Enterprise Software System Integration Using Autonomic Computing

Andrius Valatavičius[1], Saulius Gudas[1, 2]

[1] Vilnius University, Institute of Mathematics and Informatics, Software Engineering Department, Vilnius, Lithuania,
`valatavicius.andrius@gmail.com`
[2] Vilnius University, Kaunas Faculty of Humanities, Informatics Department, Kaunas, Lithuania,
`saulius.gudas@khf.vu.lt`

Abstract. Expansion of software systems in size and complexity during its lifetime business process models has to change to provide analytical insights, to improve and optimize business processes. Business process is always managed, self-adapting and evolving instance. But system management needs additional resources. This paper focuses on enterprise software system integration solutions that had been evolving but still lack of dynamic, automated and self - managed. We propose enterprise software system integration method using autonomic computing technologies to help solve most basic integration problems. We assume that autonomic computing element is perspective because its structure is similar to elementary management cycle. This similarity has been observed when analyzing different data structures of enterprise software systems.

Keywords: enterprise software system integration, elementary management cycle, autonomic computing, integration.

1 Introduction

Complexity of enterprise software systems is one of the major challenges information technologies facing today. Creating and maintaining enterprise software system interoperability is expensive. Integration solutions are rather complex and considered not always to be effective or successful even risky [6]. What is more integration solutions are managed by people that need to have deep level of knowledge about Enterprise software system structure and business processes. Businesses face certain process effectivity loss when software systems or business processes change for example double maintenance of the same data for different systems. The aim of this paper is to research approaches for Enterprise software system integration. Integration solution development were observed in practice. We assume that it is possible to create software system integration that could maintain integration solutions autonomously. The final result of the research is to create methodology that would allow to design and develop self-managed Enterprise software system integration solution using autonomic computing technologies. Some of the discussed approaches in this paper are a long way from dynamic, self-managed integration solutions, but they inspire research and discussion on the necessity of integration technology analysis and research. Is it worth to invest to full process, software system or infrastructure reengineering rather than small improvements that can enhance processes, information systems or infrastructure?

In this paper we propose approach to autonomic integration of enterprise software systems that cover four basic self-management abilities [4]: self-configuration, self-optimization, self-healing and self-protection. Our research answers question: Why Enterprise software system integration solutions need automation and how it can be achieved? In section 2 existing integration solutions and methods are identified. In section 3 we describe why we choose autonomic solutions. In section 4 we describe principles and key ideas on autonomic Enterprise software system integration. In section 5 we reach conclusions and describe further work in autonomic integration research area.

2 Related works of Enterprise system integration

There has been an inconclusive debate about whether existing enterprise software system integration solutions use domain knowledge about enterprise system structure and architecture in autonomic manner (can act autonomously).

El-Halwagi et al. Process integration [7] is prominent in the literature on process integration, mostly in Plant and Manufacturing domains. They underlined integration problems: resource cost, process flexibility, performance, and attention to quality that are also important for IT domain as well. Y. Peng et al. [9] focused on multi agent system for enterprise application integration to solve enterprise wide interoperability problems as well as planning and execution process separation problems. However they did not provide details on integration management and support. R. McCann et al. [12] underlined problems of mapping maintenance for data integration systems. X. Luna Dong, et al. [10] did a survey on data integration, underlined most common problems that exist in integration subject in 2006 is data heterogeneity and variety of data sources also in [20]. Authors described data fusion technique but to not describe it's managing and controlling actors. P.A. Bernstein et al. [5, 13] repetitive survey on data integration in two year timeline showed the improvements on integration subject and fosters a debate on what is still missing in integration area. Authors analyzed current trends on enterprise software system integration problems.

At the lowest level of software system integration there is data integration and schema matching in particular. Schema matching methods help to construct such algorithms that can link data-sources of different software systems. In Table 1 we provide comparison of different enterprise software system integration solutions. We compare common methods of software system integration by IBM autonomic maturity index (AMI) [4 pp. 12-15] and level of complexity over creating such integration systems.

Programmed integration solutions are most commonly spread (programmatically created integration solutions) doesn't have goals, usually are faster, easier to maintain (fix and adapt) for experts that created the solution, more expensive for customers, hard to maintain for new staff, hard to maintain knowledge about the solution. Not all projects are successful, and require varying amount of time to create healthy and robust solutions [6]. Table 1 shows most common and less innovative abilities of programmed integration solutions, but in practice solutions with higher levels of AMI can be achieved which are not so common.

