International Academic Institute for Science and Technology

International Academic Journal of Innovative Research
Vol. 2, No. 1, 2015, pp. 10-19.

ISSN 2454-390X

International Academic Journal of Innovative Research

www.iaiest.com

Reviewing Self-Adaptation Frameworks For The Implementation Of Enterprise Resource Planning Systems

Fatemeh Alviri[a], Shahedeh Farahmand Habibi[a]

[a]*Master Student of IT Engineering, Islamic Azad University, Qazvin, Iran.*

Abstract

Increase of workload and management of complexities in an efficient way and maintaining that in different situations, lead attentions to adaptable and self-adaptive systems. The goal of this research is to choose the best self-adaptive framework present in this research to apply for enterprise resource planning (ERP) systems. Reviewing some self-adaptive frameworks and their outlines is the way of doing the research. Based on this research Rainbow framework is selected to be applied on the ERP system as it's provided a context for developing other frameworks because of containing most major parts to develop a self-adaptive system. There are few and diffused studies on ERP adaptability so that this research tries to choose the best adaptable framework for ERP systems by gathering these issues.

Keywords: ERP systems, Adaptation, Self-adaptation

Introduction:

In recent years size and complexity of software systems and applications are growing continuously as they have to response to complicated requirements. Increase of workloads and complexity management in an efficient way in different situations lead attentions to adaptable and self-adaptive systems. Actually numerous and changing requirements of users in complex systems is the main goal of designing and developing these systems. Since some of applications are running constantly, these systems have to maintain status and stable conditions of the system without humans' interference against events and attacks which cause failure. Enterprise resource planning (ERP) systems are those kinds of systems which can be investigated from different points of view like adaptation and available methods such as self-adaptation. We should pay attention that most ERP providers have become a middle market for their

growth these days. For ERP providers, adaptive strategy and their business solutions are important in order to develop their position in market. To gain this goal, ERP providers require adaptation of their sale cycle and acquiring exact knowledge of specific industry needs. ERP systems are adjustable complex technologies, which need important investments for adapting with a specific company (Hwang, 2015). In this research reviewing self-adaptive frameworks and choosing the best framework for ERP systems will be discussed.

Theoretical basis of research:

ERP Systems

ERP is a tool for standardization and integration of business process to accelerate access of common resources in the whole organization so that ERP systems help organizations the ease of sharing information and improving operational efficiency. Successful deployment and use of ERP systems is necessary for organizational performance and survival (Shena, 2015).

Through these years, domain of ERP systems has extended. One day they were just an operational tool, but nowadays they evolved as a comprehensive approach for development of strategic infrastructures (for example adaptation, reconfiguration and integration of information flows and business process). ERP system implementation is both technical (like need of a change in technological systems) and behavioral (like need of a change in work concept, habits and routines). ERP implementation strategy from dynamic capability point of view is defined as a long-term widespread initiative in organization level to increase information flow and business process in terms of adaptation, configuration, integration and user training. Integration aspects of ERP systems can be useful in achieving business goals in all strategic, tactical and functional levels. ERP systems integrate necessary business process while increasing functional performance in the fields of human resource, marketing, accounting and operations simultaneously. Proper user training in efficient use of ERP systems has positive effect on sale and customer service. ERP implementation strategies increase process improvements. ERP systems' adaptation, promotes sharing information with customers, providers and other business partners. Organizations with higher levels of integration, probably reach to higher operational results. Challenges in ERP contains (a) combining different ERP operational modules integrated, (b) connecting with other applications of the business software and (c) upgrading old systems with other providers' IT systems (Hwang, 2015).

Mr. Umble and his colleagues and Mr. Al-Mashari and his colleagues in 2003 through an article divided ERP advantages in two categories:

Tangible benefits; which contain inventory reduction, personnel reduction, productivity increase, improvement in order management, faster close of financial cycles, increase of on-time deliveries, improvement of customer services, reduction of IT costs and supplies, improve of cash flow management, increase of income and profit, transportation cost reduction, and reduce of need to repair and maintenance system.

Intangible benefits; which contains shared data exposure, increase of new or improved business process, improving customer response, communication improvement, reduce of unexpected costs, better integration between systems, standardization of computational platforms, increase of flexibility, global information sharing, improving business operation and exposing supply chain management (SCM) process (Fan, 2008), (Al-Mashari, 2003).

