Paper 7

by Paper Aradea

Submission date: 08-Oct-2020 04:09PM (UTC+0700)
Submission ID: 1408920932
File name: B3._Paper-2_AISC_2017.pdf (1,011.21K)
Word count: 4138
Character count: 23488

Integration of Self-adaptation Approach on Requirements Modeling

Aradea[1(✉)], Iping Supriana[2], Kridanto Surendro[2], and Irfan Darmawan[3]

[1] Department of Informatics Engineering, Faculty of Engineering, Siliwangi University, Tasikmalaya, Indonesia
aradea.informatika@gmail.com
[2] School of Electrical Engineering and Informatics, Bandung Institute of Technology, Bandung, Indonesia
[3] Department of Information System, School of Industrial and System Engineering, Telkom University, Bandung, Indonesia

Abstract. Self-adaptation approaches appear to respond to environmental complexity and uncertainty of today's software systems. However, in order to prepare the system with the capability of self-adaptation requires a specific strategy, including when conducting stage requirements modeling. Activity of requirements modeling to be very decisive, when selecting and entering new elements to be added. Here we adopt a feedback loop as a strategy of self-adaptation, which is integrated into a goal-based approach as an approach to requirements. This paper discusses the integration of the two approaches, with the aim of obtaining a new model, which has the advantages of both.

Keywords: Self-adaptive systems · Requirements modeling · Goal-based · Feedback loop · Rule-based systems · ECA rules

1 Introduction

Standish Group International [1], in the report shows that the challenge of software development continues to increase, reaching 43%, while employment requirements in capturing, selecting, and implementing a custom development applications is the most difficult activity. This is partly due to the requirements modeling for adaptation needs very different from the requirements for conventional requirements, which only aims to understand the problem domain, and is done only at design time [2]. Besides the cost of system maintenance continue developing States from 60% to 80% [3]. Thus, the issue of maintenance-related system configuration and reconfiguration of the system repeated has been a challenge that requires settlement.

These facts indicate that the software system must have the ability to adapt to dynamic environment. In order to meet these needs, it is important to establish a perspective that can guide us in understanding the domain and identify possible changes [4], which need to be predicted from the beginning. Therefore, modeling changes to the system must be controlled completely, including the evaluation of needs, especially related to the design of the control system [5], it can be met, one of

T. Herawan et al. (eds.), *Recent Advances on Soft Computing and Data Mining*,
Advances in Intelligent Systems and Computing 549, DOI 10.1007/978-3-319-51281-5_24

them during the activity of requirements modeling in order to realize an adaptability of the system independently, or known as self-adaptive systems (SAS). Activity of requirements modeling can be directed to arrest and formulate the needs of the system at design time, but is able to accommodate the needs at run-time. The second section of this paper discusses the related work, then on the third section we describe the proposed model, followed by a discussion of related work in section four, and we conclude this paper in section five.

2 Related Work

The focus of the research discussed in this section relate to the goal oriented requirements engineering (GORE) approach, based on a literature review and paper survey, this approach has had many successes as a basis for forming autonomous behavior adaptation needs. However, this approach to get its own challenges when it comes to meeting the needs of the nature of SAS. Table 1 shows the comparison of related research, in general model of GORE expanded by integrating elements that can bring the ability of self-adaptation. Among them, GORE collaborated with the principles of fuzzy logic as used in the model FLAGS and ADS-i*, this work focuses on addressing the uncertainty of the system, similar to LoREM, but the elements are adopted is model-driven approaches. Additionally, GORE approaches through an agent concept has been widely adopted, such as Tropos4AS, CARE, SOTA, GASD, STSs including Lorem, FLAGS, and ADS-i*, these works exploit the advantages of the concept of an agent to capture variability context and develop the behavior of self-configure. While the work GOCC and ZANSHIN, enter the control theory to bring a generic function feedback loops.

Table 1. Comparison of related work.