Enterprise application integration (EAI) systems provides graphical designers. EAI systems does not require experienced programmers to create integration. Allows process orchestration and choreography in development stage. Is not dynamic after solutions has been implemented, usually need to be maintained, and integration processes sometimes has to be stopped. One of the advantages: designer can graphically see, and monitor processes. EAI projects are also limited to one project scope. Errors are detected and logged if implemented by design. Might contain or gather logs on integration process, logs are limited to the outputs of data sources, only small amount comes from the EAI system. It is believed that EAI solutions fail at 70 percent [6]. On critical error EAI solutions break and stops working, until fixed by integrators.

Table 1. Known enterprise software system integration methodologies and solutions.

		Algorithms (programming)	EAI Systems:	Integration agents	Other methods
Short List of known integration problems	Different data sources	Assigned manually	Talend SAP PI [22, 23]	[9]	
	Schema/Ontology mapping	Mapped manually		[9]	MAVERIC [12]
	Dynamic systems (schema changes)	Adapted manually		[9]	
	Entity/ Object recognition	Do not use		No information	
	Data - sampling	Manual	[22, 23]	No information	
	Data heterogeneity	Solved manually		No information	
	Integration testing	Manual testing	[22, 23]	No information	
	Independent from staff	Staff required for maintenance and supervision			Un-known
	CRUD logic	Done manually	Talend, SAP PI [22, 23]	Use Frameworks	
	Goal oriented architecture	Usually None		[9]	
List of maximally reached capabilities	Solution maintenance **dependent** from staff	All	All	[9]	MAVERIC [12]
	Self - abilities	None	None	Able to change parameters, react to predetermined changes	
	Autonomic Maturity Index [4]	1	2	3	4

Integration agents are the structures designed or created by someone that can be adapted and configurator to fit integration needs. For implementation and configuring integration agent, requires specific technical knowledge about integration agents. Some agents require extensive amounts of knowledge (mostly programming) to set up correctly [16]. Other agent might be readily prepared, but are not customized for specific business needs. Configured agents require adaptation if schema changed of any data source. After configuration agents can be very powerful tools, they can cooperate with other agents share information analyze data. Functionality may improve when connected to other agents. Some agents provide support of planning and execution via integration of processes, however unclear what steps are needed to implement this kind of agent technology [9], [17], [18]. Integration agents have goals. Knowledge element is usually described. Requires qualified experts to design, setup and/or configure. Usually implemented for specific scenarios or limited amount of processes (limited to project size). Does not contain or gather knowledge about business process or its behavior. Programmed integration solutions when broken requires technical and experienced staff to make amends. Programmed integration solutions usually focuses on the lower forms of integration: schema matching [5]. Not all integration agents can adapt to changes.

Other advanced solution provides methods for Dynamic process integration sometimes concerning for linking different businesses (B2B process integration) [0], [7, 8], [14, 15]. Such methods are solving major problems in business process integration, most of them are tested in the limited and fabricated environment, results of such methods were promising but they are mostly theoretical. Unknown what is the structure of given solutions, except in a semiotic approach to organizational modeling using Norm Analysis [8].

To conclude overview of existing methods other integration solutions from Table 1 does not reach highest AMI level and methods that are still researched (Other) barely scratch the surface of Adaptive level. The literature [5, 6], [14], [17] shows now consensus on increasing software system integration autonomic level, which means that most solutions does not use knowledge of business processes for creating integration solutions with self-abilities nor tries to reach autonomic maturity level.

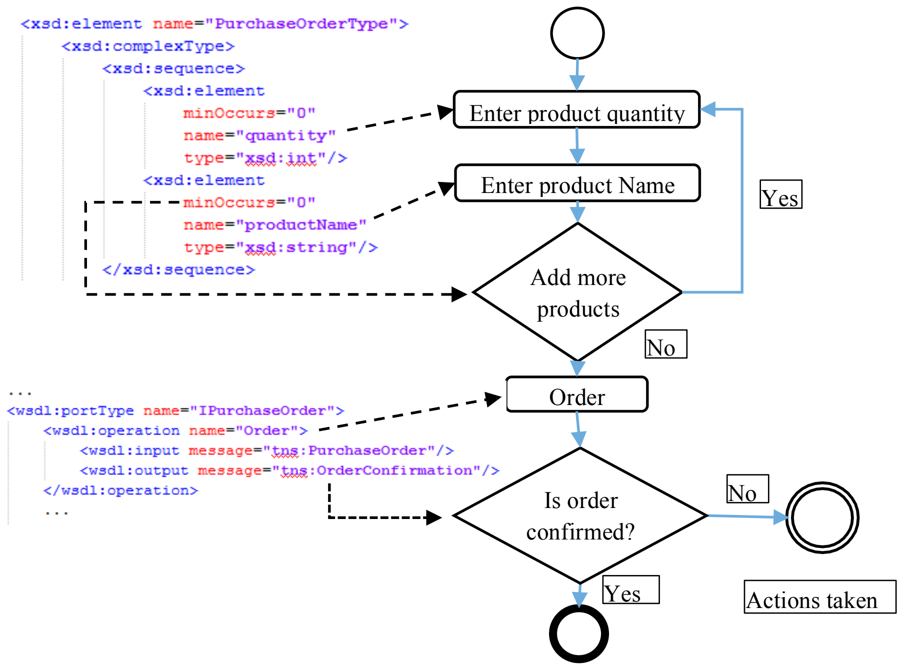
3 Integration solution using autonomic computing approach

