User Centered Method for Enhancing the Adoption of Software Systems in Palestine

Conference Paper · July 2017

DOI: 10.1145/3102304.3102317

CITATIONS
2

READS
161

3 authors:

Yousef-Awwad Daraghmi
Palestine Technical University- Kadoorie
42 PUBLICATIONS **521** CITATIONS
SEE PROFILE

Motaz Daadoo
Palestine Technical University- Kadoorie
23 PUBLICATIONS **107** CITATIONS
SEE PROFILE

Derar Eleyan
Palestine Technical University- Kadoorie
57 PUBLICATIONS **573** CITATIONS
SEE PROFILE

User Centered Method for Enhancing the Adoption of Software Systems in Palestine

Yousef-Awwad Daraghmi
Computer Systems Engineering
Palestine Technical University
y.awwad@ptuk.edu.ps

Motaz Daadoo
Computer Systems Engineering
Palestine Technical University
m.daadoo@ptuk.edu.ps

Derar Eleyan
Applied Computing
Palestine Technical University
Birzeit University
d.eleyan@ptuk.edu.ps

ABSTRACT

Software systems play major roles in improving people work and life quality. Developing countries as Palestine should adopt these systems to cope with the development and improve services provided to people. However, the adoption of software systems in Palestine has not reached the expectations because these systems may not fit with the Palestinian work environment. Software development methods influence the adoption of software systems, but the methods used in Palestine were mostly engineered for developed countries causing the produced software to be inadequate. Therefore, to determine which software method works better for Palestine, we studied existing systems and identified the factors influencing the acceptance of software systems. Based on these factors, we proposed a software development method that fits with end-user workflow and work environment to build usable software. We integrated three system development methods so that we can consolidate the advantages of each one and overcome their drawbacks. After that, we examined the usefulness of the proposed method empirically by developing real life software system. The research contributes to the software engineering field with an integrated software development method that focuses on users and usability for developing accepted software systems.

CCS CONCEPTS

Software and its engineering → Software creation and management; software development methods.

KEYWORDS

Software methods, Participatory Design, Rapid Application Development, Usability engineering.

ACM Reference format

Y. Daraghmi, M. Daadoo, D. Eleyan. 2017. User Centered Method for Enhancing the Adoption of Software Systems in Palestine. In proceeding of the International Conference on Future Networks and Distributed Systems (ICFNDS), Cambridge, UK, July 2017. 10 pages. ISBN 978-1-4503-4844-7

1 INTRODUCTION

Software systems aim to improve people life by, for example, increasing the efficiency of daily work, supporting people cooperative activities, monitoring and controlling people environments, and improving people interactions together or with the physical world. In Palestine which is a developing country, the private and the public sectors have called for software systems to improve the quality of services provided for citizens and consequently improve life quality and consequently software systems are increasingly being introduced to the Palestinian life in many sectors such as e-business, e-learning, and e-health [1]–[3]. However, the adoption of software systems has not reached the expectations due to several factors [4]. One important factor is that the software development process plays a major role in the degree of acceptance of new software products [5], [6]. Software development methods may decrease or increase the resistance to change from traditional systems to new software systems. Most software development methods used so far in the development of software applications in Palestine appear inadequate in supporting all requirements and workflow activities in real life [7]. Consequently, the developed software does not fit with user needs and therefore users remain reluctant to adopt it to their work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
© 2017 Copyright held by the owner/author(s), ICFNDS '17,July 19-20, 2017, Cambridge, United Kingdom. Isbn: 978-1-4503-4844-7/17/07. . . $15.00 DOI: http://dx.doi.org/10.1145/3102304.3102317

The literature about software development in Palestine is very little and does not provide enough knowledge about how software companies build their systems. A study showed that software companies do not fully specific software practices which makes the testing of the final system impossible or in most cases the final system fails [8]. in some known cases, some methods used in software development in Palestine ranges from old to new methods. Old methods such as the water fall method focus on the user at the beginning of development and do not consider the emerging requirements at late stages [9]. Therefore, recent methods have put the user at the center of development and allowed the continuous collection of requirements through iterative cyclic design stages. The most well-known user-centered design methods are the Participatory Design (PD) and the Rapid Application Development (RAD). The PD method is criticized as it takes long time to develop complete software, it does not enable the production of ready to use software, and it does not provide specific rules for the design [10], [11]. An approach to overcome the PD's limit is integrating it with RAD to speed up the development. However, RAD also has shortcoming summarized by its inability to engage user in all design stages, lacks of rules to set up plans and teams, and depriving designers of breakdown points needed to review the design and write full documentations [9], [10]. To overcome RAD limitations, it should be integrated with usability testing during all stages of design.

In this research, we propose a software design method by integrating PD, RAD and usability testing. This method was driven after identifying all factors limiting the adoption of software systems in Palestine. These factors were identified after conducting a case study on three existed systems. After that, we used the results of the case studies to drive our software development method. The proposed method was used empirically to develop a real application called E-Auction. Then, another case study is conducted to examine the proposed method usefulness and its ability to improve the adoption.

Qualitative analyses were performed through two focus group meetings and interviews with expert users. Also, quantitative analyses were used through System Usability Scale (SUS) questionnaire. Results from these evaluations indicate that the E-Auction is usable and well received by users and provides several benefits such as time savings in terms of the users' documentation task, high accuracy and better management. This also indicates that the proposed method used to develop the E-Auction software is useful and increases the adoption of software systems in Palestine. However, we can not argue that the proposed method is better than other software development methods as this generalization requires wide range of testing.