Model	Specification of design-time	Specification of run-time
LoREM [6] (Goal-based, Model-driven) Goldsby, 2008	i* model	Application-driven process
	Forth concept of requirements: goal, requirements, mechanism selection, infrastructure adaptation	Technology-driven process
GOCC [7] (Goal-based, Control theory) Naka, 2008, 2011	KAOS model	Control loop pattern (stimulus-respond)
	Modelling of configuration architecture generator (compiler): goal relation	Parser machine detector (new pattern and conflict)
	Three-layer (collect, analyze, act)	
FLAGS [8] (Goal-based, Fuzzy goal)	KAOS model and LTL	ECA rules
	Adaptive goal, run-time trigger	Supervision manager: mapping goal to BPEL

(*continued*)

Table 1. (*continued*)

Model	Specification of design-time	Specification of run-time
Baresi, 2010	Fuzzy goal, goal operator temporal	
ADS-i* [9] (Goal-based, Fuzzy goal) Serrano, 2011	i* model	Implementation BDI abstraction to JADEX
	Strategic dependency and rationale	
	Heuristic of i* abstraction into BDI	Reasoning machine: qualitative reasoning
	Fuzzy logic: task analysis and softgoal	
ARML [10] (Goal-based, Ontology) Qureshi, 2012	Techne model	ECA rules
	Integration of goal and shared ontology	Integration of PDDL3 and Hierarchical task network
	Context, resources, domain assumption	
GASD [11] (Goal-based, Model-driven) Wang, 2012	Prometheus model	OWL ontology
	Goal tree concept (selection algorithm): goal, plan, restriction model	Mechanism of knowledge based
	Designing of BDI architecture	JADE run-time
STS [12] (Goal-based, Ontology) Dalpiaz, 2013	Tropos/i* model	BDI agent
	Goal model, context model, plan specification, domain assumptions	Monitor-diagnose-reconcile-compensate
Zanshin [13] (Goal-based, Control theory) Souza, 2013	Techne model	PID controller
	Awarness requirements	Qualitative adaptation. Evolution requirements: ECA rules
	Variaton point (VP), control variable (CV), diferensial relation	
SOTA [14] (Goal-based, Natural language) Dhaminda, 2014	i* model and context-free grammar	Checking: labeled transition system analyzer.
	Goal pattern: achieve, maintain avoid	Mapping goal to event
	Precondition and postcondition goal, actor utility, entity and dependency	
Tropos4AS [15] (Goal-based, Model-driven) Morandini, 2015	Tropos/i* model	BDI agent
	Goal type, environment, failure	Transitions rules: goal state-intuition
	Environment and goal condition	
	Variability design	Inference rules

A variety of approaches are integrated self-adaptation to approach these goals, in addition to having the advantages of each still has some shortcomings, such as the principle of fuzzy logic and natural language constructions require approaches are quite complex and highly dependent on the specification requirements are very complete. In addition, the agency approach and model-driven substantially less can be directly mapped at run-time concepts related needs evolution cycle. While the application of control theory is very focused on the control feedback loop, in which the needs of the entity of the problem domain that is represented in the domain of models has not been included, as well as the need to determine variability and system independent, requires a special architectural design, so the need for further research to analyze the relationship. Inspired from the above description, arises an idea of how to formulate the mechanism of the process of adaptation of generic (providing a formal framework that can give rise generic function feedback loop for adaptation needs at run-time), but can accommodate variability context and behavior of real-world systems (through the ability of an agent in forming a knowledge of design time).

Here, we see loop feedback implementation has a promising chance if be equipped with data mining approach. A data mining algorithm can be functioned to face uncertain condition at run-time. these views become one of our future research concern to formulate a model that we are developing.

3 Integration of Self-adaptation Approach and Requirements

One approach GORE adopted in concept of requirements modeling are Tropos [16]. Stages requirements in Tropos consists of phases, (a) early requirements, a view to identifying the needs of stakeholders and how it relates to each other, (b) late requirements, capturing changes in a domain that is caused by the need for the system to-be and the true nature of the system. While abstraction of the model feedback loop which is used as the concept of self-adaptation, consists of monitors, analyzers, planners, executors, and knowledge (MAPE-K) [17]. Taking into account the context-aware scenario, MAPE-K implements architectural patterns ECA (event-condition-action), consists of three main components [18], which is a component context processor, the component controller, and component action performers, wherein (a) events, modeled and observed by one or more context processor, this component depends on the definition and modeling of context information, (b) condition, described the behavior of the application and observe the event, by empowering component of the controller is represented by the behavior description component, (c) action, triggered by action performers through a rule that has been specified.

3.1 The Concept of Integration

In order to realize the adaptability of the system, we steer stages in Tropos requirements may have the capability to monitor the variables of each goal decomposition of changes occurring, and can manage the changes at run-time (Fig. 1). This is done to supplement the capability requirements Tropos models, in capturing and representing variability of context and behavior of the system.

