Evaluating Enterprise Resource Planning Analysis Patterns using Normalized Systems Theory

Ornchanok Chongsombut and Jan Verelst
Department of Management Information Systems
University of Antwerp
Antwerp, Belgium
e-mail: ornchanok.chongsombut, jan.verelst@uantwerp.be

***Abstract*— A dramatically increasing competition in business environment has brought a new characteristic of enterprise information systems called "evolvability". Its on-going changes significantly impact the way information systems are being analysed and designed in practice. Based on Normalized Systems theory, information systems should be designed in 1-1 modular structure to be free from so-called combinatorial effects. Combinatorial effects are one of the biggest obstacles to implementing evolvability of information systems. Combinatorial effects actually occur when a change has a ripple effect on the information system size. Hence, information systems should be designed to minimize combinatorial effects in order to enhance the evolvability of software. Moreover, Normalized Systems theory provides an important practical way of developing evolvable information systems, even huge application systems for organizations. The purpose of the paper is to present an analysis of the analysis patterns of the well-known Microsoft Dynamics CRM 2016 adhering to the design patterns of Normalized Systems theory. Additionally, the paper shows the evaluation of the Enterprise Resource Planning (ERP) analysis patterns from an evolvability point of view and demonstrate both conformance with Normalized Systems theory and violations against it.**

Keywords- Normalized Systems; Evolvability; ERP; Analysis Patterns; Microsoft Dynamics CRM

I. INTRODUCTION

At present, one of the most challenging aspects of designing enterprise information systems is evolvability. Currently, organizations are faced with rapid changes in business environments such as markets, stakeholders, technologies, and so on. Consequently, a high level of evolvability is becoming a highly important issue for software engineering [1][2].

In order to sustain its growth, an organization must deal effectively with all stake-holders. Moreover, the organization should have an information system that collects as much data as possible related to the organization and provides accurate and valuable information about these stakeholders. The information system should be designed to retrieve information in a timely manner for effective decision making and to enhance the overall performance of business operations. Accordingly, ERP systems have become the most important part of enterprises' information systems. Thus, the ERP is generally considered as an integrated business process package [3][4]. However, ERP systems are costly and time-consuming to develop. Moreover, ERP systems have extremely complicated structures, and therefore, they are not easy to implement. Due to both the complexity of ERP systems and organizations' requirement for customized solutions to serve their business objectives, ERP systems should be evolvable. Furthermore, organizations need to be able to respond effectively to their business environment changes in order to maintain a competitive advantage [2][5].

There are a number of products in the ERP market available as both open source and commercial packages. In fact, businesses usually choose standard ERP solutions such as Microsoft Dynamics, SAP, Oracle, Siebel, and PeopleSoft [3]. However, the functionalities of all ERP packages can be changed to meet changing business processes. Therefore, the evolvability of ERP customized solutions has been becoming important in developing ERP systems in order to reduce the cost of maintenance. Here, evolvability means software should be easy to change over time [2][5]-[9].

Based on the stability concept from system theory, a bounded input function should result in bounded output functions, even as $T \rightarrow \infty$. Stability has been applied to Normalized Systems theory, which clarifies how to develop information systems to maximize evolvability [2][5]-[7][9]-[11]. Certainly, information systems should be designed to be able to cope with a set of anticipated changes to increase the level of evolvability. According to the Law of Increasing proposed by Lehman [12], information systems change as time goes by and their structure becomes increasingly complex. It implies that information systems are also faced with the ever increasing size and complexity of their structure and functionality [2][7]. To approach these issues, modularity has been suggested dividing a complicated system into subsystems and coping with the evolvability requirement by allowing the modules to change independently [2][7][13]. However, coupling between modules is the biggest obstacle to evolvable information systems. Coupling relates to the possibility of a change in one module affecting another module. Regarding Normalized Systems theory, a practical guideline for devising evolvable modularity has been provided [2][7][11][13]. Indeed, Normalized Systems theory aims to facilitate the development of highly evolvable information systems [2][5][6][9]-[11][13][14]. To ensure the evolvability of information systems that adhere to Normalized Systems theory, it has been argued that information systems should be developed without combinatorial effects. Combinatorial

