International Journal of Advances in Computer Science & Its Applications
Volume 6 : Issue 2 [ISSN 2250-3765]
Publication Date : 31 August, 2016

A Suggested Sustainability Pattern for Online Social Networks:The Facebook as a Case Study

Fawaz A. Al Zaghoul, Noor S. Al-Anbaki, Mohammed T. Al-Baldawi

Abstract—**The survival of a software product for a long term is a challenging issue. Software doesn't die but it becomes obsolete.**

The aim of this paper is to propose a design pattern that could help maintaining the sustainability of software products. The proposed pattern could be used in the architectural engineering phase of developing an online social network system to help maintaining its sustainability.

We built our work based on an agreed upon definition of software sustainability from the software engineering perspective, we discussed the main factors that lead to maintain a sustainable software product, and introduced the risks that might lead to software death.

Finally, a planning phase was introduced to help analyzing the need for a sustainable software and evaluating it economically.

It is highly recommended to use the proposed pattern and to consider the proposed factors due to their comprehensiveness and flexibility.

Keywords—**sustainability, online social networks, pattern.**

I. Introduction

In the last decade, software sustainability has become one of the key challenges in the world of computational science and engineering after it has been suggested as a new research field in 2010.

The term "Sustainable" is essentially associated with ecology. The Oxford Dictionary defines sustainability as 'the quality of being sustained' that is capable of being endured and capable of being maintained" [18]. From a software perspective, this definition refers to time or longevity and maintenance of a software product.

Dr. Fawaz A. Al Zaghoul
King Abdullah II School for Information Technology / University of Jordan
Jordan

Noor S. Al-Anbaki
King Abdullah II School for Information Technology / University of Jordan
Jordan

Mohammed T. Al-Baldawi
Architecture and Design College/Al-Ahliyya Amman University
Jordan

In 1983, sustainability was first referred to in a conference conclusion: 'Humanity has the ability to make development sustainable to ensure that it meets the needs of the present without compromising the ability of future generations to meet their needs' [8]. This definition was proposed by the Bruntland commission, a group assigned to create a "global agenda for change" by the General Assembly of the United Nations.

This definition opened the door to a high level vision for both researchers and consumers to look forward to achieve this goal.

In 2004, a triple perspective of sustainability has been adopted which considers sustainability to include three components: environment, society and economy [9] as shown in Figure 1. Where economy is a subsystem of human society, which is itself a subsystem of the environment. This view is considered after ecological economist Herman Daly asked, "what use is a sawmill without a forest?" [3].

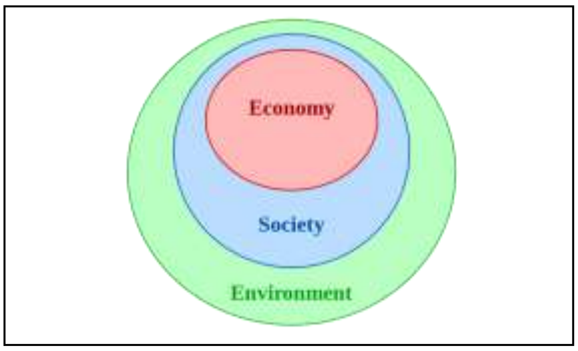


Figure 1. Triple Perspective of Sustainability

This new concept was quickly adopted by engineers who focused on how to maintain sustainability in their constructions. Structural integrity, weather ability, energy efficiency; ventilation, cleaning access, durability and maintainability were the main factors that they adopt [6], [20], [23]. They even used smart materials i.e. materials that transform energy from one type to another to preserve long lasting structures [2].

The concept of sustainability motivated software engineers as well to adopt this vision in their products. Although several attempts have been made, but there is still no agreed upon definition of software sustainability.

"The ability to modify a software system based on customer needs and deploy these modifications" [19]. This definition implies that the software should be maintainable.

"Software you use today will be available - and continue to be improved and supported - in the future" [21]. This definition is proposed by the Software Sustainability Institute that was founded in 2010. Which implies that the software should be available, extensible and dependable.

While the software engineering research community considered sustainable (Green) Software to be the software product that reverses the loss of environmental resources and employ sustainable development towards conservation and management of resources [25].

Despite these different points of view, along this paper it would be acceptable that sustainable software refers to "long-living software systems that could be evolved over its lifetime and has the highest capability to survive".

