IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-4, Issue-4 E-ISSN: 2347-2693

Proposed Quality Paradigm for End User Development

Archana Srivastava [1*], S.K.Singh[2] and Syed Qamar Abbas[3]

[1*] PhD Student, Amity University Lucknow
[2*] Professor, Amity University Lucknow
[3*] Professor & Director, Ambalika Institute of Management & Technology, Lucknow

www.ijcseonline.org

Received: Mar/26/2016 Revised: Apr/04/2016 Accepted: Apr/22/2016 Published: Apr/30/ 2016

Abstract— **The IT industry across the globe has rapidly evolved in recent times. The evolution has been primarily driven by factors like changing regulation policies, progression in information technology, globalization, changing customer demands and business needs, collaborations, mergers & acquisitions. Due to changing demands and business needs of end users, End User computing strategy can be expected to have a positive influence on end user behaviour. End users were significantly more satisfied with applications they can develop themselves. This paper review the concept of end user development, commonly used end user development methodologies and mentions some end user development software. It also proposes the EUD qualities that if inbuilt into the system will enhance the end users inclination towards the software hence will increase the end user satisfaction.**

Keywords- End User Computing (EUC), End User Development (EUD), End User Programming(EUP), End User Software Engineering (EUSE), ubiquity, Programming-by-example, script based creation, block Programming

1. INTRODUCTION

End-User Development (EUD) is defined as a set of methods, techniques, and tools that allow users of software Systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artifact [1]. End users have specific goals in their own domains, which are not related to software development. The end users that we consider here are people who have some basic technological knowledge but are not professional programmers.

End users usually do not have training in professionals programming languages, formal development processes, or modeling and diagramming notations. Moreover, end users often lack the time or motivation to learn these traditional techniques, since end users usually write code in order to achieve a short- or medium-term goal rather than to create a durable software asset that will produce a continuing revenue stream. Consequently, supporting EUD requires providing appropriate tools, social structures, and development processes that are highly usable, quickly learned, and easily integrated into domain practice.

EUD overlaps with two similar concepts, end-user programming and end-user software engineering. End-user programming (EUP) enables end users to create their own programs. This subset of EUD is the most mature from a research and practice perspective, so we focus a later section of this article on that portion of EUD. The difference between EUP and EUD is that EUD methods, techniques, and tools span the entire software development lifecycle, including modifying and extending software, not just the "create" phase.

End-user programming (EUP) can be considered as a way of “programming to achieve the result of a program, rather than the program itself". In EUP, the developer's goal is to actually use the program; this contrasts with professional programming, where the goal is to create a program for other people to use, often in exchange for monetary compensation.

The other related concept overlapping with EUD is end-user software engineering (EUSE). EUSE is a relatively new subset of EUD that began about a decade ago. It aims to address the problem of end users software quality by looking beyond the “create” part of software development. EUP is the “create” part, EUSE involves systematic and disciplined activities that address the software quality such as reusability, efficiency, usability etc. Its emphasis is on the quality of the software end users create, modify, or extend; thus its research focuses on methods, techniques, and tools that promote the quality of such software. This area has arisen because of the ample evidence that the programs end users create are filled with expensive errors

The strongest advantage the end-user developer has is their domain expertise. Because they are domain experts, they see no need for requirements gathering. Tradeoffs in functionality are more readily resolved due to the extensive domain knowledge and first-hand observations of systems limitations. This allows changes to be made faster and to be more readily accepted. Another advantage is that previous solutions can be leveraged to initiate or add needed functionality for new solutions. Most end-user development solutions are created in less than a week, and experience long term use by their team or organization [2].

2. COMMONLY USED END USER DEVELOPMENT METHODOLOGIES

The objective of the end-user programming methods is to bridge the gap between usage and programming of an application [3,4] and often these methods focus more on reusing of legacy software than creating new software components or source code. For example, the simplicity, support for immediate feedback and avoidance of misleading appearances are important in end-user programming tools [4,7,8]. The following briefly introduces end-user programming approaches taken from [4,7,8]:

a) *Programming-by-example—* it includes using a particular instance of execution, input-output relations, or existing programs as basis for creating new programs [11]. For example, modification of a working example speeds up development as it provides stronger scaffolding than writing code from scratch [5].

b) *Visual programming—* it is concerned with replacing the textual programming notation with a graphical one with blocks and connectors [11]. Visual programming concepts and tools assist the user to create small applications on top of their things [6].

c) *Script-based creation—* this approach makes programming easier and more natural for users who want customized applications and are capable of doing basic programming without having to set up e.g., C++ or Java environments [11]. Script languages sacrifice execution efficiency and provide an interpreted development environment, a higher abstraction level for programming than typical system programming languages, and weaker typing than system programming languages [7].