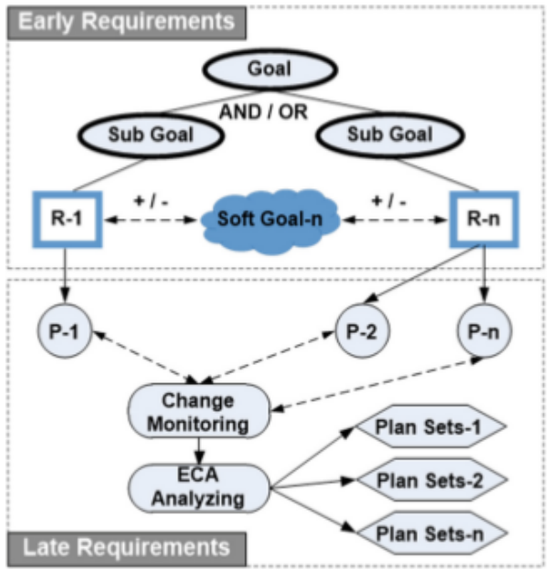

Fig. 1. Integration of self-adaptation approach on requirements modeling activity.

According to Zhuo-qun [19] in the modeling of i* model, variables can be derived from the actor's inner task [4], the task is an entity that can be detected and the source of the parameter type, if a task has uncertainty, then all tasks that have a relationship dependencies with the task of the need to be monitored, and when determining what is monitored, the values of the parameters can be used to represent them. Based on these opinions, if the theory is applied to the model Tropos then this task equivalent to the plan, in which each plan may have a dependency relationship with the goals, resources, or other notation. While monitoring the variable and handling needs of adaptation, the model was developed as a criterion of feedback loop mechanism guarantees that the process will be established.

Early requirements other than the function to identify the needs of stakeholders, this phase also aimed to capture the monitoring requirements of any entity that allows the change. Thus modeling the goal at this stage is expected to represent the needs of early to determine the mechanism of adaptation. Begins with decomposition AND/OR towards goal into sub-goals, identify requirements (R−1, R−2, R−n) of each goal that may change and affect the parameters, and has contributed a positive or negative (+/−) to one or more soft goals (non-functional) requirements, topped off by defining variables and parameters of each of these goals (P−1, P−2, P−n). In the late phase requirements, delegates goal in addition to capturing the needs of system-to-be, directed also to analyze the needs of adaptation. Starting with the restructure dependency goals, monitor changes that affect the parameters of each goal (P−1, P−2, P−n), analysis by ECA methods to analyze behavioral changes, and determine the variation of adaptation is based on the establishment of the rule which is defined as a plan (Plan Sets-1, Plan Sets-2, Plan Sets-n).

3.2 Case Illustration

Cases discussed is related to the adaptation needs of the system of lectures at universities. Where there is a class system actors who represent the needs of lecturer, students, and the program (prodi). The problem that occurs is the system must be able to ensure that teaching and learning activities can be held as scheduled. In addition, the number of face-to-face classroom sessions must achieve a minimum 14 and a maximum of 16 sessions, if it does not satisfy the range, it must be done through the establishment of additional tuition replacement schedule.

In the illustrative model of Fig. 2, hard goal or goal "to monitor the course" decomposed into two sub-goal of "organizing the lecture" and "cancellation of lecture" through OR-decomposition, meaning that if the lecture was held in accordance with the criteria, then the goal "cancellation lecture" does not need to be achieved but if otherwise, then the solutions developed is to determine the replacement schedule. Modeling for the replacement schedule need not be discussed in this paper, in the illustration Fig. 2 needs replacement schedule delegated to other actors through the goal "choose schedule". Goal "organizing the lecture" decomposed with AND-decomposition, the three sub-goals of the (start on schedule, completed on schedule, filling of attendance and the minutes) must all be achieved through the plan "monitoring" and an additional plan "check the number of lectures" for the goal "fill the attendance and the minutes", which contribute positively/full satisfaction (++) against soft goal "availability of classroom" and "lecture targets". While the plan for

Fig. 2. Modeling of lecture monitoring system.

achieving the goal "lecture cancellation" has contributed partial negative (−) against both soft goal, considering the cancellation of the lecture could adversely affect the achievement of the lecture and meeting room availability, but will not impact negatively if the determination of the replacement schedule can be achieved.