 ISBN: 978-1-61208-590-6

effects occur when the impact of a change depends on the size of the information system; in other words, they have a ripple effect on the entire information systems. To increase the evolvability of information systems, these combinatorial effects should be minimized. To simplify the way to eliminate combinatorial effects, four theorems and five elements have been established in Normalized Systems theory (this will be discussed fully in Section 2). To date, a number of studies have already been done on Normalized Systems theory and implemented in several software projects [2][5][6][8][9][11][13]-[17]. Nevertheless, the analysis pattern of ERP packages applying Normalized Systems theory has a number of limitations applying Normalized Systems theory.

In the paper, we analysed the partial analysis patterns of the well-known Microsoft Dynamics CRM 2016 adhering to the design patterns of Normalized Systems theory. Additionally, the paper evaluates the ERP analysis patterns from an evolvability point of view and demonstrates conformance with Normalized Systems theory and violations against it.

The remainder of the paper is structured as follows. We start in Section 2 with some works related to our study. In Section 3, the Normalized Systems theory is discussed, emphasizing the design patterns of Normalized Systems theory. In Section 4, the partial analysis patterns of ERP package are analysed, by focusing on conformance and possible violations with respect to Normalized Systems theory. Finally, we provide the final conclusions, limitations and suggestions for future research.

II. RELATED WORK

In this section, some related works on creating evolvable IT artefacts based on Normalized Systems theory and the evaluation of ERPs' reference model will be discussed briefly.

A. Creating Evolvable IT Artefacts Adhering to Normalized Systems Theory

Normalized Systems theory has recently proposed a framework for developing evolvable modularity [18]. To create the evolvability of information systems, they should not only support current requirements, but also future requirements [9]. Normalized Systems theory suggested that evolvable information systems should be free from combinatorial effects [2][11]. Oorts et al., showed how the Normalized Systems theory could be applied to develop evolvable software and presented the practical advantages of Normalized Systems theory using a case study [16][19]. Additionally, Op't Land et al., conducted the research to evaluate the possibilities of developing information systems based on Normalized Systems theory. This consequence was consistent with previous findings [2][16][19][20]. They argued that the total development and maintenance time were significantly reduced from other application developments by using NS expander [2][16][19][20].

The conformance and violations to Normalized Systems principles of IT artefacts such as source code, business processes workflow and so on have been investigated [18]. Similarly, Vanhoof et al., analysed GAAP Reporting in Accounting area to list both conformance with Normalized Systems theory and violations against its principle [8]. Furthermore, Normalized Systems theory has suggested that its theorems and elements lead to high evolvability of information systems [2][5][6][8][9][14][16][18]-[20].

B. The Evaluation of ERP Packages

The well-known SAP Reference Model was analysed adhering to Normalized Systems theory principles. Some indications were found that seem to reflect Normalized Systems theorems. Moreover, there were some processes of the SAP Reference Model seem to be unrelated to the Normalized Systems theory principles [6]. Mendling et al., stated that there are some error probabilities in enterprise models [21][23]. Additionally, they are usually concealed. Therefore, they evaluated the SAP Reference Model using a verification tool based on Petri net to explore the errors in SAP [21]. The 600 processes of SAP Reference Model were analysed. Consequently, several errors in SAP Reference Model were found [21][23].

III. NORMALIZED SYSTEMS THEORY

Normalized Systems theory provides a practical way of developing evolvable information systems through the so-called pattern expansion of software elements [19].

A. Normalized Systems Theorems

To guarantee high evolvability of information systems, Normalized Systems theory proposes four theorems [10]. Furthermore, how the four Normalized Systems theorems are manifested in a practical way is shown in Table I.

TABLE I. NORMALIZED SYSTEMS FOUR THEOREMS IN PRACTICE [18]

Normalized Systems Theorems	The practical way of developing information systems
Separation of Concerns	• Multi-tier architectures separating presentation logic, application or business logic, database logic, etcetera
Data Version Transparency	• Polymorphism in object-orientation • Wrapper functions
Action Version Transparency	• XML-based technology (e.g., for web services) • Information hiding in object-orientation
Separation of States	• Asynchronous communication systems • Stateful workflow systems

B. Normalized Systems Elements

Five expandable elements were proposed to ensure the evolvability of Normalized Systems applications. The internal structure of these five elements is described by Normalized Systems design patterns such as data elements, action elements, work-flow elements, connector elements, trigger elements [6][19][16].