This paper highlights the most important factors that make the software product sustainable and would discuss the risks that face the software against being sustainable. The work in this paper would be mainly maintained in the architectural design phase by proposing a sustainability pattern for online social networks through an adopted case study: the Facebook system model.

The paper is organized as follow: Section 2 presents related literature. Section 3 demonstrates the factors that make software sustainable. The risks would be investigated in section 4. Section 5 introduces how to plan and preserve a sustainable software. Section 6 explains the case study. Section 7 clarifies the proposed pattern. The conclusion is given in section 8.

II. Related Work

Although many articles in the literature have focused on the field of sustainable products, but only some of them highlighted the area of software sustainability from the view of point that has been adopted in the scope of this paper.

Mahaux & Canon in [17] argued that designing sustainable software should be integrated in the requirements engineering process as requirement engineers can also reduce the relative impact of the development and disposal phase of software engineering: by enabling software to last longer, which in turn preserve software fitness that is related to qualities such as reliability, adaptability, maintainability or context-awareness of software.

While Koziolek in [15] through his systematic review concluded that the quality of software architectures determines sustainability to a large extent. He categorized more than 40 architecture-level metrics to measure the sustainability of software architecture both during early design using scenarios and during evolution using scenarios and metrics.

He proposed two definitions of sustainable software. In the first definition, he defined sustainable software as 'a software-intensive system that operates for more than 15 years'. In the second definition, he defined sustainable software as 'a long-living software system which can be cost-efficiently maintained and evolved over its entire life-cycle'.

Durdik and others in [7] based their work on a real fact. That is, unfortunately, in many software development projects, sustainability is treated as an afterthought, as developers are driven by time-to-market pressure and are often not educated to apply sustainability-improving techniques.

They created a catalog of "software sustainability guidelines" to support project managers, software architects, and developers during system design, development, operation, and maintenance.

III. Sustainability Factors

Several researchers argued that sustainability should be considered as non-functional requirements, that according to [22] can be defined as constraints on the services or functions offered by the system, they apply to the system as a whole, rather than individual system features or service.

Taina in [13] suggests that a sustainable software product should have the following properties:

Fit for purpose: defines how software helps its system reach its goal;

Reduction: defines how software supports its system in waste reduction;

Beauty: defines the value of the system in sustainable development.

While in [4] software sustainability is defined as a composite, non-functional requirement which is 'a measure of a systems extensibility, interoperability, maintainability, portability, reusability, scalability, and usability'.

In [12] sustainability is assumed to comprise of maintainability, modifiability, portability and evolvability.

According to the definition adopted in this paper, it is agreed that sustainability is one of the non-functional requirements of the software product, as it appears clearly after the components of the system interact with each other and with the environment.

On the other hand, it is important to consider it in the early stages of software development if it is desirable to build a long living software product.

The main factors i.e. software system attributes that enhance software sustainability, that are proposed in this paper are shown in Table1 along with their definitions.

Although the output of the software engineering process is not as tangible as the architecture or construction engineering deliverables but we argue that the success of software systems is related to these factors.

Along this paper, we are going to consider one of the most well-known online social networks system model, the Facebook. And we will select the main factors that can make such a software product survive would be highlighted through this example.

International Journal of Advances in Computer Science & Its Applications
Volume 6 : Issue 2 [ISSN 2250-3765]
Publication Date : 31 August, 2016

TABLE I. SOFTWARE SUSTAINABILITY FACTORS

Sustainability Factor	Definition
Usability & HCI	How easy it is for users to use the product effectively and efficiently. Does the software have an attractive user interface?
Accessibility & Availability	To what extent is the software accessibe?
Portability	How easy it is to use the software on other platforms?
Maintainability	How easy it is to locate and fix an error in the software; and is the software supportable? And what is the cost of maintenance?
Scalability	The extent to which software can grow and still supply the users with the required functionality.
Technology	The extent to which software can be integrated to evolved new technologies.
Requirement Volatility	The extent to which software can be adapted to requirements change without risking its performance.
Self-Funding	From an economical perspective, can the software product fund itself in order to enhance its evolution?
Development Tools	What are the tools used to develop the software? Is the software developed using a programming language that is still applicable and supported? Does it relay on reused components that can follow its evolution?
Language Selection	Does the software support one local language or it enhances translation to some other international languages?
Environment	Does the software properly consider external rules and regulations?
Documentation	The extent of formal documentation that is dynamically updated each time the software evolved along system's lifetime.
Reputation of Software Vendor	Determines the likelihood of long term customer satisfaction with vendors and their product [12].
Stakeholder Behavior	How stakeholders react to market, interact with key organizations and individuals within the project system's environment? And how this could control the influence of their reaction will carry and affect their decisions to increase the chances for project success [5].
Risk	Both sustainability and risk management are referencing future consequences of decisions taken at early stages of software development. Avoiding or mitigating risk sources e.g. security vulnerability will highly enhance sustainability.
User Feedback	Does the product consider user's opinions and satisfaction?