Based on the modeling, we can capture the issue of adaptation to be resolved, namely the third sub-goal of "organizing the lecture" is a goal that must be monitored, because the decomposition made an AND-decomposition, meaning that variable in every goal that is linked dependencies to each other. This can be represented by defining each parameter value in the plan that will be developed, namely by setting a rule to analyze aspects of dynamic behavior. For example the study program will be given a notification message when lectures in class was held or not held. Assumptions for the lecture was held, when the course of time according to the schedule. Based on the concept in the pattern of the ECA, we consider the situation "lecturer and students enter the classroom" as an event (e) that triggers an evaluation by the ECA rules:

```
If <lecturer and students enter the classroom (e1) AND
start as scheduled (c1) AND completed on schedule (c2)
AND filling attendance (c3)>, then <send notification
{prodi} (a1),"course achieved">
```

Elements of c1, c2 and c3 is a condition (c) to ensure that the lecture was held based on the criteria or not, as an example is late more than 10 min of schedule, was considered not fit the criteria. This condition represents a situation where the rules of action (a) will be activated, based on the condition (c) specified when the event occurs (when the "lecturer and students enter the classroom"), and a1 is "send notification (in the program)" is an action that is executed when a condition that occurs in accordance with the criteria or is true. Figure 3 illustrates the flow of information among the components in the pattern of ECA.

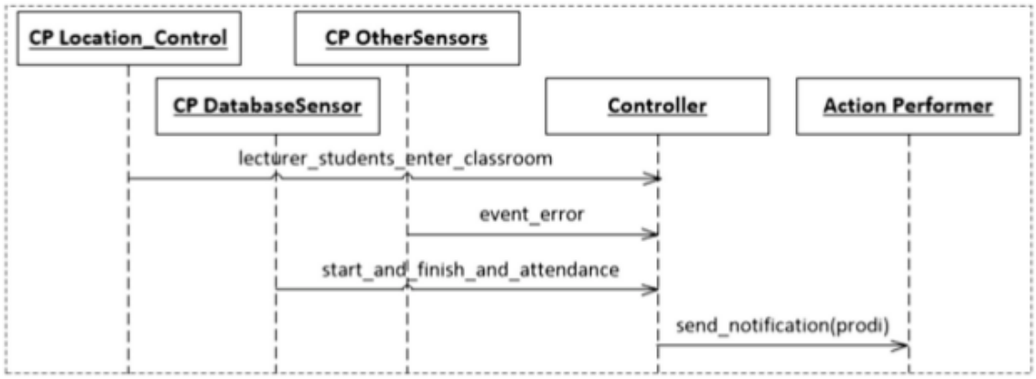

Fig. 3. The dynamic behaviour of ECA pattern.

The controller observed the occurrence of an event when lecturer and students enter the classroom, this event was captured by the location controller component, which is an instance of context processor (CP). With the use of sensors in the classroom, location controller can sense when lecturer and students entering the class. When this happens, the event "lecturer_students_enter_classroom" generated. Then, the controller evaluates "start_and_finish_and_attendance" and "event_error". Finally, if the third condition is true, the controller will trigger action "send_notification" which has been specified in the ECA, this action is the executed by the performers.

In the requirements activity, evaluation controller begins by defining variables that must be monitored, which is derived from the resource "course schedule". Based on the path dependencies, all the variables associated with "time" holding "course", i.e. start, finish, filling attendance and event news, as well as delay time tolerance. Tolerance delay time is context variables are added. In addition, to maximize the system's behavior, also added requirements to detect "errors" associated with "event" unexpected, e.g. sending error messages (msg_error) or lost connections (conn_error). Thus obtained variables are represented as: course_time and error_event = (course_time, event_error).

As for the needs analysis process, we should set the value of each parameter and the rules for the lecture can be considered established criteria or not. The action was carried out for both of these criteria is, if the lecture is held it will be sent a notification to the course that lectures held, and if the college is not established then the notifications are delivered is not the implementation of tuition based on a schedule, and the goal "choose the schedule" will be delegated to actors program a study to establish a replacement schedule. The criteria that the college is considered established, if the college began as scheduled, completed on schedule, charging absent and the minutes of a maximum of 15 min before the lecture is finished, and the delay tolerance for all these parameters is 10 min. Thus, the rule can be specified as in Table 2.

Table 2. Rule of course monitoring.

Rule	Statement
Rule-1:	*If (course_time = time.start and time.finish and time.attendance) and (event_error = null), then send notification {prodi}"course achieved"*
Rule-2:	*If (course_time = time.start and time.finish and time.attendance > 10 minutes) and (event_error = null), then send notification {lecturer} "check your teaching schedule"*
Rule-3:	*If (course_time = not time.start and time.finish and time.attendance) and (event_error = null), then send notification "course not conducted" and delegate goal to actor {prodi}*
Rule-4:	*If (course_time = time.start and time.finish and time.attendance) and (event_error = not null), then send notify user*
Rule-5:	*If not [event], then send notification "course not conducted" and delegate goal to actor {prodi}*
Rule-6:	*If not [criteria], then send notify user*

Based on these rules, it can be mapped into the ECA table (Table 3), so that there are four action as an alternative solution for the needs of adaptation. In order to define a comprehensive planning, determination of other parameters that can affect the system, as well as the formulation of reasoning mechanisms can be developed further.