2 RELATED METHODS

2.1 Participatory Design

Participatory design is defined as "a rich diversity of theories, practices, analyses, and actions, with the goal of working directly with users (and other stakeholders) in the design of social systems including computer systems that are part of human work" [12]. PD started in Scandinavia in 1970s then moved to North America [13]. Participatory design creates a more intimate social atmosphere between engineers and end users. It also allows the integration with formative evaluation methods to assess user satisfaction during the development process [14].

PD is considered one of the User Centered Design (UCD) models that focuses on user needs and allows end users to participate in the design so that they will be satisfied with the new system [15], [16]. User Centered Design (UCD) is a design philosophy popularly recommended as it constantly addresses the requirements of end-users throughout the development process [17]. UCD was derived from Human Computer Interaction (HCI) theory that focuses on user needs and provides flexible models to gather user requirements and produce a suitable interface [18]. UCD as a process and as a philosophy; " It is a philosophy that places the person (as opposed to the 'thing') at the center; it is a process that focuses on cognitive factors (such as perception, memory, learning, problem-solving, etc.) as they come into play during people's interactions with things" [19]. It is also stated that UCD is a design philosophy that addresses user requirements throughout the development process and helps develop interactive software systems [17].

The PD approach lacks the aspects that make it relevant to professional system development. Although there is a considerable amount of empirical research dependent on it, it has not achieved a sufficient influence on information system applications. Therefore it requires integration with other practical development methods such as RAD [10]. PD is also criticized because it is imprecise and does not provide a fully specified design process [11]. Also, PD does not enable production of ready-to-use system because it put emphasis on the early systems development phases [20]. It is also shown that PD design sessions take a long time [20]. Usually, participants need to meet several times for negotiations, planning and designing. Further, the use of technology is not clear in the design which makes users bored and unhappy because they prefer a hands-on experience. In other words users like to get their hands dirty and they prefer practical tasks [20].

2.2 Rapid Application Development

New software frameworks aim to enable developers to build high quality applications rapidly in order to consider emerging requirements of new businesses [9]. Old models such as waterfall models have been used to develop computer applications. In these methods, the development process takes a long time and requirements change before the system is completed, resulting in inadequate or even unusable systems [9]. RAD was developed to overcome these problems by developing systems quickly and enabling the developer to identify the critical requirements iteratively.

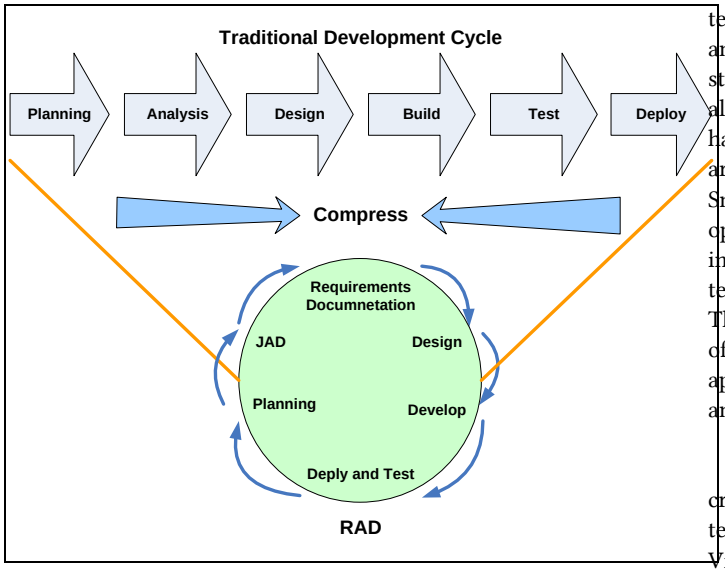

Figure 1: RAD versus traditional software development methods.

RAD is a software methodology that was introduced in 1991 by James Martin. Martin defines RAD as "a development life cycle designed to give much faster development and higher quality results than the traditional life cycle". It is designed to take maximum advantage of powerful development software that has evolved recently [21]. Software development in RAD may consist of three or four stages. The four-stage cycle consists of requirements planning, user design, construction, and cutover, while in the three-stage cycle, requirements planning and user design are consolidated into one iterative activity. Further, Systems in RAD are developed quickly in a series of increments [21].

RAD is a methodology for compressing the traditional life cycle stages; analysis, design, build, and test phases into a series of short, iterative development cycles [21]. Iteration allows for effectiveness and self-correction. Iterative development means creating increasingly functional versions of a system in short development cycles. Each version is revised with the end user to produce requirements that feed the next version. The process is repeated until all functionalities have been developed. This iterative development helps designer consider emerging requirements.

2.2.1 RAD Dimensions

In RAD, software projects usually operate along four dimensions [22]. First, people who may perform their tasks quickly or slowly. Second, process which may include difficult tasks and consume much time. Third, product which may be defined in a way that it builds itself or it may require the best effort from the employees. Fourth, technology which may assist the development or it may require more effort. People in RAD are the team which works on the project. RAD uses hybrid small teams that consist of about 6 people, including both developers and full-time users of the system plus anyone else from the stakeholders. In RAD, developers talk and listen and customers also talk and listen. Developers chosen for RAD teams should have high skills and be multi-talented and motivated people who are analysts, designers and programmers all rolled into one. Small teams, combined with iterative development cycles optimize speed, unity of purpose, effective and cooperative informal communication, and simple project management. The team should organize the process and avoid repeating the work. They should use development fundamentals to assure the quality of the product and manage the risks. They also should use appropriate and effective technology based on the product size and characteristics [22].

RAD uses specialized tools that support visual development, creation of working prototypes using multiple languages, teamwork and collaboration, and use of reusable components. Visual basic is one of the new programming languages that supports Object Oriented Programming (OOP) and enables developers to build new systems rapidly. It is shown that to write a code, visual basic roughly costs 25% of the time needed to write the same code in another language such as C [22].