IV. Sustainability Risks

What makes the system die! So many factors can make the system open to a high risk and threat its survival.

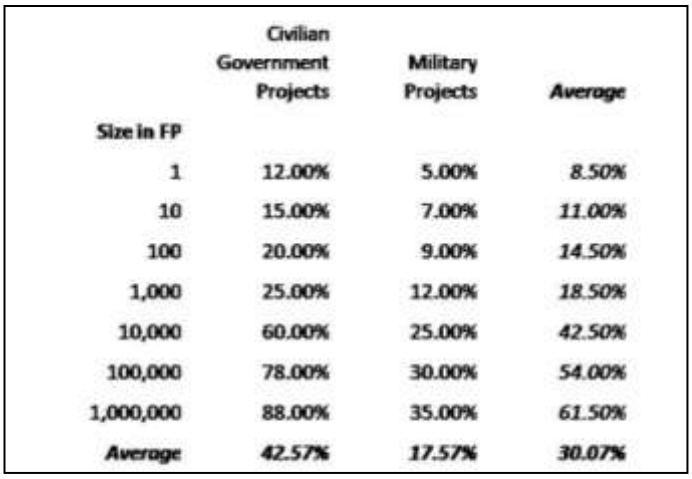
Sustainability risk is estimated in [1] by considering nine indicators: lifetime in production, lifetime, competence risk, technology evolution risk, risk of changing business model, market risk, lifetime certainty, complexity risk and technology evolution fitness. The output of the assessment is an indication of the expected lifetime of the technology's sustainability.

If the necessary factors that lead to develop a sustainable software product were not considered properly, we might reach that the software cannot be used for a long time.

Lack of accurate documentation for example may cause inconsistencies with source code which make it difficult, costly and time consuming for developers to understand the code, trace errors and develop new versions of the software product. An example of the Aging Software Applications with structural decay, error prone modules, and geriatric problems can be shown in Figure 2.

Where the lack of tractability for the up to date user or market requirements may highly threat the product life time.

The most important risk that is considered in this paper is the one that threat the software economically. Without a continuous fund the system lifetime cannot be maintained, the evolution and development of the software to meet the market needs cannot be achieved and there would be no ability to integrate the system to fit with new technologies.



Size in FP	Civilian Government Projects	Military Projects	Average
1	12.00%	5.00%	8.50%
10	15.00%	7.00%	11.00%
100	20.00%	9.00%	14.50%
1,000	25.00%	12.00%	18.50%
10,000	60.00%	25.00%	42.50%
100,000	78.00%	30.00%	54.00%
1,000,000	88.00%	35.00%	61.50%
Average	42.57%	17.57%	30.07%

Figure 2. Estimated percent of Aging Software Applications with Sustainability Problems. Capers Jones © 2008

V. Sustainability Planning and Reservation

Opting to develop a sustainable software product is an important issue that needs an extra planning phase that could be added after or within the requirement engineering phase.

Lack of detailed planning may lead to extra cost in the far future. A study of U.S Army and Air Force Maintenance Centers showed that over 50% of their software-related effort was spent on Maintenance and Sustaining Engineering [24]. As shown in Figure 3.

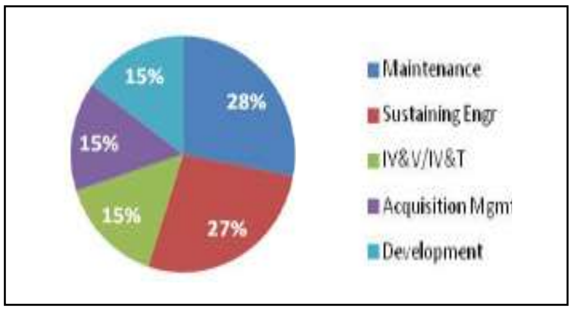


Figure 3. Software Effort/Activity of U.S Army and Air Force Maintenance Centers

International Journal of Advances in Computer Science & Its Applications
Volume 6 : Issue 2 [ISSN 2250-3765]
Publication Date : 31 August, 2016

There should be justified reasons that motivate the development of a sustainable software product. These reasons might be one or all of the following:

- Administrative decisions.
- Competence and presence in emerging markets.
- Financial issues.
- Users' satisfaction.
- Software product ultimate goals.
- Policies.