d) *Repository-based creation of applications—* it Supports the reuse of software components. For example, the [8] presents an end-user programming approach for Web applications that consist of a (i) Pattern library, (ii) Pattern language, and (iii) Command language. The Pattern library contains patterns (e.g., for dates, times, phone numbers, email addresses, and URLs), parsers (e.g., an HTML parser), and wrappers for Web sites such as Google or Amazon. The library patterns can be glued together with a pattern language called text constraints, which uses relational operators such as before, after, in, and contains to describe a set of regions in a page.

e) *Tailoring of applications—* can be based on *customization* that modifies the parameters of components; *integration* that creates or modifies assemblies of components; and *extension* that creates new components by writing program code. The direct activation technique also belongs to this category and requires that the tailoring functionality is accessible from the use context when the need for tailoring occurs [9].

f) *Block Programming* (Tempel, 2013): It is a paradigm in which each block is a basic software component, which can be part of another software component. LogoBlocks (Begel, 1996) is one of the origins of this paradigm. It is derived from Logo, and was the inspiration for many block programming environments. Programmers assemble their program using these blocks (McCaffrey, 2006).

g) *Mashups*: It uses modern web technologies to collect information from the web and assemble it in a new single location (Grammel and Storey, 2008).

h) *Domain Specific Language (DSL)*: Systems are written for a specific domain and thus are easy to use by domain experts (Fowler, 2010), (van Deursen, et al., 2000). However, they are expensive to develop and are limited by their functionality. Even though it is easier to develop graphical tools for these languages, some of these systems still require training regarding the syntax. Other systems expose control structures in graphical tools.

i) *Form-Based Systems*: They are another name for spreadsheets. They are the most used form of end-users development (Blackwell, 2006). Spreadsheets enable end-users to instantly review the results in the corresponding cells by writing simple queries.

j) *Scenario Based Creation*: It provides scenario-driven business service assembly software environment that uses encapsulated, iconographic building blocks, each representing a discrete web service component to be executed within a business service to logically depict service processes as well as complex relationships between these processes, their audiences, and means of deployment.

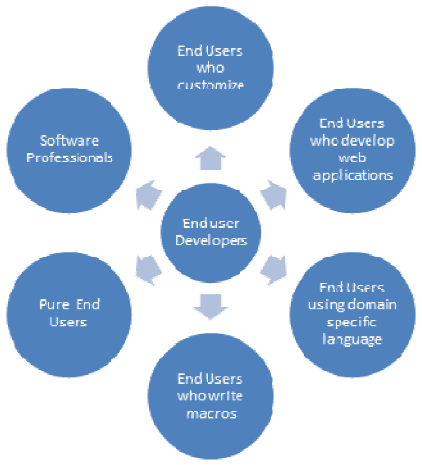
3. COMMON END USER DEVELOPMENT ASPECTS

Enhancing user-participation in the initial design process of IT-systems is one step towards better capturing user requirements. Research is done on providing domain-specific, possibly graphical modeling languages that users find easy to express their requirements in. Such modeling languages are considered an important means to bridge the 'communicational gap' between the technical view of the software professionals and the domain-expert view of the end-users [13].

All end-users perform different programming tasks; there exist a vast, heterogeneous pool of end-users who are likely to benefit from a diversity of tools to support their programming activities. Developing such tools efficiently requires a better characterization of what features are valued by each end-user sub-population. What quality attributes will be needed by specific end-users.

Despite the potential benefits to an organization of user development of applications there are many risks associated with it that may lead to potentially dysfunctional consequences for the organization's activities. These risks result from a potential decrease in quality and control as individuals who have little or no formal information systems training increasingly take responsibility for developing and implementing systems of their own making [17].

4. CATEGORY OF END USERS


End Users who customize
Software Professionals
End Users who develop web applications
End user Developers
Pure End Users
End Users using domain specific language
End Users who write macros

a) *Pure End Users:* The end users are the individuals who use the software product after it has been fully developed and marketed.

b) *End Users who write Macros:* Macros are useful if end users want to create their own custom macros. These macros can be to perform specific actions, apply custom formatting and much more. Web macros give web browser users ways to "program" tedious tasks, allowing those tasks to be repeated more quickly and reliably than when performed by hand.

c) *End Users using domain specific language*: Some domain-specific languages expand over time to include full-featured programming tools, which further complicates the question of whether a language is domain-specific or not. domain-specific languages which are embedded into user applications and which are (1) used to execute code that is written by users of the application, (2) dynamically generated by the application, or (3) both.

d) *End Users who develop web applications*: The ubiquity of the World Wide Web and the resultant ease of publishing content to a huge audience has been an important element in the expanding skills and expectations of computer users. Today most web pages are built by end users simply to present information or for creation of interactive web sites or web applications such as online forms, surveys, and databases still require considerable skill in programming and web technology.

e) *End users who customize*: Customization is doing some modification over an existing Application of form as per the client requirement.

f) *Software professionals*: A software professional is a licensed professional engineer who is schooled and skilled in the application of engineering discipline for the creation of software. A software programmer creates the codes that make a program run.