Table 3. ECA of course monitoring.

Event	Condition	Action
"lecturer_students_enter_classroom"	*course_time = time. start and time.finish and time.attendance; event_error = null;*	*send notification {prodi} "course achieved"*
"lecturer_students_enter_classroom"	*course_time > 10 min; event_error = null;*	*send notification {lecturer} "check your teaching schedule"*
"lecturer_students_enter_classroom"	*course_time = not time.start and time. finish and time. attendance; event_error = null; not {event};*	*send notification "course not conducted" and delegate goal to actor {prodi}*
"lecturer_students_enter_classroom"	*event_error = not null; not {criteria} {msg_error};*	*send notify user*

4 Discussion and Future Work

The proposed framework when compared to the Tropos framework adopted, basically can be a complement to enhance the ability of adaptation requirements. When evaluated in general, the framework of this proposal can be developed as a generic framework that can be applied to various types of systems, in addition to the feedback loop automation through reconfiguration and evolution mechanism can improve the adaptability. So that this concept can reduce maintenance costs and increase flexibility in dealing with change factors for requirements engineering activities.

Comparison adaptation framework through two phases of requirements can be seen in Table 4. The work that has been done, is an initial foundation for the work ahead more specific, namely (a) identifies the need for an extension of the modeling language is adopted, taking into account the initial concept that has composed, (b) the alignment of the model which has been developed into a model that is adopted, to define the operational semantics that can accommodate the abilities of adaptation mechanisms, mainly dealing with strategy reconfiguration and evolution of software devices. Data mining approach become one of our future study, to determine appropriate time in adaptation, (c) evaluating the model in more detail to determine the level of success of approaches that have been proposed.

Table 4. Comparison of requirements modeling framework.

Phase	Tropos framework	Proposed framework
Early requirements	Stakeholder requirements • Description: actors, goals, plans, resources • Strategic dependency • Decompositions, means-ends, contributions	Adaptation requirements • Identify stakeholder requirements • Defining goals that could potentially change, as the requirements (R−n) • Determine the variable and parameters goals (P−n)
Late requirements	Delegating goal as a representation system-to-be • Introducing the actors • Specification of dependencies • Decompositions, means-ends, contributions to the actors	Delegating goal as adaptation requirements • Identify the cause of the goal parameters change (change monitoring) • Analyzing the behavior change • Determine the variation changes (ECA) • Determine the plan (plan sets-n)

5 Conclusion

Activity of requirements modeling at design time for the adaptation needs of the system, in essence proposes concepts and analytical techniques for designing monitoring needs and adaptation at run-time. Uncertainty at design time managed to combine design with specifications variability monitoring, through achievement criteria satisfaction goals. Here we propose an extension of an existing model. We see the feedback loop approach through ECA rules have the advantage to realize a generic function in reasoning at run-time. While Tropos models with the agent's concept has advantages in terms of capturing the variability of context and system behaviour, particularly related to the needs of the entity of the problem domain that is represented in the domain models.

We assume the integration of the two approaches can be complementary disadvantages and advantages of each. Therefore, we propose a framework as our preliminary study, and this approach still requires more in-depth study. Mainly deal with how to integrate a feedback loop with a centralized approach to the design needs of independent software-based agents. Currently, we are evaluating and formulating the system in more detail, including committing study toward alignment from both approach with formulating analysis technique using data mining at sensor data through data history, and context inference in determining context needs as run-time at the knowledge base.