2.2.2 RAD Drawbacks

Studies stated that although RAD enables user involvement in the design which gives a higher possibility of meeting user requirements, it does not give the user the chance to participate and be fully engaged in the design [9], [10]. In RAD, 'user involvement' deals with technical issues as a major issue while 'user participation' explores technical issues plus social and job considerations. Further, other shortcomings are summarized in the inability of RAD to advise on how to build a project plan, how to set up a team and how to manage user-developer relations, insufficient design place because in most cases the design process in RAD usually takes place in a room instead of the work field [9], [10]. Also, RAD does not propose a breakdown that gives developers the time to discuss issues with users, or scenarios that enable a developer to explain his understanding of how users work and what their requirements are. RAD also uses the technique of demonstrating prototyping rather than cooperative prototyping and does not provide specifications documentation which makes evaluation difficult [9], [10].

2.3 Usability Testing

Usability testing refers to the evaluation of information systems that involves testing of participants who are representative of the target user population, as they perform representative tasks using an information technology in a particular context [23], [24]. In software applications, usability of an interface system is a measure of the amount of effort an end user must expend to learn its use and then use it to perform tasks successfully [25]. Usability engineering field has emerged from the integration of evaluation methods used in the study of Human Computer

Interaction (HCI) aimed at providing practical feedback into design of computer systems and user interfaces [26].

Usability engineering is integrated with system development methods mainly rapid and iterative methods [27]. This integration enables users to see a prototype earlier in development process and they can assess their usability and functionality. If such assessment indicates that changes are needed, a further cycle of design and testing is initiated. This process continues until the system becomes acceptable to users and shows the desired functionality. Information from usability testing regarding user problems, preferences, suggestions and work practices is applied not only towards the end of system development to ensure that systems are effective, efficient and acceptable, but also throughout the development cycle to ensure that the development process leads to effective products.

There are a number of types of usability tests, based on when in the development life cycle they are applied [28]:

- **Exploratory Tests**: conducted early in the system development cycle to test preliminary design concepts using prototypes or storyboards.
- **Prototype tests:** used during requirements gathering.
- **Assessment tests:** conducted early or midway through the development cycle to provide iterative feedback into evolving design of prototypes or systems.
- **Validation tests:** conducted to ensure that completed software products are acceptable regarding predefined acceptance measures.
- **Comparison tests:** conducted at any stage to compare design alternatives or possible solutions (e.g., initial screen layouts or design metaphors).

3 THE FIRST CASE STUDY: PRE-DEVELOPMENT ANALYSIS

At the beginning of the research, a case study consisting of three subcases on three stakeholders was conducted to know the factors affecting software adoption in Palestine. The research then was slightly shifted to focus on the development process because of the strong concentration of stakeholders on the development procedures. We noticed that stakeholders focused on key points such as requirements, workflow, user interface, testing, usability and methodologies. This forced us to focus more on the development process and to conduct another case study that examines system development and design processes as shown in section 5.

3.1 Subcase 1: Jericho vegetable and fruit market

This market is for buying vegetables and fruits from farmers in Jericho and selling them to traders from other cities in Palestine. The market consists of many shops containing auctioneer, accountant and managers. The work in this market starts in the early morning until late night because shops owners spend much time in managing farmers and traders accounts, writing bills and reports by traditional ways. Accuracy is another issue because any mistake in the calculation may cause a big financial problem. According to shop owners, in the past, three electronic systems were used to make their jobs easy, and save their effort and time. But, the three systems failed because they did not satisfy the market requirements. The software did not consider the specific workflow in the market and introduced a new workflow that was rejected by shops owners. Shops owners also complained about the difficulty in using the software commenting that people in the market do not have high computer skills. Shops owners demanded several times for new software that meets their needs and has the same workflow, but the software took much time to be delivered and new requirements have emerged making the new software useless for them.

3.2 Subcase 2: stores management software

This study took place in a vocational training center which aims to provide vocational training for the Palestinian young men and women in order to become productive in the society. The center has large stores for storing equipment, tools and furniture which requires database management software. The canter demands for a reliable software and a customized database tools in order to fit with the way that the center uses in managing the stores.

3.3 Subcase 3: pharmacy electronic system

This study was conducted in three pharmacies to find the best software for managing pharmacies in Palestine. One of the pharmacies is using an electronic pharmaceutical system while the others are thinking to buy the same system but they are reluctant due to several factors. The conclusion of this study is that all pharmacies in Palestine have the same workflow and process of buying, storing and selling medicine and cosmetics. It is also concluded that the electronic system has introduced new workflow to the pharmacy which makes pharmaceutics reluctant to but it. Pharmaceutics also commented that an electronic system that can be customized to their work will be better. In addition to that, the pharmaceutical electronic system was not easy to use by employees in the pharmacies.

The final results of the first case study show that there are low adoption of software systems and high resistance to change from old systems to new systems. The results highlight the main factors that influence software adoption in Palestine. These factors can be categorized as:

1- Development factors which we are going to deal with in this research and that include:
   - Requirements: any software must meet the functional, nonfunctional and organizational requirements in order to be fully adopted. In other words, the new software must maintain the workflow; be reliable; easy to use; and achieves accuracy. People in the case studies state that the best way for them to express what they need and reflect their requirements is to participate in the developments process. They mark that they cannot remember everything at one time and it easy for them to work iteratively.

- Slow delivery: this was highlighted mainly in the vegetable and fruit market because of the continuously changing and emerging requirements. That means there will be new requirements that are not considered in the new software product.
- Usability factors such as efficiency, effectiveness, reliability, ease of use, and learning time.