 ISBN: 978-1-61208-590-6

IV. ANALYSING THE PARTIAL ANALYSIS PATTERNS OF THE ERP PACKAGE

In this section, we examine the Microsoft Dynamics CRM 2016 from an evolvability point of view and demonstrate both conformance with Normalized Systems theory and violations against it.

A. *Indications towards of conformance with Normalized Systems principles*

Here, our aim is to examine conformance between the model of Microsoft Dynamics CRM and Normalized Systems principles.

Based on the Normalized Systems theorems, most of the model of Microsoft Dynamics CRM seem to be related to the Normalized Systems principles. Firstly, the Microsoft Dynamics CRM architecture has a Multitier architecture. Moreover, the Microsoft Dynamics CRM implements cross-cutting concerns, for example, Reporting (Dashboards, Charts, Excel and SRS), Security model that focuses on access rights to the entities in the system [6]-[8]. For first and second points straightforwardly follow from the Separation of Concerns theorem.

According to the Data version transparency theorem, data entities can be modified (insert, delete, update) without affecting the calling actions [9]. In Microsoft Dynamics CRM, the information hiding has been applied to develop the software. Properties cannot be directly accessed, but can be read or written by using provided method. Additionally, Microsoft Dynamics CRM has been implemented using XML based technology that leads to conformance with the Data Version Transparency theorem.

Following the Action Version Transparency theorem, this theorem implies an action can be modified without affecting the calling actions. First, Microsoft Dynamics CRM is usually implemented through wrapper functions through the use of polymorphism in C#.NET or VB.NET. Second, the Microsoft Dynamics CRM implements cross-cutting concerns as explained above. Therefore, the developing of Microsoft Dynamics CRM relates the Action version transparency theorem.

The Microsoft Dynamics CRM relies on asynchronous service to improve overall system performance and scalability [10]. Combinatorial effects can be avoided through asynchronous processing.

B. *Indications towards violation of Normalized Systems principles*

When analysing the analysis patterns of Microsoft Dynamics CRM, some indications towards violation of the Normalized Systems principles might be noticed. Microsoft Dynamics CRM addresses challenge of customer management, therefore, this module was analysed in point of evolvability. The entities are used to model and manage business data in this module. In programing, an entity is represented by a class, such as the Account class generated from the Account entity.