ERP Adaptation Benefits

Considering David Hwang et al. article in 2015, ERP adaptation benefits can be expressed as bellow:

- ERP system adaptation, enables the company to manage stoke and administrative costs, and cause customer response improvement.
- ERP system adaptation develops information sharing with customers, suppliers, providers and other business partners.
- Adaptive ERP process, along learning proper users, let companies to recognize environmental changes.
- ERP adaptation let companies to gather necessary information (Hwang, 2015).

Adaptation

In recent years adaptation concept and related concepts emerged with the goal to reduce or remove software systems need from human interfere. Adaptation or maladjustment is a concept that describes relationship between systems and environment. On the other side, adaptation means that products, energy and information between system and environment are changing in a regular and stable manner (Umble, 2003).
According to Hwang et al. (2015) adaptation means "the degree in which a company accepts and adjusts new technologies and systems in response to changing environmental requirements" (Hwang, 2015).

Self-adaptive System

A self-adaptive system is defined like this; "a self-adaptive system evaluates its behavior and when its behavior is in contrast with software's defined goals, or higher performance is possible for the system, it will change its behavior". In a wider definition a self-adaptive system is a system which changes its behavior in response to data changes in the operational environment (Omicini, 2003).

Self-adaptive Cycle

Something that differentiates self-adaptive systems from previous systems is a feedback loop which results from interaction of a self-adaptive component and a managed resource. Proposed cycle in here is completely like the one IBM proposed which is known as MAPE-K cycle. This cycle can be seen in figure 1 (Dubios, 2010).

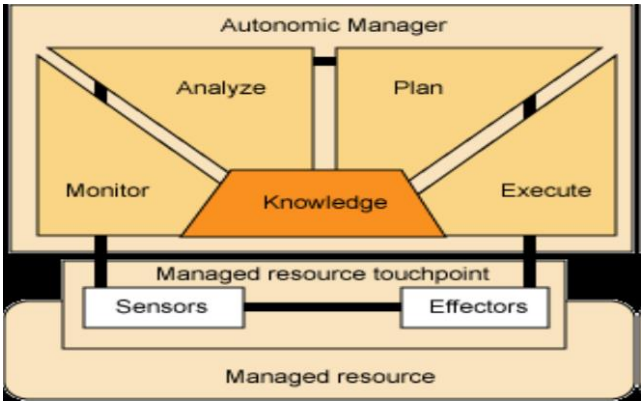


Figure 1 self-adaptive cycle

Managed Resource

Managed resource is a component under system's control. This resource can be individual or compound. A managed resource is controlled by its sensor and effector. The sensor and effector altogether create the resource touch point.

As can be seen in figure 1 there is a connection between the sensor and effector. This means that a strong compatibility exists between these two and each change in effector's structure should be applied on the sensor and vice versa.

This managed resource is inevitably like those ones which exit in non-self-adaptive usual systems but the difference here is that a resource should be adapted in such a way that a self-adaptive component could control and monitor it. Based on this definition a managed resource can be a hardware resource such as memory, processor and printer or a software resource like an application service or even a business service (Dubios, 2010).

Self-adaptive Component

Self-adaptive components are system's forming units which monitor some resources and supply the results for human users or other components. This component is in relation to one or more managed resources and receives their changes from its sensors and after analyzing changes and programming for proper reaction, affects the environment from its execute tools. As can be seen in figure 1 each self-adaptive component cycle consists of five parts.

Monitor: This part provides some mechanisms for gathering, aggregation, refinement and reporting details received from a resource.

Analyze: this part provides a tool for components so that it could model a complicated situation and recognize a logical connection between components. These mechanisms enable the component to have learning in its information technology environment and could predict future.

Plan: This part provides some mechanisms for structuring the operation which is going to be done toward reaching goals.

Execute: These parts provide mechanisms for executing programed operation. Of course it should be possible to change the operation in execution time.
Knowledge: As can be seen in figure 1, knowledge is set in the middle of the self-adaptive cycle. IBM definition of this knowledge is the data that each one of four autonomous components uses. This data is saved as a shared knowledge. This knowledge consists of log files, performance criterions and policies. The source of this knowledge can be from monitor and execute parts. While analysis and plan parts are mostly related to policies (Dubois, 2010).

