Context-aware Adaption of Software Entities using Rules

Lauma Jokste and Jānis Grabis
Information Technology Institue, Riga Technical University, Kalku 1, Riga, Latvia

Keywords: Software Entity, Context-awareness, Rule-based Adaptation, Adaptation Requirements.

Abstract: Context-aware systems gain recognition in rapidly growing information systems market. Systems run time adaption based on contextual information have been considered as a powerful mean towards better systems performance which help to reach overall organizational goals and to improve key performance indicators. This paper describes the concept where information systems can be divided into many software entities and each of them can be context dependent. Context situation dependent software entity execution routines are observed and these observations are used to formulate Context dependency rules either manually or by machine learning. Rule based adaptation allows to monitor adaptation process in a transparent way and allows to take into account human knowledge in adaptation process. The entity based adaption allows for a uniform approach inducing context-dependency to different part of the software.

1 INTRODUCTION

Enterprise applications such ERP systems are used to execute business processes. They provide a large number of different functions and are characterized by high degree of complexity. These applications are typically packaged applications and often need to be tailored for needs of specific users. However, their complexity reduces modification flexibility, and achieving the right fit between functionality provided and user needs is challenging. As the result, the enterprise applications do not always yield expected benefits (Calisir and Calisir, 2004) and suffer from poor usability (Singh and Wesson, 2009).

Adaption of applications during their usage is one of the solutions to improve performance of enterprise applications (Macías-Escrivá et al. 2013). Existing research towards adaptive enterprise applications is limited to specific types of adaptation, such as user centric (Shakshuhi et al., 2015; Lavie and Meyer, 2010), data base recommendations (Rubens et al., 2011), workflow adaption (Uchibayahi et al., 2012), process-based service provisioning (Pernici, 2007). In the case of enterprise applications with a wide-scope such an approach is infeasible because of the effort associated with development of adaptive behavior. However, enterprise applications are well-structured and consist of standardized objects such as classes, forms and reports. Each of them can be described by several characteristics. These objects are referred as to software entities (SEs) and adaptive behavior could be standardized as a set of typical adaption actions performed with the SEs. Adaption is usually performed in response to changes in operating circumstances and is aimed to satisfy specific goals (Šūpulniece and Grabis, 2015). The operating circumstances can be represented by contextual information where context is defined as any information affecting the SE (Dey, 2001). In this paper, it is assumed that adaption is invoked in response to changes in context.

The objective of this paper is to introduce the concept of context-aware adaption of SEs constituting enterprise application and to describe the overall architecture of Context-aware software entity adaptation (CASEA) system. Adaptive behavior is defined using adaptation rules combining SEs, context and adaptation actions. Adaptability is delivered in a form of recommendations to avoid obstruction of business process execution. Quality of the adaption rules and recommendations made is also evaluated. The main distinctive feature of the proposed approach is that adaption is performed in a uniform manner for all SEs constituting the enterprise applications and adaption is externalized without affecting development and maintenance of key functionality.

The paper is structured as follows. Section 2 reviews related works and formulates requirements

Jokste, L. and Grabis, J.
Context-aware Adaption of Software Entities using Rules.
DOI: 10.5220/0006366401660171
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 3*, pages 166-171
ISBN: 978-989-758-249-3

towards the adaptive system. The overall architecture of CASEA is described in Section 3. Section 4 presents the structure of Adaptation rules. Section 5 concludes.

2 FOUNDATIONS

The proposed research is motivated by a need to improve efficiency of a suite of enterprise applications used mainly by municipalities. These enterprise applications provide a wide range of services to a diverse group of users. The software vendor has observed that there is relatively large number of user support requests, some functionality is used only to limited extent and many change requests could be accommodated by existing functionality. These issues could be addressed by using adaptation. Additionally, there is a potential for knowledge exchange between users, user groups or municipalities because some municipalities have significant in-house experience while others have very limited information technology resources.