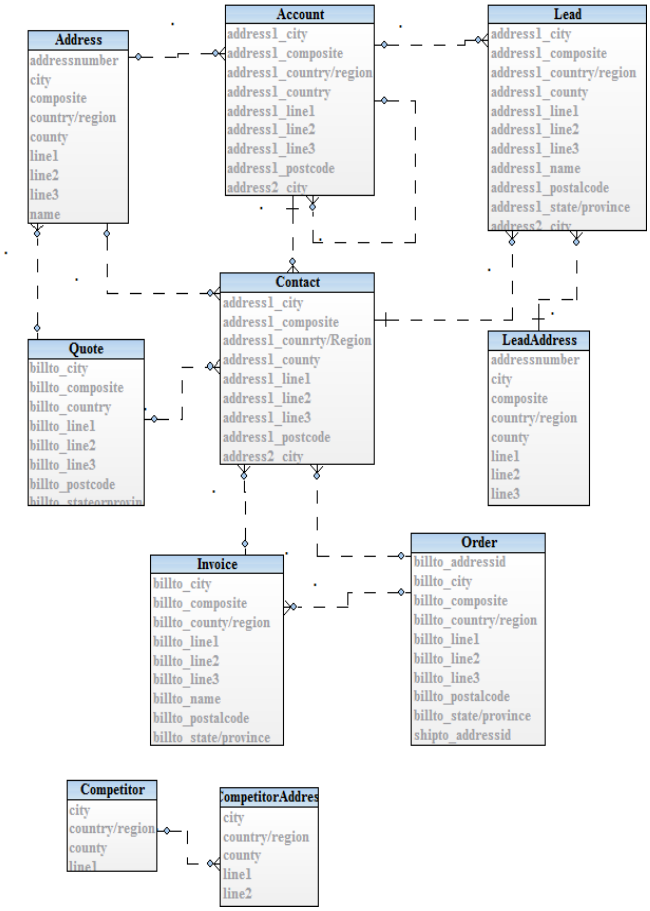

Figure 1. The partial ER diagram of Microsoft Dynamics CRM

Fig. 1 illustrates an ER diagram consisting of ten entities. We have noticed the attribute duplication of address details in many Classes such as Account, Contact, Address, Lead, LeadAddress, Quote, Invoice, and Order. Attribute duplication seems contradictory to Normalized Systems theorem, Separation of Concern. According to the assumption of unlimited systems evolution, software can be changed over time. Therefore, the eventual impact might become related to the overall system size and lead to a combinatorial effect.

V. CONCLUSION

In this paper, we analyse the analysis patterns of ERP package, Microsoft Dynamics CRM, to explore conformance with Normalized Systems theory and violations against it. While the interpretation of the analysis patterns of ERP package shows some conformance towards Normalized Systems theory, it also presents some analysis patterns towards violations of Normalized Systems principles. The finding is found the developing well-known commercial ERP package seem to relate Normalized Systems theory both four theorems and five elements [2][18]. On the other hand, a few points seem to contradict Normalized Systems

 ISBN: 978-1-61208-590-6

principles. Similarly, the finding of Mendling et al., there are some error probabilities in enterprise models. Moreover, they are usually concealed [21]-[23]. This paper makes first contribution towards presenting the possibility of ERP evaluation adhering to Normalized Systems theory in the context of evolvability. Second, the paper contributes to the ERP development applying Normalized Systems theory to achieve the evolvability.

The limitations of our study need to be acknowledged. First, we only analysed partial analysis patterns of one ERP package. We could not perform reverse-engineering and explore more source code of commercial ERP software packages to look at combinatorial effects. As part of future research, we will redesign and rebuild the existing data model of existing ERP software packages based on NS theory. In practice, we will rebuild existing ERP packages using the Normalized Systems expander to obtain high evolvability [2][16][19].

REFERENCES

[1] S. Kelly and C. Holland, "The ERP Systems Development Approach to Achieving an Adaptive Enterprise: The Impact of Enterprise Process Model-ling Tools," in Systems engineering for business process change: new directions. Springer London, pp. 241-252, 2002.

[2] H. Mannaert, J. Verelst, and K. Ven, "Towards evolvable software architectures based on systems theoretic stability," Software: Practice and Experience, vol 42, no. 1, pp. 89-116, 2012.

[3] K. Ganesh et al., "Enterprise Resource Planning: Fundamentals of Design and Implementation," Springer, 2014.

[4] E. Shehab, M. Thomassin, and M. Badawy, "Towards a Cost Modelling Framework for Outsourcing ERP Systems," in Improving Complex Systems Today: Proceedings of the 18th ISPE International Conference on Concur-rent Engineering, D.D. Frey, S. Fukuda, and G. Rock, Editors, Springer London: London, pp. 401-408, 2011.

[5] P. Huysmans and J. Verelst, "Towards an Engineering-Based Research Approach for Enterprise Architecture: Lessons Learned from Normalized Systems theory," in Advanced Information Systems Engineering Workshops: CAiSE 2013 International Workshops, Valencia, Spain, June 17-21, 2013. Proceedings, X. Franch and P. Soffer, Editors, Springer Berlin Heidel-berg: Berlin, Heidelberg, pp. 58-72, 2013.

[6] P. De Bruyn et al., "Towards Applying Normalized Systems theory Implications to Enterprise Process Reference Models," in Advances in Enterprise Engineering VI: Second Enterprise Engineering Working Conference, EEWC 2012, Delft, The Netherlands, May 7-8, 2012. Proceedings, A. Albani, D. Aveiro, and J. Barjis, Editors, Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 31-45, 2012.