Research Background

Self-adaptive Framework Classification And Done Works

Adaptive process is a concept, inspired by human's neural system which its goal is to overcome the complexity of managing heterogeneous wide systems (Khalid, 2009). For this reason human's intervention in managing large systems is reduced by developing the four self-adaptive features.

Yang et al. Framework

This framework has been proposed in 2010. In development of this framework a central fuzzy controller is considered. As it uses heuristic information to create system controlling strategies, it can support inaccurate or vague data. Fuzzy control, acts much like humans' inference, so it is very helpful to be used in self-adaptive systems which work with unreliable vague information (Yang, 2010).

Framework's Outline

This framework has two layers: adaptive logic layer and adaptive system layer. As can be seen in figure 2, these two layers communicate through software bus (Yang, 2010).
Adaptive logic layer: This layer consists of an adaptive engine and necessary logic to change the target software system. Adaptation strategies come true through fuzzy rules and get saved in fuzzy rules' base. Adaptive engine uses Fuzzy inference mechanism for adaptive decision making. There are two kinds of input data in adaptive logic layer: 1- adaptation objectives and 2- machine inputs. Adaptation objectives are objectives which will evolve target software systems. Adaptation objectives are often constant and machine inputs come from sensors used in target software systems. In general, an error between adaptation objectives and machine inputs is a condition which activates adaptation engine.
A defuzzification inference of adaptation engine is the output of adaptive logic layer. Outputs affect the software system through actuators. In summary, adaptive logic layer as a Fuzzy controller is for controlling critical features of software systems in adaptive system later.
Adaptive system layer: This layer provides main functions of software systems. Furthermore, sensors and actuators in the software system are considered for adaptation. Sensors have to provide anything inference engine needs for adaptation about system's status. Actuators execute orders of adaptive logic layer and apply changes to adaptive logic layer. So, software system becomes adaptive.

Software bus: Software bus is a conceptual and logical communication. This bus has provided unified and integrated protocols to create communication between adaptive logic and adaptive system layers. With help of software bus, these two layers can execute in different process and systems separately.

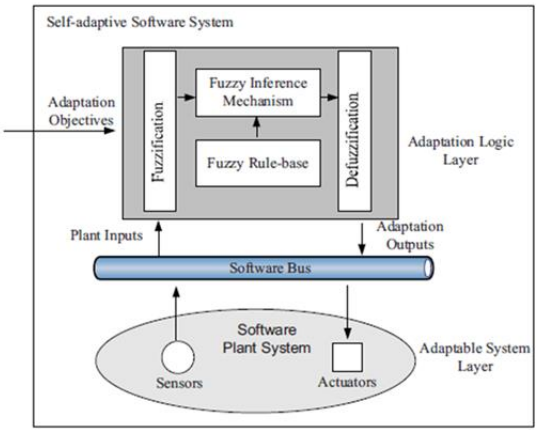


Figure 2 Yang et al. framework outline

One drawback of the system is that adaptation level of the system and amount of actuators capability on applying changes in system is undefined.

Krishnamurthy And Babu's Framework

This framework was proposed in 2010. In this framework a dynamic self-adaptive method is presented in which self-adaptive architecture patterns are combined with adaptive frame work of a self-adaptive system, which makes it possible for the system to change its architecture for adaptation, based on some predefined and fixed architecture patterns. Then these changes are reflected to the running system (Krishnamurthy, 2010).

Framework Outline

This framework is an adaptive framework which shows three major elements that is monitoring, decision making and reconfiguration. This framework works with help of an external control and monitoring mechanism. The system's running properties, update during runtime by feature mapper in architecture description language. These features are examined by defined rules in external observer's mechanism. Whenever a rule is violated, constraint violation happens and the decision making unit starts to reconfigure system's structure. This reconfiguration is done based on architecture patterns. Finally these changes are mapped to the running system. Absence of defining a specific method for gathering system's features is one of this framework's features.