5. COMMONLY USED END USER DEVELOPMENT SOFTWARES

a) WebML [Ceri, 2000] : WebML offers the developer a full scale modeling language that can be used to model a web application end-to-end (content, page flow, database interaction, etc.). Once the model is defined, an application can be generated.

b) Spreadsheets, Spreadsheets are so far the greatest success story of practical application of end-user programming. From the success of the first spreadsheet, VisiCalc, which was published in 1979, spreadsheets have further evolved and spread to be used by millions of users daily.

c) Database Management Systems, A database management program allows end users to enter, update, store, format, and print reports containing information that is stored as a series of records that share a common format in a database.

d) HANDS [18], is a programming environment for children developed within the Natural Programming Project, where usability is the prime design focus.

e) The FreEvolve platform [12], provides an API for integrating tailoring functionality with software components that allows non-programmers to tailor an application by reassembling components at run-time visually [9].

f) ToonTalk , a programming system which is itself a video game, uses a radically different programming model as well as a radical user interface. The crucial problem of generalizing examples gets solved in a simple, almost obvious way -- if you remove detail from a program, it becomes more general.

g) Template-based generation tools [Turau, 2002][Zdun, 2002]: It present more lightweight approaches to using models and templates to construct applications, where the developer designs a set of templates that are employed when creating the content of the pages flowing back to the users.

h) Atsushi Sugiura's Internet Scrapbook automates assembling Web pages from other Web sources, and he also explores Web browsers on small handheld devices.

i) WebSheets [Wolber, 2002], and FlashLight [Rode and Rosson, 2003]: It allow users to develop web applications without writing code. Some of these studies—e.g., CLICK [Rode, 2005]—are web-based, and so users do not need to install anything to start developing.

j) FlashLight [Rode and Rosson, 2003]: It is a Flash-based development environment, introduced the useful concept of programming at runtime.

6. PROPOSED QUALITY PARAMETERS FOR END USER SATISFACTION

End User Satisfaction can be defined as meeting or fulfilling the requirements of the end user. EUD can be seen as an important contribution to create a user-friendly information society where people will be able to easily access information specific to their current context and to their cognitive and physical abilities or disabilities. People will have access to adapt IT-systems to their individual requirements and the design of IT-systems can be made more socially acceptable by collaboratively involving all actors. Apart from empowering individuals to take part in design processes, EUD can also support communities by letting them share experience on how to adapt IT-systems. In particular, communities might share EUD artifacts by way of repositories for reusable and potentially domain-specific components. These repositories will help people in choosing and assembling components appropriate for their requirements by making available the explanations, recommendations and critique of their peers [13].

By incorporating EUD features in the software customers are getting a general purpose software which they can further customize as per their requirements as users are the domain expert of their own areas. End users were significantly more satisfied with applications they had developed themselves and which possess following quality parameters as per their requirements:

1. Features related to coding
   - Content
   - Accuracy of codes
   - Timeliness
   - Creating throw away codes
   - Creating reusable codes
   - Sharing reusable code
   - Easily understandable codes

2. Verification & Validation of codes
   - Inbuilt feedback about the correctness
   - Testable codes
   - Tools for analyzing by debugging
   - Error detection tools

3. User Friendly Interfaces
   - Availability of online help

Self – efficacy: High sense of control over the environment
   - Perceived ease of use: Apart from extrinsic motivation intrinsic motivation (enjoyment) should be present.
   - Ease of Use & Feedback
   - Inbuilt System Assistance for EUP

4. Reliability Features in End User programs
   - Testing
   - Verification & Validation

5. Identify Risk involved in End user development
   - Version control
   - Change Control
   - Data Integrity Control
   - Availability Control

6. Code revision related features
   - Flexible codes
   - Scalability features
   - Ease of Maintenance

7. End User Computing Capability
   - High Computing Capability of End Users
   - Less End User Training & learning Time Constraint

8. Security Issues
   - Security features in codes for more control by end users
   - Authentication features

All the above mentioned features will enhance the end users inclination towards the software hence will increase the end user satisfaction.