2- Non-development factors:

- Computer skills: in general end users have low computer skills and although some of them had training computer courses, they still prefer simple and easy to use software applications.
- Cost: end users find new software price high compared to their income.
- Environment factors such as language, religion and politics.

4 THE PROPOSED METHOD

Based on the results of the pre-development case studies, we choose RAD and the PD in order to enable fast delivery and meet all requirements. We also integrate RAD and PD with the usability testing methods. This integration of the methods is to let them fit with work environment and to consolidate the advantages of each method and overcome its drawbacks. Evidence on such integration is shown in related studies as PD and RAD was integrated together achieving better system design and management [10]. This approach is used to develop new systems rapidly and to identify the major requirements by enabling end users to participate in the design process. Another example is User Centered RAD (UCRAD) which is an integration of RAD with user centered techniques. It is used to develop successful applications that have good functionality, simple features and usable interface [29].

4.1 Internal PD and RAD details

Here we discuss the processes of both PD and RAD to show how these methods can be integrated.

4.1.1 PD Process:

The PD model consists of three modules including different techniques to interact with users. The main technique is design meetings in which developers and stakeholders meet to plan,

design and evaluate. Also, Interviews are used as individual interviews or group interviews for eliciting requirements or evaluating designs. Further, observation mainly participatory observations are used to identify how users work. These modules are [20]:

Module 1: Pre-Design (Project Plan, Contract and Design Group)

Participants: Systems developers and stakeholders representing the organization.

Prerequisites: A clearly stated mission and allocation of resources, acceptance of PD principles and end-user participation.

Activities: Pre-design scheduling by setting project goals, project planning, allocating representative users and establishing efficient principles for the design process.

Outcome: Preliminary project plan, project contract and established design group.

Module 2: (Requirements Analyses, Design, and architecture). It is the core of PD and consists of three sub-modules:

Sub-module 2A: (Organization Analysis)

Participants: Systems developers, users representing the organizational.

Prerequisites: Final project plan and project contract.

Activities: analyses of the organization's work and design verification proceed during the PD work. Then, external data collection starts during the design process. Organizational analysis and early design practices are performed in an iterative process. Documentation is used to keep the organizational focus. In parallel, design decisions made in the group are verified at higher organizational levels.

Outcome: design decision protocol, updated documentation concerning organization analysis.

Sub-module 2B: (Information Systems Analysis)

Participants: Systems developers and user representatives.

Prerequisites: documentation including issues resulting from at least one round of organizational analysis and verification.

Activities: prototyping system architecture is performed iteratively, based on the organizational analysis. Then, the prototype is demonstrated during design meetings. The implementation of the prototype is updated between meetings. The documentation is continually updated as prototyping progresses.

Outcome: Updated version of the prototype and documentation concerning system architecture.

Sub-module 2C: (Technology Analysis)

Participants: Systems developers, user representatives and engineers.

Prerequisites: documentation concerning system architecture and the prototype.

Activities: The outcomes of the information system analysis and prototyping activities determine the technologies that can be used to implement the system. These technologies are evaluated during design meetings. The documentation is updated based on the results of these evaluations.

Outcome: Updated documentation based on evaluation results.

Module 3: Post-design (Specification, and Full Implementation)

Participants: Systems developers, engineers, user representatives, external stakeholders from organization

Prerequisites: a verified design decision protocol and a completed documentation of the design.

Activities: implementation of prototype version and completion of the requirements specification. Formative evaluation is performed to validate contract goals.

Outcome: Requirements specification, prototype and evaluation report.

4.1.2 RAD Process:

RAD has four stages which can be summarized as [21], [22]:

- The requirement planning stage in which the requirements analyses team meets with the client users to discover the initial requirements. The requirements planning stage should result in a list of entities as well as action diagrams that define the interactions between processes and data elements and should take between one and four weeks.
- The user design stage in which the analysis team meets with end users during JAD workshops. In the workshops the analysis team revises the requirements and finds out more details, develops the entities collected in the requirements planning into a data model and diagrams, develops test plans, and creates layouts for essential parts of the system. In order to keep development iterations as short as possible, and to gain the maximum benefit of RAD's agile nature, core requirements should be identified and targeted for the initial prototype, and secondary requirements should be identified and targeted for future development iterations. The design team should "design for the change" by identifying areas that may change, developing a change plan and using Object Oriented Programming (OOP) design.
- The construction phase in which the design team develops the application in iterative cycles of development, testing, requirements refining, and development again, until the application is complete. The development team should convert the data model that was developed during the user design stage into a functional prototype. Once the prototype has been developed within its time box, the construction team tests the initial prototype using test scripts developed during the user design stage. Finally, the construction team, design team, and customer meet in focus group meetings to review the application and determine the requirements for the next iteration.
- The implementation stage or deployment stage which integrates the new application components and applies them into the field. The development team prepares data and implements interfaces. The design team trains the users while the users perform acceptance testing. The design team helps the users transfer from their old procedures to new ones. This involves trouble shooting after the deployment, and identifying system's drawbacks. The amount of time required to complete the implementation stage varies with the project. After that, the team deploys the new system and organizes and stores project assets such as reusable code components, project plan, project management plan, and test plan.

During these stages, meeting sessions which is called Joint Application Development (JAD) is the main technique used to interact with users, gather and analyze data.

4.2 Integration Details:

The PD modules and RAD stages can overlap ensuring user participation in every iterative stage from the beginning of the development to the end, and ensures that the language used fit with user experience. The PD is used because it organizes the participation by highlighting the important points in the process and it saves time by scheduling all activities. It also allows the participants to "get their hands dirty" in the design process. Further, it provides full documentation about each iteration input and output. In this way, PD overcomes RAD drawbacks. Then, RAD ensures time saving feature and enables PD to be used in a practical and technical situation.