In order to identify ways for providing adaptability, related research is reviewed and requirements are formulated according to the review and industrial experiences.

2.1 Related Work

Design of an adaptive systems deals with multiple issues. These include the overall design of the adaptive system, context processing and definition and execution of adaptation policies.

The design of adaptive systems is guided by the classical MAPE architecture (Andersson et al., 2009). This architecture distinguishes the managed subsystem (i.e., enterprise applications), managing subsystem (i.e., adaptation module) and environment. The managed and managing sub-systems jointly make up a self-adaptive software system. The Music development framework (Hallsteinsen et al., 2012) uses the MAPE approach to create variants of a software application depending on the current context. Adaption is performed according to the runtime adaption model. Adaption modules are also a part of runtime models based approach for developing dynamically adaptive systems (Loukil et al. 2017). Both approaches decouple the adaptation module from other parts of the system and focus on developing new adaptive applications. Alferez et al. (2014) elaborates a framework for adaptive service composition. Adaption is performed by the Model-based reconfigurator according to context information using the Variability model, which defines various features of the services used in composition and serves as a basis for formulating adaptation policies. Verification of the adaptation results in an important part of the framework.

In the area of dynamically evolving information systems adaptive and context-aware systems gain rapidly growing popularity. Two basic forms of context data are raw or low level context data obtained directly from sensors and rich or high level data where some qualitative data processing has been applied (Dey, 2000). Just a single context factor is often taken into account, e.g., location (Pils et al., 2006), personal context (Dey, 2000). In enterprise applications perspective combination of context data can give a significant contribution.

Yan et al. (2012) have proposed an autonomic middleware framework which focuses on qualitative context processing. Ashbrook and Starner (2003) were one of the first researchers introducing the context prediction technique where users' next location was predicted gaining information from GPS. Lately, context prediction also plays an important role in context-aware adaptive information systems by applying prediction algorithms such as Collaborative-based Context Prediction (Voigtmann et al., 2011).

Salehie and Ladan (2012) develop adaptive applications by using the Goal-Action-Attribute model. The model specifies that attributes are used to measure performance of adaptive applications, goals define expectations towards adaptation and actions represent possible changes in software. The actions can be of various type though the papers focuses on autonomic computing. Association rules (Tan et al., 2006) is another technique used for defining adaptation policies. Decisions and behavior represented by the association rules are often context dependent. Association rules mining is used to find relations between contextual information and problems occurred in information systems (Liu et al., 2008). Rules in adaptive systems can be defined manually or generated automatically by machine learning methods (Kolski et al., 1997). Yang et al., (2013) emphasize the importance of contextual factors that affects users' preferences. Context-based inference rules discovered from association rules mining are introduced which allows to identify frequent associations between situation features and relevant content characteristics (Ke and Liu, 2011). Adaptation rules are also used as a basis for executing adaptive actions focusing on business process adaptation (Carvalho et al., 2015).

The aforementioned papers focus on association rules mining techniques and lack information about usage of obtained rules in general and specifically in the case of enterprise applications. Also adaptation process is oriented to specific software instances, such as business processes adaptation. The approach described in this paper focuses on the end-to-end enterprise applications adaptation process starting from identification of context dependent SEs, defining context dependencies by association rules mining and continuing with run-time adaption which is made based on observed context situations and corresponding context dependency rules. The feedback about usage of the rules is also collected to improve credibility of the defined rules.