References

1. The Standish Group: Chaos Manifesto: Think Big, Act Small. The Standish Group International (2013)
2. Perini, A.: Self-adaptive service based applications: challenges in requirements engineering. In: RCIS, CIT-Irst., FBK, Trento, Italy (2012)

3. Sherry, J., Hasan S., Scott, C., Krishanmurthy, A., Ratnasamay, S., Sekar, V.: Making middleboxes someone else's problem. In: Proceedings of the ACM SIGCOMM 2012, New York, USA, p. 13. ACM Press (2012)
4. Aradea, Supriana, I., Surendro, K.: Prinsip Paradigma Agen Dalam Menjamin Keberlangsungan Hidup Sistem. Prosiding KNSI. Universitas Klabat Sulawesi Utara (2015)
5. Aradea, Supriana, I., Surendro, K.: An overview of multi agent system approach in knowledge management model. In: Proceedings of the ICITSI, STEI, ITB (2014)
6. Goldsby, H.J., Cheng, B.H.C.: Automatically generating behavioral models of adaptive systems to address uncertainty. In: Czarnecki, K., Ober, I., Bruel, J.-M., Uhl, A., Völter, M. (eds.) MODELS 2008. LNCS, vol. 5301, pp. 568–583. Springer, Heidelberg (2008). doi:10.1007/978-3-540-87875-9_40
7. Nakagawa, H., Ohsuga, A., Honiden, S: GOCC: a configuration compiler for self-adaptive systems using goal-oriented requirements description. In: Proceedings of the 6th International Symposium on SEAMS, May 21–28, pp. 40–49. ACM, USA (2011)
8. Baresi, L., Pasquale, L., Spoletini, P.: Fuzzy goals for requirements-driven adaptation. In: Proceedings of RE, pp. 125–134. IEEE (2010)
9. Serrano, M., Sampaio, J.C.: Development of agent-driven systems: from i* architectural models to intentional agents' code. In: CEUR Proceedings of the International i* Workshop (2011)
10. Qureshi, N.A., Jureta, I.J., Perini, A.: Towards a requirements modeling language for self-adaptive systems. In: Regnell, B., Damian, D. (eds.) REFSQ 2012. LNCS, vol. 7195, pp. 263–279. Springer, Berlin (2012). doi:10.1007/978-3-642-28714-5_24
11. Wang, T., Li, B., Zhao, L., Zhang, X.: A goal-driven self-adaptive software system design framework based on agent. Phys. Procedia **24**, 2010–2016 (2012). ICAPIE Organization Commite, Elsevier B.V.
12. Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Adaptive socio-technical systems: a requirements-based approach. J. Requirements Eng. **18**(1), 1–24 (2013)
13. Souza, V.E.S., Lapouchnian, A., Angelopoulos, K., Mylopoulos, J.: Requirements-driven software evolution. J. Comput. Sci. Res. Dev. **28**(4), 311–329 (2013). Springer
14. Dhaminda B., Hoch, N., Zambonelli, F.: An integrated eclipse plug-in for engineering and implementing self-adaptive systems. In: 23rd International WETICE. IEEE (2014)
15. Morandini, M., Penserini, L., Perini, A., Marchetto, A.: Engineering requirements for adaptive systems. J. Requirements Eng. **21**, 1–27 (2015). Springer
16. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: TROPOS: an agent-oriented software development methodology. J. Auton. Agent. Multi-Agent Syst. **8**(3), 203–236 (2004)
17. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Comput. **36**(1), 41–50 (2003)
18. Daniele, L.M.: Towards a rule-based approach for context-aware applications. Master thesis. University of Cagliari, Italy (2006)
19. Zhuo-Qun, Y., Zhi, J.: Requirements modeling and system reconfiguration for self-adaptation of internet-ware. In: Fourth Asia-Pacific Symposium on Internet-ware. ACM (2012)

Paper 7

ORIGINALITY REPORT

8%	4%	6%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.semanticscholar.org Internet Source	2%
2	Submitted to Universiti Teknologi Malaysia Student Paper	2%
3	Iping Supriana, Kridanto Surendro, Aradea, Edvin Ramadhan. "Self-adaptive cyber city system", 2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA), 2016 Publication	2%
4	Zhuo-Qun, Yang, and Jin Zhi. "Requirements modeling and system reconfiguration for self-adaptation of Internetware", Proceedings of the Fourth Asia-Pacific Symposium on Internetware - Internetware 12 Internetware 12, 2012. Publication	1%
5	Wiwin Suwarningsih, Ayu Purwarianti, Iping Supriana. "Chapter 30 Dependency Scheme for Revise and Reasoning Solution", Springer Science and Business Media LLC, 2017 Publication	1%
6	Riswan Efendi, Noor Azah Samsudin, Nureize Arbaiy, Mustafa Mat Deris. "Fuzzy random auto-regression time series model in enrollment university forecasting", 2017 5th International Symposium on Computational and Business Intelligence (ISCBI), 2017 Publication	1%

Exclude quotes	Off	Exclude matches	< 1%
Exclude bibliography	On		