The five-phase formative evaluation (five tests) can be injected easily in each stage in the previous model according to the functionality of each test and each stage. The five-phase formative evaluation is used in each stage to validate the results against user requirements and system specifications which enhances systems usability. The five tests have the following functionalities:

- testing the preliminary design concepts, plans and schedules;
- testing the prototype and validating the initial user requirements;
- assessing the implemented version and feeding back user comments;
- validating the complete system against requirements specifications;
- Finding alternative solutions.

The stages of the proposed method are shown in Figs. 2, 3 and 4. The first stage is called the predesign stage and it integrates PD module 1 and RAD prototyping including the requirement analyses phase and the construction phase. To ensure that the prototype meets user requirements, the exploratory, usability and comparison tests are performed during this stage. Fig. 2 shows the first stage which takes place at the start of the project.

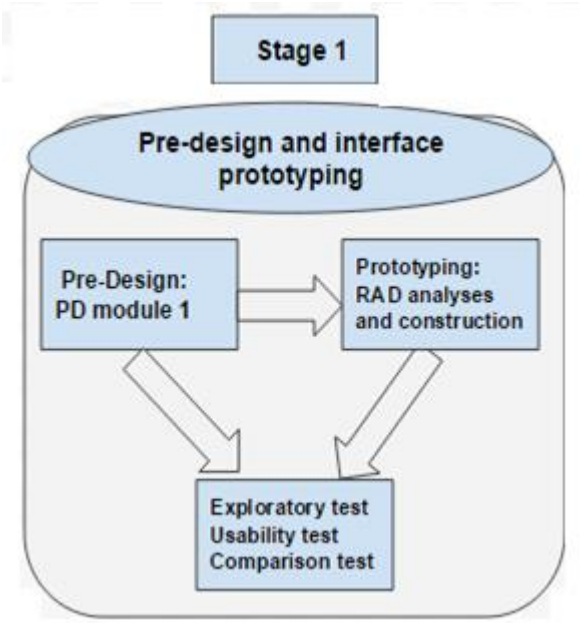

Figure 2: Stage 1 which takes place at the start of the project. This stage integrates PD module 1 and RAD prototyping accompanied with three tests to ensure user satisfaction.

Then, the output of the first stage goes to the second stage. In this stage, the design process and building the final architecture of the system start which force the programmers to begin implementing the final system. To achieve good results, this stage integrates PD module 2 and RAD implementation phase. Also, usability tests are performed here to ensure satisfactory output. Fig. 3 shows the flow of the second stage.

The output of stage 2 goes to stage 3 which is the final stage and produces the final system. The team aims to deploy the final system at the end of this stage therefore they perform full implementation of the system. To achieve successful deployment, this stage integrates PD module 3 and RAD deployment phase as shown in Fig. 4. Also, this stage contains usability tests mainly the validation test.

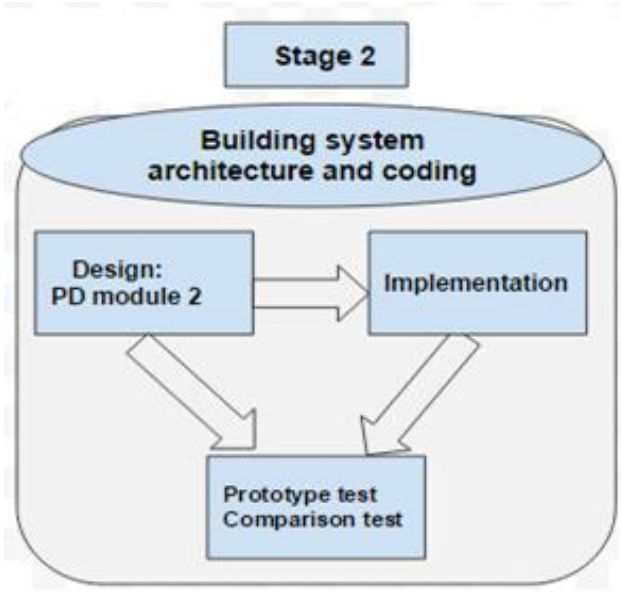

Figure 3: Stage 2 integrates PD module 2, RAD implementation phase and two usability tests to ensure user satisfaction.

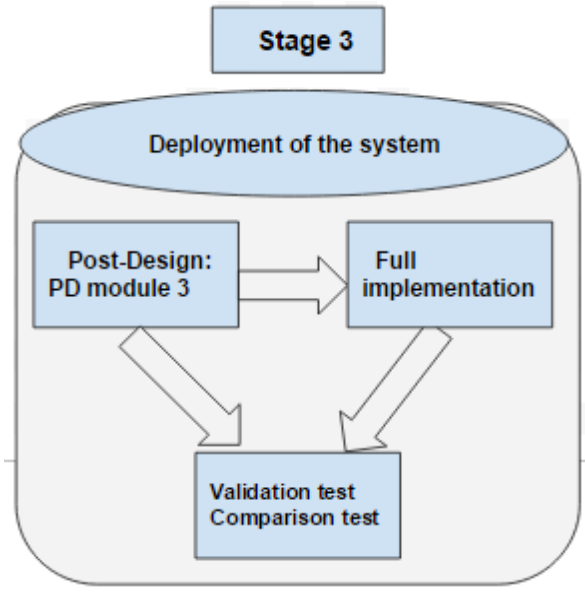

Figure 4: Stage 3 which takes place at the end of the project. This stage integrates PD module 3 and RAD deployment phase (full implementation). To validate the final system efficiently, the validation and comparison tests are performed.

