International Journal of Electronics Communication and Computer Engineering
Volume 6, Issue 3, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Dynamic Structure for Adaptive Software System

Maryam J. Abdullah

***Abstract* – Software system is built to service users in its environment domain. It must have the ability to change its services in the case of user requests or environment changes after the deploy time. When design phase of the development life cycle is built, the dynamic software architecture must be designed in the way to give the system an ability to change its services at run time. This paper introduces an Independent Architecture Framework (IAF) for dynamic software system. The proposed IAF is designed in way to represent each functions independently. Then these function are connected to build the complete software system services. Therefore, the new IAF enable the software system to easily add, update, upgrade and swap its services at run time.**

***Keywords* – Architecture, Dynamic Software System, Fix Software System, Dynamic Structure, Function Process, APOC Architecture Framework.**

I. INTRODUCTION

Software system product executes the behaviors to serve the users in their domain. A software system cannot change in its behaviors is the static software system, otherwise, it is named a dynamic software system. Dynamic software system has the ability to change its execution behaviors during the run time without need to shout down [1].

Software system development life cycle consists the main phases of: requirement, analysis, design, implementation, test and deployment [2]. In the requirement phase, the software system services are determined. The analyzer can break down services into a set of functions using requirements gathered in requirement phase and can determine functions with high possibility of change during executing time of software system product [3], [4]. Structure of a software system is an output of the design phase and an input to of the implementation phase [5], [6], [7]. Structure of a software system is used in the implementation phase to program the software system [5, 8, 9]. Software system structure breaks down the system into many components [10], [7]. Each component holds the functions. The Action- Priority- Observer -Component architecture framework (APOC) shows the system as set of components connected to execute software system behaviors [11], [12]. Each component has many parameters with value pass to next and get from previous component. These parameters processed by a function that was hold by the system components [11], [12].

In this work, IAF is proposed to build adaptive software system. Each service of the software system is executed by process line. This process line consists of a set of functions. IAF has the ability to accept and control the function process changes at the run time, and modify the process line. In the IAF, the dynamic software system must define each function independently and hold it by a manager. The storage of functions is generated from implementation phase and continue increasing in the number of functions after software product deployed in its environment. Storage can be modified at run time. In the IAF, the function changes can be completed without any effect on the current software system service execution.

The structure of paper is as follows: Section 2 software system is overviewed. In section 3 APOC architecture framework is presented. The illustrations of dynamic structure in the proposed architecture (IAF) are presented in section 4. In section 5 simulation results of the proposed architecture and a comparison with a common architecture are made. Section 6 provides conclusions and summery of the work.

II. SOFTWARE SYSTEM

Software system is programmed to be a set of packages. Package is consist of a set of Classes. Each class hold functions to process data [9]. Software system services represented by the process line that consists of a set of functions. Data result from the end function in process line is the result that needed by end user. A software system with the ability of modifying its process line during run time is the dynamic software system otherwise it is the fix or static software system.

A. Static software system

Static software system product is system with static structure [5]. There is no ability or interface to its process line to be modified at run time. Therefore static software system services cannot be changed. Static software system has a storage of functions and its services construct by connect between these functions. Software system behaviors are limited by what lines of process that are built at implementation phase.

B. Dynamic software system

Dynamic software system is a system can generate new behaviors or modify its behaviors already exist to adapt environment changes or user requests at the run time [5], [13], [14], [7]. Observer is used to sense any environment changes or user requests, then asks software system to adapt changes by modifying its current run services process. Dynamic behavior for software system gives the software product the ability to be modifying its functions process during run time without need to shout down. Execution software system functions process do not effected by adaptation processes. Adaptation processes result are deployed into software system functions at run time.

III. THE ACTION- PRIORITY- OBSERVER - COMPONENT ARCHITECTURE FRAMEWORK (APOC)

APOC software system architecture divided system into components [11], [12]. Each component has an action link,

International Journal of Electronics Communication and Computer Engineering
Volume 6, Issue 3, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

priority link, observer link and component link. Each link with parameter value is processed by an updated function. Component uses its updated function to process each received input link value. For sequential architecture, software system is viewed as one APOC component which holds the entire software processes. While software system that consists of more than one process running parallelly is divided to many APOC components. Adaptation is done by changing parameter values held by APOC component to determine the status of component in the behavior strategy for the software system [11], [12].