The decision to develop a sustainable software product should be evaluated against the company's existing resources and funding plan. As owning sustainable software imposes the following costs [14]:

- *Initial product developing cost.*
- *Product enhancement cost:* to enhance the product with new features during its lifetime.
- *Product maintenance cost.*
- *Customer support cost.*
- *Product redevelopment cost:* necessary for removing or replacing error-prone modules.
-

After cost evaluation is performed through the requirement engineering phase, a suitable software process model should be considered.

A waterfall based model would be most suitable as it contains a lot of documentation that is necessary to support a sustainable software product. Such a model should be modified to enhance incremental delivery of multiple releases along the software life time.

Figure 4 shows the sustainable software decision phase.

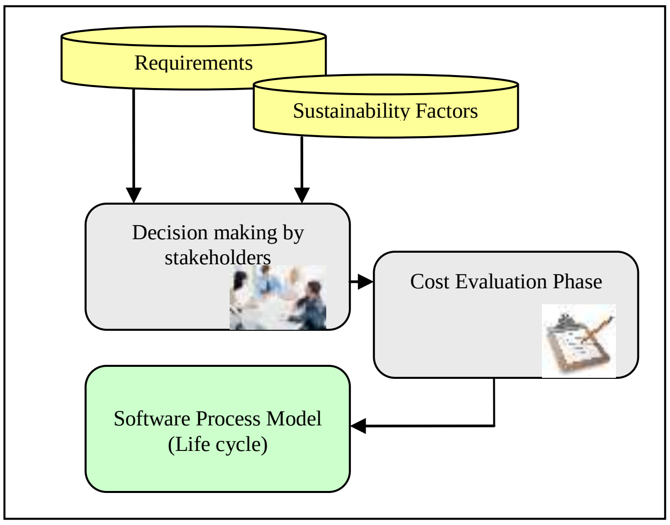

Requirements
Sustainability Factors
Decision making by stakeholders
Cost Evaluation Phase
Software Process Model (Life cycle)

Figure 4. Sustainable Software Decision Phase

VI. A Case Study: The Facebook System Model

Facebook is an online social networking service launched in February 2004 [10]. It has attracted millions of users in the last decade. According to [11] the number of active users from 2008 to 2015 can be shown in Figure 5.

Although several other social networks available like hi5, Myspace and others, that perform mostly the same functionality of Facebook, but until now Facebook is the largest online social network.

We can argue that if the Facebook continue evolved the way it is developing, it would be one of the largest sustainable software products.

The sustainability of its system model comes from several factors, but on top of them is that Facebook is self-funded. The revenue generated by advertising is substantial and is used to fund the product [16].

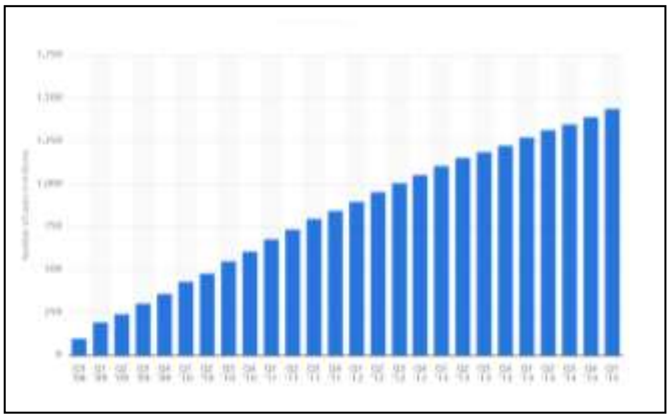


Figure 5. Facebook Active Users between 2008-2015

Other factors that support the Facebook sustainability are its enhancement by:

- Usability: it is easy to use the product by users of any age. And it has a simple and attractive interface.
- Efficiency: the Facebook secret for image compression made it a suitable choice for many users. It also supports for accessing a lot of people with one post rather than trying to contact them individually.
- Accessibility: Facebook supports ease of integration with other services, which lead to log in to several sites using the Facebook account.
- Portability: it is easy to use the Facebook from any other platforms especially from your mobile phone.
- Maintainability: the software is supported by a software supporting team that may fix any problems within a maximum of few days.
- Scalability: Facebook has no restrictions on number of active users joined and available.
- Technology: Facebook is highly attached to the new technologies available.

