 1960s, it was a subset of engineering where women worked (Meyer, 2013). Hardware planning, design, implementation, and maintenance were the core computer tasks at the time, whereas software development was a painting project. With the rising use of software in computing[1], early software development approaches such as waterfall and parallel development models have emerged. These development approaches have phases that are similar to the general engineering process, such as a comprehensive study, design, and development. Nonetheless, with such a wide range of customers with varying levels of computing knowledge, the software's user interface needed to be more user-friendly. Due to the ambiguity of the requirements, developing software utilizing predictive approaches becomes a time-consuming task. Adaptive development approaches such as extreme programming and agile development versions gained popularity as a result. The automation of engineering applications was likewise vulnerable to the same fluctuations in user requirements[2]. The dilemma worsens as non-technical decision-makers are more likely to use engineering applications when making decisions. With the advancement of the profession, engineering calculations have become more sophisticated, necessitating more time and resources in the analysis, design, and development phases[3]. As a result, engineering applications necessitated the use of both predictive and adaptive development approaches at the same time, which is not feasible.

II. Literature Review

This section deals with existing researches that relate to the aforementioned Area and associated techniques in different areas related.

A. *Tool Requirements for HydroGIS*

According to Pradeep and Wijesekara's (2012) study, there is a need to develop a HydroGIS tool to manage urban flash floods. As a result, because urban flash flooding is a result of changes to urban land use, the tool must determine the impact of the changes on stormwater generation. Users should also be able to utilize the application to come up with dynamic engineering solutions to control excess stormwater. To complete the project, they identified three process modules: (1) incorporation of land parcel adjustments (2) calculated storm generation pre and post scenarios (3) incorporation of detention pit. This tool encourages the creation of a user-centric map-based interface with onscreen data input and dynamic attribute/land parcel updates. Furthermore, because the manipulations are carried out on much smaller urban land areas, the precision of the results is compromised. The fact that the tool's potential users are nonhydrological land managers is critical.

B. *Methodologies for System Development*

There are numerous system development approaches to choose from.As in table 1.0, all of these techniques are based on the planning, analysis, design, and implementation phases of the system development cycle. In addition, the waterfall technique might be regarded the first widely utilized development methodology in the 1970s. In the 1980s, a prototype was developed as an alternative to meet unmet development requirements. Finally, in the 2000s, Agile techniques were introduced as "user requirements" emerged (Avison and Fitzgerald, 2006). As a result, the approaches indicated in Table 1.0 can be classified as waterfall and prototype development versions. When looking at development techniques, it's clear that the focus has shifted from process automation to user engineering over time. Nonetheless, the waterfall technique can be regarded the cornerstone for all methodologies. Even today, prominent approaches such as scrum display characteristics of waterfall and prototype processes. However, there should be a limit to how much attention is paid to the user's needs. It has been determined that excessive software development with gold plating,bells and whistles, or mission/feature/scope/requirement creep may be a concern. The consequences may therefore have a detrimental impact on the schedule, quality, and cost of the system development project (Shmueli and Ronen, 2017). As a result, a balance between process automation and meeting user requirements should be maintained[3]. As a result, developers must make selections about which approach to use in their development process. It's critical to consider the fulfillment of promises (anticipated positive effects) and practices (methodology's vital steps) of the chosen methodology to the developers' requirements. A variety of evaluation mechanisms are available, including cost-benefit analysis, scoring evaluation, feasibility research, value analysis, and multi-objective multi-criterion approaches, among others (Mohagheghi, 2008). However, the current study evaluates techniques using a more straightforward approach, examining the simplicity of automating the process as well as user requirements.

C. *User-Centered Design*

Because consumers perceive the user interface to be the system, the development of user interfaces for tools is criticalUser-Centred Design (UCD) is a method of creating a tool from the perspective of how it will be understood and used by a human user. It was first introduced in the 1980s. Users must change their attitudes and behaviors in order to learn the software. When it comes to UCD, though, the software is geared to help potential users[4] with their current attitudes and behaviors. It accomplishes this by putting users at the center of the design process from the planning and design of system requirements to the implementation and testing of the product. As a result, we have a tool that is effective, rewarding, and easy to use. Baek et al., 2008; Abras et al., 2004). The other two terms associated with the UCD are user experience design (UXD) and usability. UXD must do research to gain a better understanding of the users. User observations, interviews, and various approaches are used in the study to capture the users' emotions, motives, and underlying conceptions and beliefs. The information will then be utilized to create user interfaces that correspond with and support user behavior. The interactive user experience linked with a user interface is measured by usability. It is a test of user-friendliness, or the ability to understand and use something quickly.The usability measures are used to assess a user's ability to use a product without assistance. A variety of methodologies, including as heuristics, cognitive walkthroughs, formal usability inspections, Pluralistic walkthroughs, and others, are available for evaluation. The basic goal of usability testing is to identify and fix user-friendliness issues before releasing the final product. Nielsen and Molich, 1990; Nielsen, 2012, 1994) ("Introduction to User-Centered Design," 2017; Nielsen and Molich, 1990; Nielsen, 2012, 1994)

TABLE 1.0 SYSTEM DEVELOPMENT METHODOLOGIES

No	*Methodology*	*Focus On*	*Suitability for HydroGIS tool*
1	Waterfall	Process	Process automation is best, but it's tough to include user requirements later.