[7] P. Huysmans et al., "Positioning the Normalized Systems theory in a Design Theory Framework," in Business Modeling and Software Design: Second International Symposium, BMSD 2012, Geneva, Switzerland, July 4-6, 2012, Revised Selected Papers, B. Shishkov, Editor, Springer Berlin Heidel-berg: Berlin, Heidelberg, pp. 43-63, 2013.

[8] E. Vanhoof et al., "Building an Evolvable Prototype for a Multiple GAAP Accounting Information System," in Advances in Enterprise Engineering X: 6th Enterprise Engineering Working Conference, EEWC 2016, Funchal, Madeira Island, Portugal, May 30-June 3 2016, Proceedings, D. Aveiro, R. Pergl, and D. Gouveia, Editors, Springer International Publishing: Cham, pp. 71-85, 2016.

[9] J. Verelst et al., "Identifying Combinatorial Effects in Requirements Engineering," in Advances in Enterprise Engineering VII: Third Enterprise Engineering Working Conference, EEWC 2013, Luxembourg, May 13-14, 2013. Proceedings, H.A. Proper, D. Aveiro, and K. Gaaloul, Editors, Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 88-102, 2013.

[10] P. De Bruyn, "Towards Designing Enterprises for Evolvability Based on Fundamental Engineering Concepts," in On the Move to Meaningful Internet Systems: OTM 2011 Workshops: Confederated International Workshops and Posters: EI2N+NSF ICE, ICSP+INBAST, ISDE, ORM, OTMA, SWWS+MONET+SeDeS, and VADER 2011, Hersonissos, Crete, Greece, October 17-21, 2011. Proceedings, R. Meersman, T. Dillon, and P. Herrero, Ed-itors, Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 11-20, 2011.

[11] H. Mannaert, J. Verelst, and K. Ven, "The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability," Science of Computer Programming, vol. 76, no. 12, pp. 1210-1222, 2011.

[12] MM. Lehman, "Laws of software evolution revisited," in European Workshop on Software Process Technology, Springer, 1996.

[13] D. Van Nuffel, "Towards designing modular and evolvable business processes," Universiteit Antwerpen, 2011.

[14] P. Huysmans et al., "Aligning the Normalized Systems theory Constructs of Enterprise Ontology and Normalized Systems," in Advances in Enterprise Engineering IV: 6th International Workshop, CIAO! 2010, held at DESRIST 2010, St. Gallen, Switzerland, June 4-5, 2010. Proceedings, A. Albani and J.L.G. Dietz, Editors, Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 1-15, 2010.

[15] M.R. Krouwel and M. Op't Land, "Combining DEMO and Normalized Systems for developing agile enterprise information systems," in Enterprise Engineering Working Conference, Springer, 2011.

[16] G. Oorts et al., "Easily evolving software using normalized system theory-a case study," Proceedings of ICSEA, pp. 322-327, 2014.

[17] K. Ven et al., "Experiences with the automatic discovery of violations to the normalized systems design theorems," International Journal on Advances in Software, vol. 4, no 1 & 2, 2011, 2011.

[18] P. De Bruyn, D. Geert, and H. Mannaert, "Aligning the Normalized Systems Theorems with Existing Heuristic Software Engineering Knowledge," The Seventh International Conference on Software Engineering Advances, pp. 84-89, 2012.

[19] G. Oorts et al., "Building evolvable software using Normalized Systems theory: A case study. in System Sciences (HICSS)," 2014 47th Hawaii International Conference, IEEE, 2014.

[20] Op't Land et al., "Exploring normalized systems potential for dutch mod's agility," in Working Conference on Practice-Driven Research on Enterprise Transformation, Springer, 2011.

[21] J Mendling et al., "Faulty EPCs in the SAP reference model," in International Conference on Business Process Management, Springer, 2006.

[22] J Mendling et al., "Errors in the SAP reference model," BPTrends, vol. 4, no. 6, pp. 1-5, 2006.

[23] J Mendling et al., "Detection and prediction of errors in EPCs of the SAP reference model," Data & Knowledge Engineering, vol. 64, no. 1, pp. 312-329, 2008.

 ISBN: 978-1-61208-590-6