- Requirement Volatility: using the Facebook for several years made it clear that much functionality has been added and enhanced and it was able to follow up with the change in requirements.

Although for supporting many factors that tend to make the Facebook system model sustainable, but it could be in a risk of survival due to lack of fund.

In [16] the authors evaluate the implications of fake user profiles on Facebook. They proved by experiments that Facebook does not provide enough methods to reliably detect and eliminate fake profiles.

This security issue has several consequences. First, if the user is not satisfied with the level of privacy achieved, he could leave Facebook. Eliminating unwanted behaviors on the other hand, is mandatory.

Fake profiles may greatly affect the survival of such systems. Advertising companies, the main source of fund, target real users, such artificial identities harm the overall business model.

This is a subtle condition, that if not achieved will threat the sustainability of Facebook in the future.

For such systems it is a wise decision to check periodically the number of active users per month, visitors per month, new subscribers and the average number of clicks per month.

It is obligatory to check the income vs. the money it takes, per month, to pay for salaries, overhead, marketing, etc.

In online social network systems that are 24/7 in touch with users it is a wise decision to make estimations of these measures and how they are going to change in the future. And to compare these estimations with the real statistics for each month or year to measure the success of the sustainability plan.

VII. A Sustainability Pattern for Online Social Networks

In this section, a pattern is proposed for maintaining sustainability in social networking services.

As defined in [22] a pattern is "*a description of the problem and the essence of its solution, so that the solution may be reused in different settings*". Figure 6 illustrates the proposed pattern.

VIII. Conclusion and Future Work

The proposed sustainability factors introduce a significant contribution on the available research in the literature for planning and preserving a sustainable software product.

These factors cover different perspectives and take into account many dimensions that support sustainability and refer to the risk sources that threat sustainability of any software system.

The chosen case study showed that it is highly recommended to consider all or some of these factors when developing long-term software depending on the scope of the system itself and its need.

For future work, we are going to investigate a way of building a tool that can measure the sustainability of software products. Some collected software systems would be chosen for research and comparisons should be made to evaluate the measuring tool.

This new research area is quite useful as long as the world is moving toward preserving resources and enhancing green technologies.

Software systems should be considered one of the valuable assets that should be preserved as well.

Pattern name: Sustainability Pattern for Social Networks

Description: Support sustainability of the software product and enhance its survival.

Problem description: In many situations, the software product and especially the online social services is threatened by multiple risk factors that may hinder its development. The main risk factor is economic, if the software cannot be funded it will not survives.
This pattern may be used to provide recommendation in situations where the software product is in touch with users and affect their daily life. For an online social network to provide guidance for the network to preserve its users and sources of fund, and as a result maintain its sustainability.

Solution description: The requirement engineer should consider the issue of software sustainability from early stages in order to take the main sustainability factors in consideration before beginning the architectural design phase. The stakeholders should take concrete decisions of enhancing usability, accessibility, maintainability, privacy and the integration with the developing technology and market needs without jeopardizing the sources of funds especially when the software product is self-funded. Enhancing self-funding through several techniques like advertisement, gaming is a major factor to successfully build sustainable software.

Consequences: building a sustainable software product is a challenging issue, as sustainability factors may override each other. At the same time it might require a lot of efforts and resources to maintain sustainability.
Not all software products need to be sustainable as some of them is dedicated to a certain process and might require an overall change rather than trying to keep them as they are.

Figure 6. Sustainability Pattern for Social Networks

References

[1] A. Jansen, A. Wall and R. Weiss. "TechSuRe: A method for assessing technology sustainability in long lived software intensive systems," SEAA 2011: Proceedings of the 37th EUROMICRO Conference on

software engineering and advanced applications, Oulu, Finland, August 30 - September 2, 2011.

[2] Al-Baldawi, M. T. (2015). Application of Smart Materials in the Interior Design of Smart Houses. Civil and Environmental Research, 7(2), 1-15.

[3] Anderberg, S. (1998). "Industrial metabolism and linkages between economics, ethics, and the environment". Ecological Economics 24: 311–320.