7. CONCLUSION

In today's volatile economic conditions, the demands from stakeholders to deliver more value with lesser costs have been escalating every day. All organizations want high level quality products which help increase their business value. The Internet of Things is likely to increase the already large population of end user programmers. The growing number of "smart" devices and possible features suggests that niche programmer communities could potentially form around specific interests and projects. We expect that these communities, like current open source and maker communities, will include programmers of varying skill levels, motivated by the desire to improve products that they use [8].

REFERENCES

[1] H. Lieberman, F. Patern`o, and V.Wulf, Eds., End-User Development, HumanComputer Interaction Series, Springer,NewYork, NY, USA, 2006.

[2] McKendric, J. (2012). "Employees: We'll Build our own Technology Solutions, Thank You". zdnet.com

[3] Mørch, A.I. Tailoring tools for system development. JEUC 1998, 10, 22–29.

[4] Mørch, A.I.; Stevens, G.; Won, M.; Klann, M.; Dittrich, Y.; Wulf, V. Component-based technologies for end-user development. CACM 2004, 47, 59–62.

[5] Hartmann, B.; Wu, L.; Collins, K.; Klemmer, S.R. Programming by a Sample: Rapidly Creating Web Applications with d.mix. In Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, Newport, RI, USA, 7–10 October 2007; pp. 241–250.

[6] Kovatsch, M.; Weiss, M.; Guinard, D. Embedding Internet Technology for Home Automation. In Proceedings of IEEE Conference on Emerging Technologies and Factory Automation (ETFA), Bilbao, Spain, 13–16 September 2010; pp. 1–8.

[7] Ousterhout, J.K. Scripting: Higher level programming for the 21st century. Computer 1998, 31, 23–30.

[8] Miller, R.C. End User Programming for Web Users. In Proceedings of the End User Development Workshop at CHI Conference, Ft. Lauderdale, FL, USA, 5–10 April 2003.

[9] Won, M.; Stiemerling, O.; Wulf, V. Component-based approaches to Tailorable systems. In End User Development; Lieberman, H., Paternò, F., Wulf, V., Eds.; Springer: Dordrecht, The Netherlands, 2006; Volume 9, pp. 115–141.

[10] Honkola, J.; Laine, H.; Brown, R.; Tyrkko, O. Smart-M3 Information Sharing Platform. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Riccione, Italy, 22–25 June 2010; pp. 1041–1046.

[11] Kotkaluoto, S.; Leino, J.; Oulasvirta, A.; Peltonen, P.; Räihä, K.J.; Törmä, S. Review of Service Composition Interfaces; University of Tampere: Tampere, Finland, 2009.

[12] Stiemerling, O. Component-Based Tailorability; University of Bonn: Bonn, Germany, 2000.

[13] EUD-Net's Roadmap to End-User Development, Markus Klann Fraunhofer Institute for Applied Information Technology (FhG/FIT).

[14] Ruby on Rails. http://www.rubyonrails.org, accessed August 2008.

[15] Turau, V. 2002. A Framework for Automatic Generation of Web-based Data Entry Applications Based on XML. Proceedings of the 2002 ACM symposium on Applied computing. Madrid, Spain, pp. 1121--1126.

[16] Shah, S.K. Motivation, governance, and the viability of hybrid forms in open source software development. Management Science 52, 7 (2006), 1000–1014.

[17] Cale, E. G. (1994). Quality issues for end-user developed software. Journal of Systems Management(January), 36-39.

[18] John F. Pane, Brad A. Myers, and Leah B. Miller. "Using HCI Techniques to Design a More Usable Programming System", Proceedings of IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC 2002), pp 198—206, Arlington, VA, September 3-6, 2002.

Authors Profile

Ms. Archana Srivastava has done M.Sc(Maths), MCA, M.Tech(IT) and is working as Asst. Professor in Amity Institute of information Technology, Amity University, Lucknow, India. She is persuing Ph.D. in software engineering. She has 20 years of teaching experience. Her Area of interest includes Software Engineering, Software Project Management, and Software Testing & Quality Assurance.

.

Dr.(Prof) S.K.Singh is a Professor and Programme Director in Amity Institute of Information Technology, Amity University, Lucknow, India. He has done M.Sc(Maths), MCA, M.Tech(IT) and PhD in Applied Computer Science. He has more than 22 years of teaching experience. Till date he has published over 25 research papers in various renowned national and international journals.

Dr. Syed Qamar Abbas completed his Master of Science (MS) from BITS Pilani. His PhD was on computer-oriented study on Queueing models. He has more than 30 years of teaching and research experience in the field of Computer Science and Information Technology. Currently, he is Director of Ambalika Institute of Management and Technology, Lucknow. He is actively involved in academic and research work. Till date he has published over 60 research papers in various renowned national and international journals.