Georgas And Tailor's Framework

This framework is presented in 2004. This architecture describes a knowledge based approach for reasoning on possible adaptive spaces in order to create self-adaptive systems. In this approach, knowledge-based architectural adaptation management (KBAAM), reasoning and decision making about scheduling and nature of specific adaptations is based on knowledge-based adaptive policies. Both policy and knowledge of the related system are modeled, and include an architecture description of a self-adaptive system, explicitly, as first class elements of architecture. After that, these elements are managed dynamically and are being reasoning by an expert system (Georgas, 2004).

Framework Outline

In this framework an architecture-based approach is accepted for the self-adaptive software and for applying methods to systems which are made from independent components and are connected through first class connections. This model of architecture is used as a base for adaptive decomposition to three elements:

- All types of knowledge and information which need adaptation.
- Architecture operations in which through them adaptation approves.
- Adaptation policies which map between two above elements.

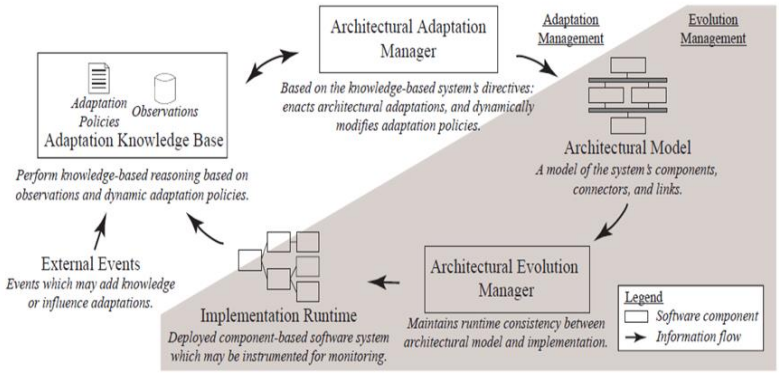


Figure 3 Georgas and Tailors' Framework Outline

A high level review of approach elements can be seen in figure 3. Central abstraction is the architectural model which defines the software system's structure. Architectural evolution manager (AEM) protects cohesion between architecture and implementation.

Events are taken from system at runtime or are created by system monitoring facilities which are gathered in adaptation knowledge base. Also external gathered events are not originating from self-adaptive systems. These events might keep other software systems' information in a case, or create cooperation adaptation policies remotely. Knowledge base's responsibility is to gather, maintain and reasoning observations, which keep self-adaptive system's information in cases. This knowledge base supports reasoning process with applying architectural adaptation policies on the body of existing observations, and defines if compatibility is needed. Responses are notified to the Architectural Adaptation Manager (AAM) based on architectural changes. These changes might be of two kinds: adaptation policy changes

which are used by knowledge base at the moment, or structural changes to the system's architectural model. The cycle is resumed by insurance of AEM for approving architectural changes which are dynamically reflected in structure and operation of a running system (Georgas, 2004).

Rainbow Framework

This framework was provided in 2004 by David Garlan et al. Rainbow framework uses software architecture and reusable infrastructure in order to support self-adaptive software systems. Using external adaptive mechanism let explicit and clear determination of adaptive strategies for worry of multiple systems.

This framework uses models and external mechanism with a closed-loop control method, to reach different goals by use of monitoring and system behavior adaptation at runtime. External control mechanisms, present more effective engineering solutions for adaptation, than internal mechanisms, as they integrate concerns of problem detection and resolution of separate modules that might be analyzed, changed, extended and reused all over different systems.

This framework provides reusable infrastructures with mechanisms to specialize the infrastructure of special system's needs by adapting an architectural-based approach. This special mechanism let developers of self-adaptive capabilities to choose aspects of the system for modeling and observation, starting adaptation conditions and how the system adapts (Garlan, 2004).

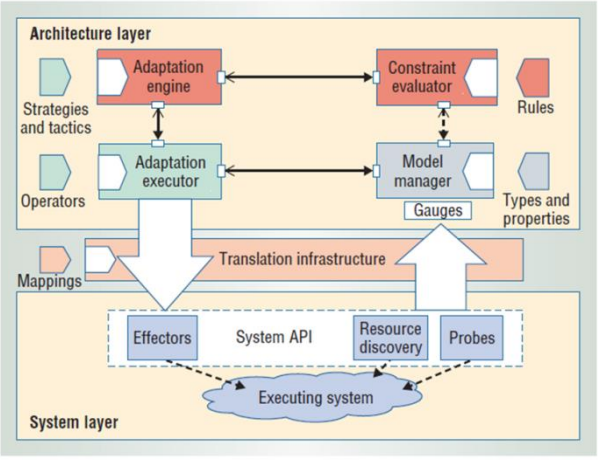


Figure 4 Rainbow Framework outline

Framework Outline

Figure 4 shows the Rainbow framework control loop for self-adaptation. Rainbow uses an abstract architectural model for monitoring runtime features of a running system, evaluates constraint violation model, and if there happens a problem, runs module level and global level adaptations on the running system.