Fig. 5 shows the three stages and how the output of each stage is taken to another stage. All steps and procedures

explained earlier can be used for software development and this can be seen in the following section where detailed explanations show the development of a real system.

5 THE SECOND CASE STUDY

5.1 Software development

In this section, we show how the integrated method was tested empirically. This case study focuses on the development process of software systems. Therefore, we formed a development team incorporating software developers and users from the vegetable and fruit market in Jericho. We studied all steps and procedures the team used to build new software for the vegetable and fruit market. The team named the new software E-Auction. The team used visual basic and .NET framework to develop the system.

From the first case study, we have developed an understanding of what would be an acceptable software product for the end users of E-Auction. This understanding led the team to establish certain key design goals for E-Auction as described below:

- Since the vegetable and fruit market users are constantly stressed in their work, the intended E-Auction needs to ensure that it does not add additional burden to them. This goal led the team to introduce design elements that closely matched workflow and the user needs. For example, individual needs such as font colors, font sizes; screen sizes and colors, and team needs related to the accountability practices that are applied for the correctness of processes.
- E-Auction must have high usability measured through ease of use, ease of learning and perceived gains in using it.

After identifying the key design goals, the team faced the challenge of practicing the integrated method to achieve these goals. Therefore, the team established an approach that focuses on the following:

- end-users from the point of view of their work and their environment,
- developing a rapid prototype with focused functionalities and involving a typical end-user in conceptual design stage and then in testing the prototype,
- providing the flexibility to welcome the changes in requirements identified by users and documenting them to help later in field testing,
- Ensuring minimal risk for all involved stakeholders by assessing the impacts of any changes made on other stakeholders.

The whole project spanned over 9 months. We were fortunate to have three participating groups from the vegetable market in Jericho city. The teams participating with the iterative development process during that period acting as a typical user. There were several small iterations in the development in which the groups looked through our screen designs and made suggestions. When the small iterations reached a reasonable milestone, it was called a stage in the development process.

5.1.1 E-Auction Design—Stage 1

The first stage of the design process was lengthy and took 5 months to fully complete. The extended timeframe was the result of many initial decisions that should lead to producing a reasonably good working prototype at the end of this stage or what is called throwaway prototype. In this stage, the development team arranged the project plan and schedules according to PD module 1. Then, the team started eliciting user requirements by collecting the forms used in the market. The team preserved the look and feel of these forms as much as possible. The team also analysed the process models with emphasis on the tasks of the employees in the market. From this exercise, the team deduced that due to the constant and dynamic operations, the

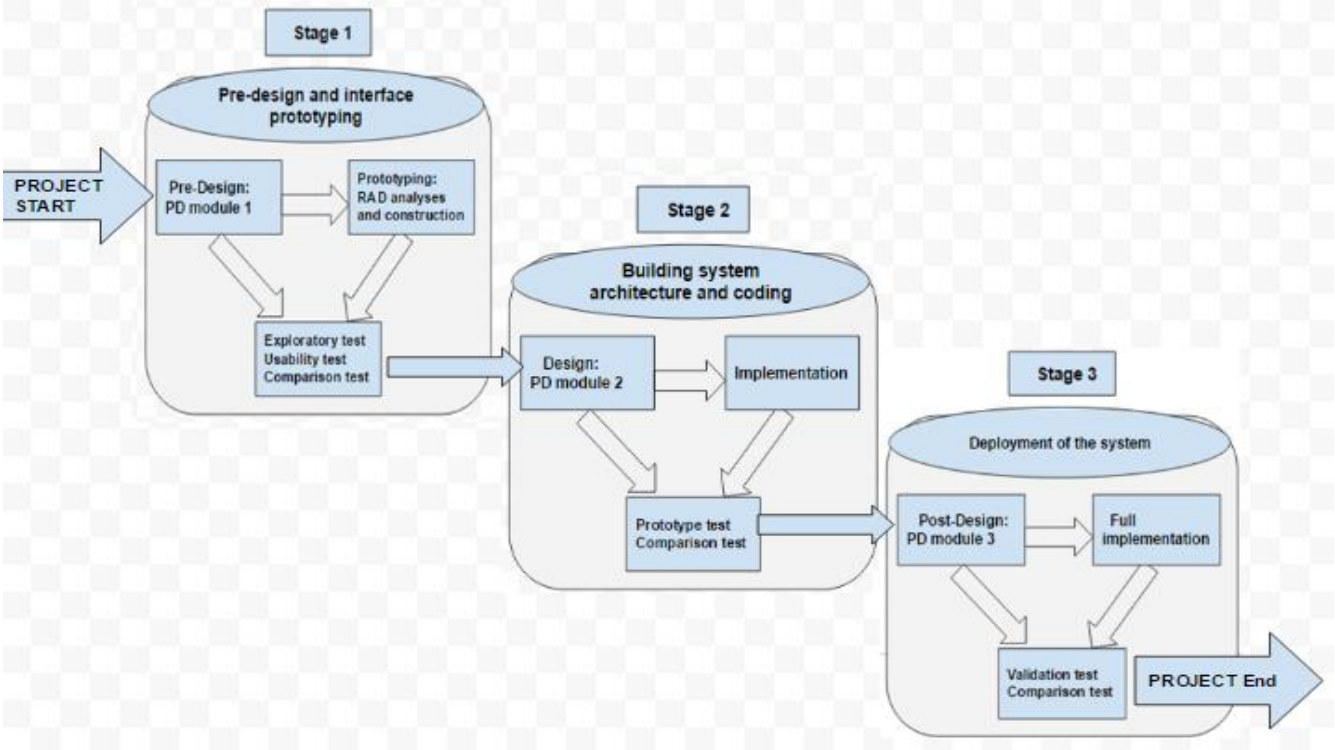

Figure 5: The complete proposed method and the overall interaction between the three stages.

team should focus on providing smooth transitions between traditional and electronic activities. The team also kept on assessing user needs and exploring the available solutions using the exploratory test, prototype test and comparison test. These tests led to the development of the main screen in the user interface that was unique to E-Auction and, at a glance, provided the essential data and information. This screen allowed the user to quickly access the information they needed. Also, the information contained in this screen is automatically updated by the system.