Idea of Autonomic computing was first proposed by Jeffrey O. Kephart and David M. Chess in 2003 as means to deal with ever growing complexity of software systems [3]. Autonomic computing methods have goals, if reached highest autonomy index staff only needed to adjust business policies and goals. Ideally should not require technical people or technical knowledge when solution is created. May require staff for configuration: setting access points to data sources, unless autonomic computing elements would use method like service discovery; Autonomic computing is ideal for solving difficult large – scale tasks, just like agents autonomic computing elements can work in group, main difference is that autonomic computing elements have goals for productivity, defense, self-repairing.

The key idea behind autonomic computing component is that it has to have four abilities: Self-configuration; self-optimization; Self-healing; Self-protection. These four abilities are composite goals of autonomic elements. Autonomic elements help to decompose complex problem into smaller ones by providing Managed element to Autonomic manager. Autonomic manager itself contains four processes: Monitor, Analyze, Plan and Execute. The latter processes are interconnected and using knowledge to operate. One year later a practical guide for building Autonomic Computing systems was released [4]. Autonomic computing toolkit in detail described the composition of knowledge element of autonomic manager component. The whole process of single autonomic manager component was called Autonomic computing control loop. Connecting different autonomic computing control loops becomes very strong methodology created solves complex problems with certain level of autonomy. Autonomic maturity index describes levels of autonomy, there are 5 levels of autonomic maturity [4]: 1) basic, 2) managed, 3) predicted, 4) adaptive and 5) autonomic. To solve complex integration problems we selected autonomic computing technology because of previously mentioned “self” abilities and because we noticed the similarities in business process modeling and software system engineering subject [2]. Elementary management cycle is very similar in structure to autonomic element control loop, we believe that similarities of these components could help solving complex business process integration issues. More thorough analysis of such similarities is discussed Section 4.

4 Principles of Enterprise software system integration

Enterprise software system integration method enhanced with autonomic computing technologies analyze such sources of web services by monitoring their activity and analyze their schema documents. For example data similar to Order creation process chain can be extracted from the description file (WSDL) of a service to business process. Fig. 1 illustrates how WSDL



```
<xsd:element name="PurchaseOrderType">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element
                minOccurs="0"
                name="quantity"
                type="xsd:int"/>
            <xsd:element
                minOccurs="0"
                name="productName"
                type="xsd:string"/>
        </xsd:sequence>
```

```
...
<wsdl:portType name="IPurchaseOrder">
    <wsdl:operation name="Order">
        <wsdl:input message="tns:PurchaseOrder"/>
        <wsdl:output message="tns:OrderConfirmation"/>
    </wsdl:operation>
    ...
```

Fig. 1. WSDL schema of Order to business process alignment

schema could be mapped (aligned) to a process. If autonomic computing knowledge element contained business process knowledge (in form of models, formal descriptions like in *https://schema.org/Order*) then theoretically such integration method could be made self-aware and understand its context from business process perspective. In the illustrated example (Fig. 1.) keywords “Order”, “Confirmation” of process is determined. Autonomic solution analyzes elements, messages and operations from data sources. Requirements for registering purchase order are also given in WSDL schema in attribute named “minOccurs”. Using trial and error methods in simulated environment could help teach autonomic computing element about integrated environment creating ability to adapt to restrictions. On data transaction failure fault messages are return. These Fault messages are monitored and analyzed with autonomic computing element resulting in knowledge that data is incorrect or there are no such product registered (in this current scenario Fig. 1). If positive “OrderConfirmation” message returned autonomic element ends with state of successful integration.