System layer infrastructure: In this layer we have defined the access interface to the system and have created an infrastructure for its implementation. A system measuring mechanism is known as probes and

observers, and measure different parts of the system. This low level system's information can be printed by probes request. Furthermore, a resource discovery mechanism can be requested for new resources based on resource and other criterions. Finally an actuator mechanism does real corrections of the system.
Architecture layer infrastructure: In this layer measuring tool, gathers information from probes and updates proper features in architecture model. Model manager, provides access to system's architecture model and controls it. Constraint evaluator, checks the model periodically and in case of constraint violation, adaptation begins. Adaptation engine defines how to do it and does necessary adaptations.
Translation infrastructure: This infrastructure helps mapping information to model and vice versa, in whole system's abstraction gap. Translation repository in the infrastructure, keeps different mappings, in which the translator's components are participated in, for example, for translating an architecture level element ID to an address IP, or manager of architectural level changes to system level operations (Garlan, 2004).

Conclusion:

In this research we tried to present an adaptive ERP. In this regard, while introducing ERP and self-adaptive features, it has been tried to express and introduce adaptive frameworks in software systems. Among those frameworks we recognized Rainbow framework appropriate for implementing an ERP system for adaptation, as it has provided a context to develop other frameworks. For future work developing ERP system based on Rainbow framework can be investigated.

References:

Al-Mashari ,M. and Al-Mudimigh,A. and Zairi,M. (2003). Enterprise resource planning: a taxonomy of critical factors. European Journal of Operational Research, No. 146 ,Pp. 352–364

Dubois ,D. Joseph.(2010). Self-organizing Methods and Models for Software Development, Ph.D. Thesis, Dipartimento di Elettronica e Informazione, Politecnico di Milano.

Fan , Yuqing and Dong Mu and Xiaoqin Feng and LinYu (2008). Analysis of Supply Chain Adaptability Based on Self-organizing. Service Operations and Logistics, and Informatics. IEEE

Garlan ,D. and Shang-Wen Cheng and An-Cheng Huang and Bradley Schmerl and Peter Steenkiste.(2004). Rainbow: architecture-based self-adaptation with reusable infrastructure. IEEE Computer Society

Georgas ,J. C. and Taylor, R. N.(2004). Towards a knowledge-based approach to architectural adaptation management.

Hwang ,David and Ma Ga (Mark) Yang and Paul Hong (2015). Mediating effect of IT-enabled capabilities on competitive performance outcomes: An empirical investigation of ERP implementation. Journal of Engineeringand and Technology Management .No. 36, Pp. 1–23

Khalid ,A. and Mouna Abdul Haye and Malik Jahan Khan and Shafay Shamail (2009).Survey of frameworks, architectures and techniques in autonomic computing. Autonomic and Autonomous Systems, ICAS '09. Fifth International Conference on.

Krishnamurthy ,V. and Babu ,C.(2010). Effective self-adaptation by integrating adaptive framework with architectural patterns. A2CWiCAmrita ACM-W Celebration on Women in Computing in India .

Omicini ,A. and Ricci ,A.(2003). Reasoning about organisation: Shaping the infrastructure.

Yung-Chi Shen and Pih-Shuw Chen and Chun-Hsien Wang (2015).A study of enterprise resource planning (ERP) system performance measurement using the quantitative balanced scorecard approach. Computers in Industry, No. 75, Pp. 127–139

Elisabeth J. Umble and Ronald R. Haft and M. Michael Umble (2003). Enterprise resource planning: implementation procedures and criteria success factors. European Journal of Operational Research , No. 146 , Pp. 241–257

Yang ,Q. et al.(2010). Toward a fuzzy control-based approach to design of self-adaptive Software. Internetware conference.