Input: initial requirements and paper-based forms (bills, weekly and monthly reports, receipts etc).

Processes: planning, prototyping, assessing user needs, and building evolutionary prototype, formative evaluation (exploratory tests, usability tests, and comparison tests).

Output: evolutionary prototype (main screen user interface) and initial requirements documentation, basic architecture and basic system design.

5.1.2 E-Auction Design—Stage 2

The goal of this stage is to implement a ready to use version of the system. In this stage, the team redesigned E-Auction based on the experiences gained from Stage-1. The team involved the three groups in the testing. The requirements were then refined incrementally. Two more stakeholders other than the three groups were also introduced to the project. Testing with additional users revealed additional requirements. Also, as the three groups became comfortable with E-Auction and found what computers can do for them, they requested additional features that would reduce the possibility of errors in their jobs. One such example is the addition of the empty vegetable boxes input and output panel. This feature displays the net empty boxes in each shop and in the farms. During this stage, the team benefited more from the participatory design approach as the three groups got more involved. While the PD module 2 helped the team to directly involve the thought process of users in the design and creation of a better interface of the E-Auction, RAD allowed the team to quickly incorporate the desired changes by employing

short design cycles and allowing for subsequent iterations and refinements with end-users.

The second stage in the iterative process required many meetings with the users to test the prototype and obtain an evaluation regarding the suitability of the current User Interface (UI). At the end of this stage, a field demonstration was given to all groups and open suggestions were invited that allowed the team to compare all designs and find the best one. As such, the group participation allowed to converging towards the final design of the E-Auction.

Input: initial requirements documentation, an evolutionary prototype and basic architecture.

Process: updating the requirements, design and architecture, coding the system, and prototype evaluation.

Output: implemented version (complete user interface and back end database), a complete architecture and system design.

5.1.3 E-Auction Design—Stage 3

This is the final stage of the E-Auction design process focusing on the full implementation which includes data management and populating the databases with real data. This stage also involved designing the required database schemas to support the real-field testing of the E-Auction at the market. Once again, the team iterated this enhanced version of the E-Auction with the three groups according to PD module 3 and validation tests.

Input: requirements documentation and the implemented version with complete architecture.

Process: full implementation and validation tests.

Output: deployed version (ready to use E-Auction).

5.2 Field Test and Usability Evaluation

At the end of the second case study, a real field test was performed to ensure the usability of the E-Auction. We created a set of test cases that were used in the iterative testing. The testing users would perform those tasks with a talk aloud events in the presence of the developer and the software engineer. We conducted our field test involving actual end-users using live data in a real market environment but for a fixed short period of time. Although the field test was for a fixed duration, it was meant to increase the likelihood of the users accepting a transition to a computer-based system. During this time, all manual data and paper-based activities were suspended and were replaced by the computerized support through E-Auction. To accommodate the transition from the manual paper-based practice to the E-Auction, each shop was equipped with a desktop computer and a printer. At the end of the field tests, we asked all end users including users who participated in the development process to fill System Usability Scale (SUS) questionnaire.

Then, we carried out qualitative evaluation of usability of the E-Auction focusing on the main categories: effectiveness, efficiency and acceptability and sub categories: reliability, ease of use, functional completeness, navigation, learning time, and look-and-feel. We conducted three focus group meetings with members from the field tests. In these meetings, we encouraged the users to speak freely about their experiences, both positive and negative, using the E-Auction software. Their statements were recorded. These statements talked about two aspects: useful features or the features in E-Auction that helped them to do their job better in some sense; and desired features or the new features that they wished to be incorporated when E-Auction goes into full use.

6 RESULTS

The formative or iterative usability tests performed during the development stages reduced the amount of effort we put on the final usability tests because end users were familiar with the system and their comments were fed back to the design during the development stages.

We analyzed the collected data qualitatively according usability categories such as ease of use, response time, colours and resolutions, consistency of operations, navigation, learning time, and look-and-feel. The focused groups' response was positive towards all usability issues. The focused groups stated that the use of E-Auction adds several benefits to the market as follows:

- Saving a considerable amount of time in their overall work and movement. In the E-Auction, switching between farmers and traders' files to access information was seamless to locate the actual physical files. Users have become able to print the bills and any documents easily and in a short time.
- Achieving better accuracy and reducing calculations errors.
- According to users, E-Auction is easy to learn and then easy use. The incorporation of all groups has considerably reduced the navigational burden on the users in the use of E-Auction.
- The work flow in E-Auction is the same as the flow in the real life. Users do not find any strange object. All paper forms were exactly transformed to electronic forms.
- Users can easily, constantly and instantly monitor and stay up to-date with important and relevant information related to farmers, traders, and stores.
- The system is more reliable than the traditional system that was used before. E-Auction makes information available at any time and users perform functions continuously without interruption.

We also use a quantitative method to analyze the SUS questionnaire. We found that the average score for users who participated in the development process is 89% while the average score for users who did not participate is 76.2%. The two scores reflect good usability and show high quality of E-Auction.