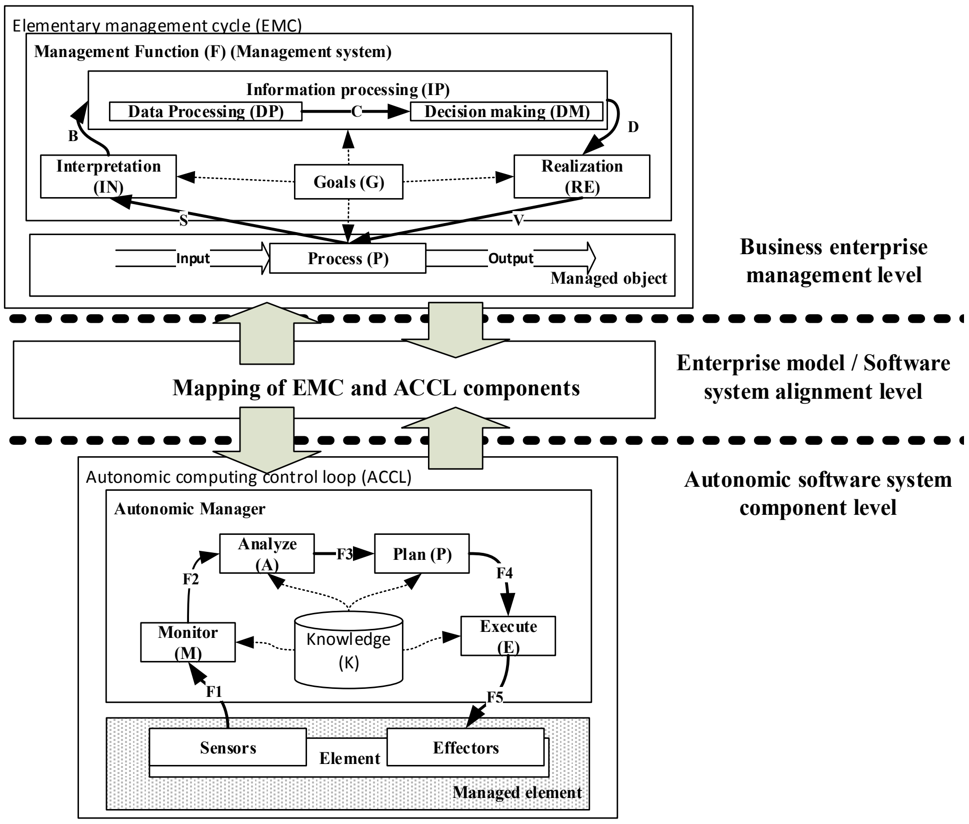

Elementary management cycle (EMC)
Management Function (F) (Management system)
Information processing (IP)
Data Processing (DP)
C
Decision making (DM)
B
D
Interpretation (IN)
Goals (G)
Realization (RE)
S
V
Input
Process (P)
Output
Managed object
Business enterprise management level
Mapping of EMC and ACCL components
Enterprise model / Software system alignment level
Autonomic computing control loop (ACCL)
Autonomic software system component level
Autonomic Manager
Analyze (A)
F3
Plan (P)
F2
F4
Monitor (M)
Knowledge (K)
Execute (E)
F1
F5
Sensors
Element
Effectors
Managed element

Fig. 2. Comparison of Autonomic computing component [3] (bottom) with Elementary management cycle [2] (top)

The elementary management cycle (EMC) is introduced in [2] as basic building block of enterprise management modeling from control theory point of view (Fig. 2). Management function (F) is complex structure which comprises a sequence of goal driven steps (information transformations): IN – interpretation, DP – data processing, DM – decision making, RE – decision realization, and **data/information flows** (S, B, C, D, V) between steps. Managed object is material transformation process (P) with inputs (materials, energy) and outputs (products, services), and is controlled by management function F. The EMC framework covers almost natural structure and behavior of enterprise management from data/information/knowledge/goal interactions viewpoint. The EMC framework and Autonomic computing control loop [4] are similar mainly because they were developed modeling real world behavior as knowledge and goal driven cycles of information processing. The similarities shown in Figure 2.

The components of Elementary management cycle and Autonomic computing components are those: a) Interpretation is related to Monitoring and Analysis –analyzing information from related objects got from information streams. b) Information management process is related to planning and executing elements because manipulations are done to information. c) Realization is more to execution but it is also related to planning in autonomic control loop. d) Goals are very closely related to knowledge. e) Both are responsible for controlling or managing lower level elements. f) Both are in the cycle of a loop showing that given methodologies are both dynamic and can exploit main agilities.

Similarities between the EMC autonomic computing elements give a promise that compatibility of EMC and autonomic computing component could really help to improve enterprise software integration methods.