2.2 Requirements

The surveyed related work provides basis for formulating requirements towards the CASEA approach. Also the authors experience with municipalities' information systems and users' requirements towards a better usability of the system has been taken into account. The requirements are:

1. The MAPE loop is used to guide the adaptation with clearly identifiable managed and managing sub-systems as well as context monitoring system.
2. The adaption module is decoupled from the core parts of the enterprise applications to avoid modification and disruption of the key business functions. It should be scalable to enable processing of large data amounts.
3. Adaptation process should not focus on specific software instances and should be applicable for different kind of SEs following a uniform design.
4. Behavior of SEs is altered by a set of well-defined adaption actions to ensure predictable adaption results and to reuse adaptation actions across different SEs.
5. The adaptation actions are triggered by changes in application execution context including business and technical context and a context module is used for pre-processing contextual data coming from various sources.
6. The adaptation process should be able to learn from itself and information about application usage should be exchanged across the platform to achieve knowledge sharing efficiencies.
7. Adaptation should be done in an unobtrusive way for application users.

Given the size of enterprise applications, the requirements specifically target development, reuse and scalability issues. Effort required for developing and maintaining adaptive components should be outweighed by adaption gains. The adaptive components should be developed in a uniform manner to reduce complexity.

3 CASEA ARCHITECTURE

It is assumed that application consists of software entities of different granularity and type. The SE is defined as an executable software artifact which can be either a software as a whole or a single workflow, procedure, job, data input field or even a data value of classifier. Software division in SEs can be represented as an object tree following the principles of software ontology (Kiefer et al., 2007). A SE from a lower level can inherit context dependencies from the upper level.

There is an open set of adaptation actions which can be performed on every entity. Adapting a system at the SE level allows to adjust it to specific context situation and specific user needs more precisely. The architecture described in this section is based on the Recommendations metamodel (Jokste, 2015). The metamodel implies that each SE has a context set which includes all the necessary information about context affecting the entity.

In order to create the architecture supporting the adaptation process based on contextual information, several requirements have been determined by analyzing existing enterprise applications execution, users' activity data and log files from audit protocols by applying different data mining and association rules mining techniques. Requirements coincide with those summarized in Section 2.2.

We came to conclusion that CASEA should be built as an independent data base module where new incoming context sources and elements, Context dependency rules and Adaptation rules could be added or removed without changing software code (Fig.1). The rules should contain information about SE, such as type, name, application etc. The module would allow to choose an appropriate adaptive action for each SE for the certain context situation.

CASEA consists of three core parts – input part, execution of the module and output part. The main input data are contextual information which can be received from different internal or external data sources with a certain regularity. Received context information is then saved in context data base. If raw context data is received, context element values normalization process should be done in order to convert context data into usable format. Replication mechanism is defined for context database. Each time when new context information is received, replication mechanism calls the rules compliance checking procedure which checks if there exists Context dependency rules which correspond to the received context element values. If compliance is found, then the appropriate Adaptation rule is executed. Execution of Adaptation rules can be also initiated by users' action, e.g. if user enters the system, replication mechanism checks the current context information and verifies compliance with Context dependency rules.

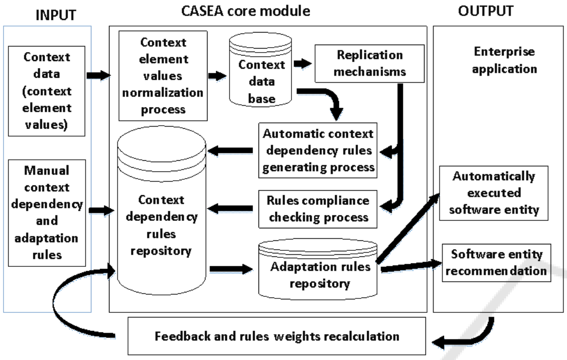
INPUT
CASEA core module
OUTPUT
Context data (context element values)
Manual context dependency and adaptation rules
Context element values normalization process
Context data base
Replication mechanisms
Automatic context dependency rules generating process
Context dependency rules repository
Rules compliance checking process
Adaptation rules repository
Enterprise application
Automatically executed software entity
Software entity recommendation
Feedback and rules weights recalculation

Figure 1: Context-aware SE adaptor module.

It is important to gather knowledge about the quality of the defined or generated rules. That is achieved by continuous feedback monitoring process based on which rules weights can be recalculated. The feedback monitoring process includes recording users' response to adjustments made.