IV. Independent Architecture Framework (IAF)

In this work, the IAF proposes a dynamic structure to design a software system able to change execution services at the run time. IAF arranges software system into two groups of components; the first group of components responds to achieve adaptation process and the second group responds to achieve software system services.

Software system process is either one process line or more than one process line running parallelly. Figure 1 shows how the IAF is dividing process line into a set of functions. The two subsections below illustrate adaptation behavior for the software system built by IAF and adaptation components for IAF, respectively.

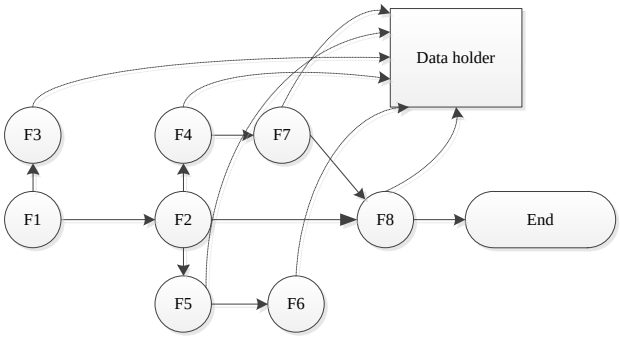

Fig.1. A line of process in dynamic structure.

A. Behavior Architecture

In this work, the IAF gives the software system ability to develop its process line at the run time. This ability is given to software system from the first time in the design phase of software system development life cycle. Functions that consist the process line can be updated, upgraded or added new instructions. In the adaptation status, software system must keep providing services to the end user. Services provide in the adaptation status must be acceptable until providing the new process line fit with new status. Generating the new process line is started by setting a notification from observer to adapter. This notification holds variables explain new status. Depending on these variables, process line is built. Figure 2 illustrates steps of generation the new process line by getting list of references to function exist in the software system storage. Merging functions and comparing results to new status variables is continuing until achieving the new fit process line.

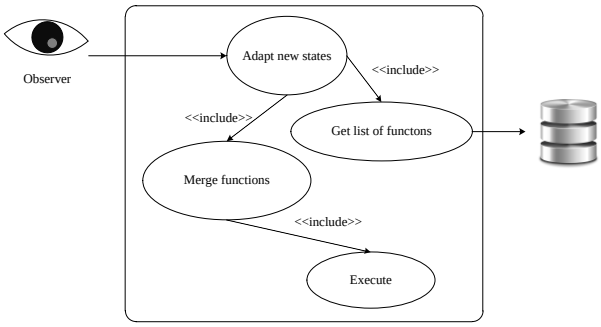

Fig.2. Adaption of a new environment status.

1) Add, update and upgrade functions.

Figure 3 illustrates how to add the new function to the software system storage. First the storage manager checks for the function existence, if not exist, the function is added. Updated or upgraded is made when a programmer have to modify over a function already exists in the software system storage. Update a function is made by applying sub changes within its instruction. Upgrade is made by replacing a function already exists in the software system with the new version. In the booth cases, update and upgrade are modifying over new function copy. Functions before modification are saved to be gotten back by the software system when it needs for them.

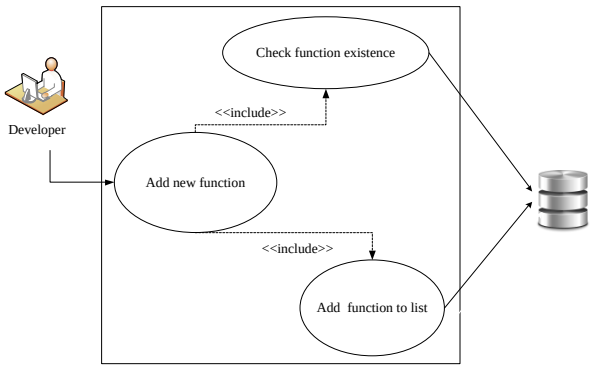

Fig.3. Addition of a new function to software system storage.