No	*Methodology*	*Focus On*	*Suitability for HydroGIS tool*
2	Iterative and incremental	Process	Shorter waterfall steps, but more challenging to include user needs.
3	Spiral	Process	Based on risk reduction
4	Prototyping	User	Best for user-friendly development and workflow effects
5	Rapid application development	User	Based on meeting the requirements, a time boxing strategy is used.
6	V-Model	Process	Based on the testing
7	Cleanroom	User	Iterations with box structure
8	Behaviour driven development	User	User behaviour centric
9	Joint Application Development	Developer	Developer-centric is not so much a development process as it is a tool.
10	Scrum	User	An adaptable strategy focused on the meeting of user requirements
11	Crystal Methods Methodology	User	Develop depending on the developer's abilities rather than the user's needs.
12	Rational Unified Process	User	Iterative evolution of a complex system
13	Feature-driven development	Process	Less sophisticated processes are found in the bigger project development process.
14	Agile Development	User	A adaptive approach based on the user requirement satisfaction

III. Methodology

A. *Overall methodology*

A thorough literature review is conducted to identify existing system development techniques. The tool's requirements, such as user needs, data needs, process / computation needs, and technological needs, are then identified. It then divides those requirements into numerous categories. Once all of the separate features have been automated, integrate them and test them for integration, system, and acceptability until they pass.

B. *Process development using the waterfall method*

Throughout the calculation process automation, it discovered the required hydrological models and calculation sequences. The method was then manually repeated, and all final and interim results were recorded on excel sheets. This method can be considered of as a phase in waterfall development that involves evaluating the manual system and finding bottlenecks in the existing system. To determine the amount of the detention storage, the concept of inflow hydrograph attenuation is used.

C. *Using a user-centric tool in conjunction with a process-centric algorithm*

It was decided to stop modifying the prototype once the user friendliness had reached an acceptable level of 70%, as well as the need to incorporate the procedure for future assessments. The process automation, on the other hand, will continue until it is 100 percent accurate against the recoded set of outcomes. The user interfaces associated to the process codes, which were produced using the same languages on the same platforms, were created once the process correctness was satisfied.

IV. Discussion

The purpose of this research was to create process codes while creating the model. When a model had to be perfect, the endeavor ran into problems with time-consuming reengineering and re-coding. It realized that it would have to wait until the model development was finished before starting to code the processes and computations. The effort then moved on to user interface construction and usability testing without devoting any time to model calibration. The method is a creative application of existing techniques. The work did not include the creation of a user interface with process automation in software modules that may be changed at any time to a prototype. Even the model's creators agree that the modules are completed correctly.

V. Conclusion

The hybrid development technique, UcT.PcD, is revealed to be a blend of waterfall, prototype, and repetitive development methodologies. This was a two-phased development approach that used waterfall development to automate the engineering process and prototype - repeated development methodologies to develop the user interfaces. Initially. Then, according to Pradeep and Wijesekara (2017), combining software development approaches can result in the intended output if the developers are aware of the desired outcomes. As a result, when compared to current studies, it is obvious that software life cycle models are used with Process-Centric Development and UserCentric Tool (PCD.UCT) model.

Acknowledgment

This work was supported by Major. R M M Pradeep ,one of the senior lecturers of the Faculty of Computing at General Sir John Kotelawala Defence University.

References

[1] Pradeep, R.M.M., Wijesekara, N.T.S., 2015. Modification of User-Friendliness into a HydroGIS Tool, in 8th International Research Conference 2015. Sir John Kotelawala Defence University, Kandawala, Ratmalana

[2] 2022. [Online]. Available: https://www.researchgate.net/publication/308802359_The_evaluation_of_the_cognitive_learning_process_of_the_Renewed_Bloom_Taxonomy_using_a_web_based_expert_system. [Accessed: 13- Jan- 2022].

[3] H. McLoone, M. Jacobson, C. Hegg and P. Johnson, "User-centered design and evaluation of a next generation fixed-split ergonomic keyboard", 2022. .

[4] [6]Iwmi.cgiar.org, 2022. [Online]. Available: http://www.iwmi.cgiar.org/Publications/IWMI_Research_Reports/PDF/pub013/REPORT13.PDF. [Accessed: 19- Jan- 2022].

[5] Files.isec.pt, 2022. [Online]. Available: https://files.isec.pt/DOCUMENTOS/SERVICOS/BIBLIO/Documentos%20de%20acesso%20remoto/Integrating-GIS-with-hydrological-modeling.pdf. [Accessed: 19- Jan- 2022].

[6] "IOS Press Ebooks - New Trends in Software Methodologies, Tools and Techniques - Proceedings of the seventh SoMeT_08", Ebooks.iospress.nl, 2022.

[7] [Online]. Available: https://ebooks.iospress.nl/volume/new-trends-in-software-methodologies-tools-and-techniques-3. [Accessed: 19- Jan-2022].

[8] Citeseerx.ist.psu.edu, 2022. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.834.5912&rep=rep1&type=pdf. [Accessed: 19- Jan- 2022].

[9] Keerthirathne, W.H., Wijesekara, N.T.S., 2017. Determination of a Design Rainfall Pattern by Comparing with its Effect on Streamflow on Greater Colombo Watershed in Sri Lanka, in: UMCSAWM Water Conference. UNESCO Madanjeet Singh Centre for South Asia Water Management, Moratuwa, pp. 35–40.

[10] Thakuri, P.S., Wijesekara, N.T.S., 2017. Climate Change Impacts and Adaptation Measures for Pahala Divul Wewa, Anuradhapura, Sri Lanka, in: UMCSAWM Water Conference. UNESCO Madanjeet Singh Centre for South Asia Water Management, Moratuwa, pp. 59–62.