5 Conclusions and Future Work

Our new approach suggests using autonomic management elements for commercial software system integration. Unlike other researched methods (dynamic integration, agent integration) our approach provides self-management capabilities, with possibility to implement self-healing, self-optimizing, self-configuration and self-protecting abilities. Our approach provides support for autonomic software system integration while linking software system layer with business process layer to gather knowledge. We compared similarities between elementary management cycle and autonomic manager component. We believe that because they are similar, autonomic computing methodologies can be used to analyze and integrate business processes. Underlined further research aims and goals: Design autonomic computing methodology to support existing best-practice solutions in schema matching, object matching choreography and orchestration; describe autonomic computing capabilities for integration decomposing and adapting its elements. Discovery of Enterprise software system layer being connected to business process layer provided further background for studying what exactly useful information could be gathered from business process that would help to automate software system integration solution creation.

This research should rise debate on further Integration evolution steps. Still a lot of work remains on defining relationship (mapping) between the smallest autonomic components used for integration and elementary management cycles (EMC) to fully grasp the knowledge of business layer impact on IT and software system layer. Defining such relationship would further help to partially automate integration solution developing and maintenance jobs.

Reference

1. A. Valatavicius, D. Dilijonas. Dynamic B2B process integration (in Lithuanian). In: Proceedings of "Informacinės Technologijos", pp. 34-39. Kaunas (2014)
2. S. Gudas. Foundations of the information systems' engineering theory (in Lithuanian). Vilnius University, Vilnius (2012)
3. J. O. Kephart, D. M Chess. The Vision of Autonomic computing. IBM Thomas J. Watson research center (2003)
4. B. Jacob, R. Lanyon-Hogg, D. K Nadgir, A. F. Yassin. A Practical Guide to the IBM Autonomic Computing Toolkit (2004)
5. E. Rahm, P. A. Bernstein. A survey of approaches to automatic schema matching. In VLDB. Volume 10, pp. 334-350 (2001)
6. G. Trotta. Dancing Around EAI 'Bear Traps'. In Business Process Management (BPM) Best Practices (2003)
7. El-Halwagi, M.M. Process Integration. Elsevier. ISBN-9780123705327 (2006)
8. S. Tan, K. Liu, Z. Xie. A semiotic approach to organizational modeling using norm analysis. In: Universityof Reading (2006)
9. Y. Peng, T. Finin, Y. Labrou, B. Chu, J Long, W.J. Tolone, A. Boughannam. A multi-agent system for enterprise integration. In: Proceedings of PAAM (1998)
10. A. Halevy, A. Rajaraman, J. Ordille. Data Integration: The teenage years (2006)
11. X. Luna Dong, F. Naumann. Data Fusion – Resolving data conflicts for integration. In: VLDB Endowment, pp 1654-1655 (2009)
12. R. McCann, B AlShelbi, W. Le, Hoa Nguyen, L. Vu, A. Doan. Mapping Maintenance for Data Integration Systems. In: VLDB (2005)
13. P.A. Bernstein, J. Madhavan, E. Rahm. Generic schema matching 10 years later. In: VLDB. Volume 4, (2011)
14. G Pavlin, M Kamermans, M. Scafeş. Dynamic Process Integration Framework: Toward Efficient Information Processing in Complex Distributed Systems. In: Proceedings of the 3rd International Symposium on Intelligent Distributed Computing (2009)
15. R-D. Kutsche, N. Milanovic. Model-Based Software and Data Integration. In: MBSDI (2008)
16. B.J. Overeinder, P.D. Verkaik and F.M.T. Brazier. Web Service Access Management for Integration with Agent Systems.In: SAC'08 (2008).
17. I. Zinnikus, C. Hahn, and K. Fischer. A Model-driven, Agent-based Approach for the Integration of Services into a Collaborative Business Process. In: AAMAS 2008, pp. 241-248 (2008)
18. L. Li, B. Wu, Y. Yang. Agent-based Ontology Integration for Ontology-based Applications. In: CRPIT (2005)
19. Pavel Shvaiko, J´erˆome Euzenat. Ontology matching: state of the art and future challenges. In: IEEE Transactions on Knowledge and Data Engineering (2013)
20. X. Luna Dong, F. Naumann. Big Data Integration. In: VLDB Endowment, pp. 1188-1189 (2009)
21. G. Hohpe and B. Woolf. Enterprise Integration Patterns. ISBN-13: 978-0321200686 (2011)
22. Talend Data Integration solution, https://www.talend.com/products/data-integration
23. Process Integration (PI) & SOA Middleware, http://scn.sap.com/community/pi-and-soa-middleware