2) Adapt new statues.

In case of an environment change or a user request, the observer asks the software system to adapt new status and uses its function storage to generate new process line. The generation of new process line is illustrated in Figure 4. It has the following sequence:

1) The observer sends a massage to notify the adapter for new status.
2) The adapter asks the function manager to provide a list of functions exist in the software system storage.
3) Merge between functions to get new process line.
4) Test the new process line by the adaptor.
5) If the process line result is fitting with new status, the process line will be deployed and the result will show to the end user.
6) Otherwise, remerge functions to get new process line.

International Journal of Electronics Communication and Computer Engineering
Volume 6, Issue 3, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

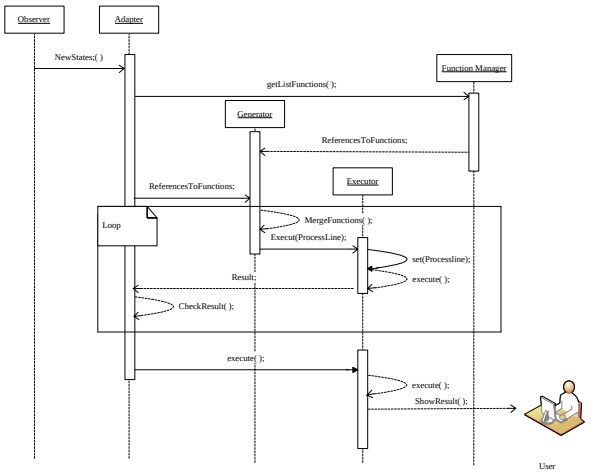

Fig.4. Generation of a new process line.

B. Adaptation process

Adaptation components for IAF are the observer, adapter, generator and function manager. The adaptation process for a dynamic software system is divided into subtasks. Each component of the IAF is responsible for a subtask. Figure 4 shows how the IAF components interact in case of new environment status and software system adaptation are needed. The observer component is sensing any changes on environment variables. When the observer gets a change over variables will send message to the adaptor. The adaptation massage holds variables value of new environment status used by the adaptor component. The adapter component holds current environment variables and checks if current process line can fit with the new environment status. If no, a new process line is generated by the generator component and is tested by the adaptor component. The generator component uses the function storage to generate a new process line. The function manager component holds references of all functions exist in the function storage. Any change over the function storage is updated, upgraded, and added new function is controlled by the function that manager component.

The data holder component holds global access data by software system functions. In execution of the process line, each function is using the data holder component to get current data value, execute process and set result by the data holder, as show in Figure 5.

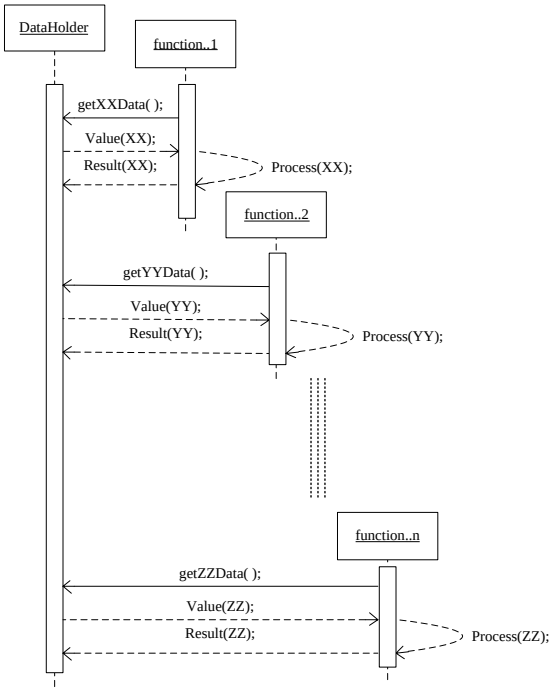

Fig.5. Execution of a function.