7 CONCLUSIONS

This research contributes to the software engineering field with an integrated software development method that focuses on users and usability. As we have noted from this particular research experience, involving real users in the iterative development and refinement of software products intended to support real practices would certainly have a positive impact on

user acceptance of software systems. However, for such an approach to be practicable, we need the cooperation and full participation of well-selected end-users. Our experience in the development of E-Auction was, in a small scale, somewhat like a personal software process involving a very small team. Finding and involving end-users was non-trivial and time consuming but the benefits are clear. In real life and large scale systems, such an approach to software development may demand more cost and effort, but we trust that this additional effort will pay dividends in the life-cycle of the software product.

REFERENCES

[1] ANERA, "Software Engineering and IT Training for Palestinians," 2017. [Online]. Available: http://www.anera.org/category/education/information-technology/. [Accessed: 08-Apr-2017].

[2] Paltrade, "THE STATE OF PALESTINE NATIONAL EXPORT STRATEGY INFORMATION AND COMMUNICATION TECHNOLOGY (ICT) SERVICES," 2014.

[3] Ministry of Telecom and Information Technology, "The Palestinain Stratigic Plan for the ICT Sector for the years 2017-2022," 2017.

[4] T. P. Trust, "The ICT sector in the Palestinian Territory," 2012.

[5] C. Boivie, J. Aborg, Persson, and M. Lofberg, "Why usability gets lost or usability in in-house software development," *Interact. Comput.*, vol. 15, pp. 623–639, 2003.

[6] J. Grudin, "Interactive systems: bridging the gaps between developers and users," *IEEE Comput.*, vol. 24, no. 4, pp. 59–69, 1991.

[7] A. Isawi and B. Abdulhaq, "Software Development Process Improvement for Small Palestinian Software Development Companies," An Najah National University, 2011.

[8] M. Alnajjar1 and S. S. A. Naser, "EVALUATING SOFTWARE ENGINEERING PRACTICES IN PALESTINE," *Int. J. Soft Comput. Math. Control*, vol. 4, no. 1, pp. 35–47, 2015.

[9] I. Sommerville, "Software Engineering," *(7th ed), Addison Wesley*, 2004.

[10] P. Beynon-Davies and S. Holmes, "Integrating Rapid Application Development and Participatory Design," *IEE Proc.-Softw*, vol. 145, no. 4, pp. 105–112, 1998.

[11] S. Pilemalm, P.-O. Lindella, N. Hallberga, and H. Eriksson, "Integrating the Rational Unified Process and Participatory Design for Development of Socio-Technical Systems: a User Participative Approach," *Des. Stud.*, vol. 28, no. 3, pp. 263–288, 2007.

[12] S. Kuhn and M. J. Muller, "Participatory Design," *CACM*, vol. 36, no. 41, pp. 26–28, 1993.

[13] M. J. Muller, "PICTIVE -An Exploration in Participatory Design," *Proc. SIGCHI ACM Conf. Hum. Factors Comput. Syst. Reach. through Technol.*, pp. 225–231, 1991.

[14] C. Weng, D. W. McDonald, D. Sparks, J. McCoy, and J. H. Gennari, "Participatory Design of a Collaborative Clinical Trial Protocol Writing System," *Int. J. Med. Informatics*, vol. 76, no. 1, pp. S245–S251, 2006.

[15] D. Schuler and A. Namioka, "Participatory Design: Principles and Practices," *Lawrence Erlbaum Assoc. New Jersey, USA*, 1993.

[16] A. Waller, V. Franklin, C. Pagliari, and S. Greene, "Participatory Design of a Text Message Scheduling System to Support Young People with Diabetes," *Health Informatics J.*, vol. 12, no. 4, 2006.

[17] D. A. Norman, "User Centered System Design: New Perspectives on Human Computer Interaction.," *Hilsdale, NJ Lawrance Erlbaum Assoc.*, 1986.

[18] A. Seffah, J. Gulliksen, and M. C. Desmarais, "Human-Centered Software Engineering," *Human Computer Interaction Series*, vol. 8. Springer, 2005.

[19] R. Katz-Haas, " User-Centered Design and Web Development," *Usability Interface*, vol. 5, no. 1, 1998.

[20] S. Pilemalm and T. Timpka, "Third Generation Participatory Design in Health Informatics—Making User Participation Applicable to Large-Scale Information System Projects," *J. Biomed. Inform.*, vol. 41, no. 2, pp. 327–339, 2008.

[21] J. Martin, "Rapid Application Development," *Macmillan, New York*, 1991.

[22] S. McConnell, *Rapid Development: Taming Wild Software Schedules*. Redmond, Washington: Microsoft Press, 1996.

[23] J. Nielsen, *Usability engineering*. New York: Academic Press, 1993.

[24] J. Rubin, *Handbook of usability testing: how to plan, design and conduct effective tests*. New York: Wiley, 1994.

[25] ISO 16982, "Ergonomics of Human–System Interaction—Usability Methods Supporting Human-Centered Design," vol. 16982. 2002.

[26] john M. Carroll, *Human-Computer Interaction in the New Mellinnium* . New York: Addison-Wesley, 2002.

[27] B. Bygstada, G. Ghineaa, and E. Brevika, "Software Development Methods and Usability: Perspectives from a Survey in the Software Industry in Norway ," *Interact. Comput.*, vol. 20, no. 3, pp. 375–385, 2008.

[28] A. W. Kushniruk and V. L. Patel, "Cognitive and Usability Engineering Methods for the Evaluation of Clinical Information Systems," *J. Biomed. Inform.*, vol. 37, no. 1, pp. 56–76, 2004.

[29] E. Lank, K. Withee, L. Schile, and T. Parker, "User Centred Rapid Application Development," *Guelfi N Savidis A, eds. Rise. Springer-Verlag Berlin Heidelb.*, pp. 34–49, 2006.