[4] C. C. Venters et. al., "The blind men and the elephant: Towards an empirical evaluation framework for software sustainability,"WSSSPE'1: workshop on sustainable software for science: practice and experiences, SC'13, 17 November 2013, Denver, CO, USA.

[5] Carroll, A., & Buchholtz, A. (2014). Business and society: Ethics, sustainability, and stakeholder management. Cengage Learning.

[6] Diesendorf, M. (2007). Greenhouse solutions with sustainable energy (p. 252). University of New South Wales Press.

[7] Durdik, Z., Klatt, B., Koziolek, H., Krogmann, K., Stammel, J., & Weiss, R. (2012, September). Sustainability guidelines for long-living software systems. In Software Maintenance (ICSM), 2012 28th IEEE International Conference on(pp. 517-526). IEEE.

[8] G. H. Brundtland, and United Nations World Commission on Environment and Development, "Report of the World Commission on Environment and Development: Our Common Future." United Nations, 1987.

[9] H. Svendrup and M. G. E. Svenson. "Defining the concept of sustainability – a matter of systems thinking and applied systems analysis," Systems Approaches and their Application, pp: 143- 164, 2004.

[10] http://en.wikipedia.org. Last visited on 1st, Jan. 2016.

[11] http://www.statista.com. Last visited on 1st, Jan. 2016.

[12] Hoxmeier, J. A. (2000). Software preannouncements and their impact on customers' perceptions and vendor reputation. Journal of Management Information Systems, 17(1), 115-139.

[13] J. Taina. "Good, bad, and beautiful software - In search of green software quality factors," CEPIS UPGRADE, volume XII, pp. 22–27, 2011.

[14] Jones, C. Geriatric Issues of Aging Software, Crosstalk, The Journal of Defense Software Engineering, Vol. 20, No. 12, USAF Software Technology Support Center, Hill AFB, UT, December 2007.

[15] Koziolek, H. (2011, June). Sustainability evaluation of software architectures: a systematic review. In Proceedings of the joint ACM SIGSOFT conference--QoSA and ACM SIGSOFT symposium--ISARCS on Quality of software architectures--QoSA and architecting critical systems--ISARCS (pp. 3-12). ACM.

[16] Krombholz, K., Merkl, D., & Weippl, E. (2012). Fake identities in social media: a case study on the sustainability of the Facebook business model. Journal of Service Science Research, 4(2), 175-212.

[17] Mahaux, M., & Canon, C. (2012). Integrating the complexity of sustainability in requirements engineering. In First international workshop on Requirements for Sustainable Systems.

[18] Oxford English Dictionary. 2012. Oxford Dictionaries.

[19] R.C. Seacord, J. Elm, W. Goethert, G. A. Lewis, D. Plakosh, J. Robert, L. Wrage, and M. Lindvall. "Measuring software sustainability," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, USA, 2003.

[20] Samuel, A., & Weir, J. Introduction to Engineering Design: Modelling, Synthesis and Problem Solving Strategies, 1999.

[21] Software Sustainability Institute. Available: http://www.software.ac.uk/about. Last visited on 1st, Jan. 2016.

[22] Sommerville, I. (2011). Software Engineering. International computer science series.

[23] Sullivan, A. (2003). Economics: Principles in action.

[24] U.S. Army, Army Software Operations, Maintenance and Sustainment Study Overview, Software Sustainment Collaborators Workshop, September 14, 2011.

[25] United Nations, "Agenda 21." United Nations Conference on Environment and Development (UNCED), Jan-1992.

About Author (s):

Prof. Fawaz A. Al Zaghoul, received his PhD in Software Engineering on 1987 from Liverpool University/UK. He is currently a full Professor of SE at the University of Jordan. His research interests are Software Systems Development Methodologies, Software Engineering, Algorithms reengineering, Computer Networks Application, Database Applications, and Quality Metrics.

Noor S. Al-Anbaki is a PhD Candidate in the Department of Computer Science at the University of Jordan. She received her B.Sc. and M.Sc. degrees in Computer and Software Engineering from the University of Technology / Iraq in 2004 and 2008 respectively. Her research interests are software enhancement, mobile computing, distributed database and context aware systems.

Dr. Mohammed T. Al-Baldawi received his PhD degree in Interior Design from the University of Baghdad in 2001. He is currently an Associate Professor at Al-Ahliyya Amman University. His research interests are sustainable interior design, smart houses design, commercial design, photography, scientific research and labor market impact on interior design gradates.