V. RESULTS AND DISCUSSIONS

IAF proposal gives software system the ability to change its process line to adapt environment change or user request. IAF separates adaptation components from functional behavior components. Adaptation components are fixed for any dynamic software system in any domain. Developer reuses adaptation components at any time they use to build new dynamic software system and focuses on functional behavior components of software system. In Spite Of the system has one or parallel process lines the IAF is break the dynamic software into many components to have an efficient adaptation process.

VI. CONCLUSION

In this paper, an IAF is developer for a dynamic software system, to have modifications at run time without need to stop its process execution. Dynamic software system uses IAF to break its process line into a set of small functions which can be added, updated and upgraded on already function storage. Process line can be modified or generated depending on adaptation status requested. The simulation results approved effectiveness of developed IAF dynamic software architecture to adapt environment status changes and accept changes at run time.

REFERENCES

[1] Nick Rozanski, and Eon Woods. Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives. 1nd ed. Addison-Wesley Professional (April 30, 2005).

[2] Robert J. Allen, Remi Douence, David Garlan. Specifying and Analyzing Dynamic Software Architectures. Proceedings of the 1998 Conference on Fundamental Approaches to Software Engineering (FASE '98), March 1998.

[3] Ms. Shikha maheshwari, Prof.Dinesh Ch. Jain. A Comparative Analysis of Different types of Models in Software Development Life Cycle. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 5,May 2012.

[4] Matthias Scheutz. APOC- An Architecture Framework for Complex Agents. 2005 by Idea Group.

[5] Peyman Oreizy, Michael M. Gorlick, Richard N. Taylor, Dennis Heimbigner, Gregory Johnson, Nenad Medvidovic, Alex Quilici, David S. Rosenblum, Alexander L. Wolf. An Architecture-Base Approach to Self-Adaptive Software. Intelligent Systems and their Applications, IEEE (Volume:14 , Issue: 3),1999.

[6] David Garalan, Dewayne Perry. Introduction to special issue on software architecture. IEEE Transactions on Software Eng., April 1995.

[7] Yuriy Brun1, Giovanna Di Marzo Serugendo, Cristina Gacek, Holger Giese,Holger Kienle, Marin Litoiu, Hausi M¨uller, Mauro Pezz`e, Mary Shaw. Engineering Self-Adaptive Systems through Feedback Loops. Springer-Verlag Berlin Heidelberg 2009.

[8] Felix Bachmann, Len Bass, Paul Clements, David Garlan, James Ivers, Reed Little, Robert Nord, Judith Stafford. Documenting Software Architecture: Documenting Behavior.2002 by Carnegie Mellon University.

[9] Chrysanthos Dellarocas, Mark Klein. An Architecture for Constructing Self-Evolving Software Systems. Howard Shrobe ,ISAW '98 Proceedings of the third international workshop on Software architecture,1998.

[10] Jeff Magee, Jeff Kramer. Dynamic Structure in Software Architectures.SIGSOFT'96 CA, USA, 1996 ACM 0-89791-797-9/96/0010.

[11] Monica Farah-Stapletona, Mikhail Augustonb .Behavioral Modeling of Software Intensive System Architectures. Procedia Computer Science 20 (2013) 270 – 276.

[12] Matthias Scheutz, Virgil Andronache. Architectural Mechanisms for Dynamic Changes of Behavior Selection Strategies in Behavior-Based Systems. Part B: Cybernetics, IEEE Transactions on (Volume:34, Issue: 6), 2377 – 2395, Dec. 2004.

[13] Robert G. Crispen , Lynn D. Stuckey. Structural model: Architecture for Software Designers. Jr. TRI-Ada '94 Proceedings of the conference on TRI-Ada '94,Pages 272-281,1994.

[14] xinjun mao, menggao dong, lu liu, huaiming wang. An Integrated Approach to Developing Self-Adaptive Software in Open Environments.journal of information science and engineering 30, 1071-1085 (2014).

AUTHOR'S PROFILE

Maryam J. Abdullah
has a master degree in software engineering from computer science, DePaul University, USA, 2009. She is currently a teacher assistant and joined the Electronic Web Site at University of Baghdad. She has a good experience in software language programming. Her areas of research include artificial inelegant, project management and software programming.