4 ADAPTATION PROCESS

Software consists of a set of SE which can be designated as SE, thereby the software (set of SE) can be designated as SE = $\{se_1, se_2, se_3, \ldots, se_n\}$. Each SE can be dependent of one or many context element one or many values. One context element can affect one or several SE. All context elements affecting the context-aware software are designated as CE = $\{ce_1, ce_2, ce_3, \ldots, ce_n\}$.

CASEA is rule-based and includes two different kinds of rules. Context dependency rules define context ranges which affect the SE and Adaptation rules which initiate adjustments appropriate for the certain run-time context situation. Context dependency rules can be expressed as context-based association rules where execution of SE related adaptive action associates to a particular context situation. Adaptation rules can be expressed as event-condition-action (ECA) rules and designates the adaptation action in a context situation.

Due to the fact that context element values should be taken into account when defining Context dependency rules, association rules should be extended by including context element values, thus simple Context dependency rule is defined as **$se_n \Rightarrow ce_n(v)$, where $se_n \in SE$ and $ce_n \in CE$**, meaning that SE se_n associates with context element ce_n, where variable v indicates the context element value or values interval of the context element measurable property on which se_n depends. The quality of the rule is represented by weights RW = $\{rw_1, rw_2, \ldots, rw_n\}$. The rule weight describes the impartiality of one single or multidimensional Context dependency rule. If one context situation is favorable to several SE of the same kind, then SE with the highest weight can be executed. The rule weights can be recalculated in run-time based on several events, such as the number of times rule compliance is found, users' feedback etc. Context dependencies might consist of several context value combinations, thereby multidimensional form of association rules should be used.

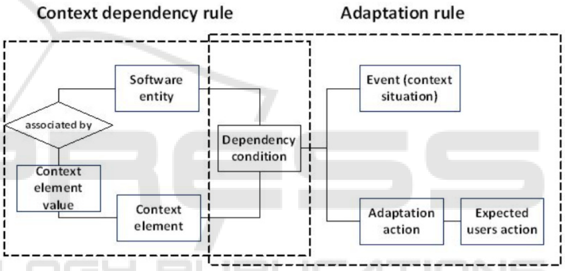

Figure 2: Interrelationship between Context Dependency and Adaptation Rules.

Adaptation rules define the adaptation task which should be executed when context situation defined in Context dependency rules occurs. For adaptation rules we chose the form of ECA rules (also known as reaction rules and post-condition rules) where event is a run-time context situation, condition is context element values (certain values or value intervals) and action is an executable adjustment (SE which can be executed in different ways). To ensure quality of the adaptation process, the Adaptation rule should include information about expected action of the user when Adaptation rule is executed. If a user has received a recommendation based on current context situation to execute a certain process in application, system expects the user to execute this process. If the user follows the recommendation, then it has been useful and thereby the Adaptation rule is considered as apposite. Connection of Context-dependency and Adaptation rules is shown in Fig.2.

5 APPLICATION EXAMPLE

CASEA is intended for contextualizing large scale enterprise applications. In this section we demonstrate a shot example case from the Single Municipalities System (SMS) developed by the Latvian company ZZ Dats Ltd. SMS includes different mutually integrated applications providing IS support for implementation of municipalities' functions and public electronical services for citizens and companies.

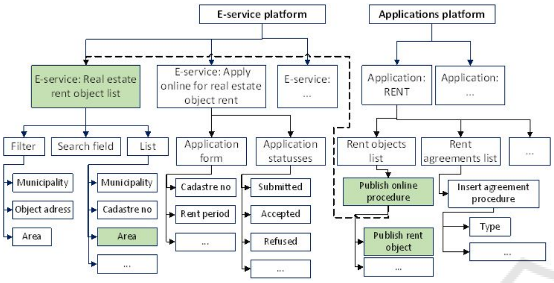

Figure 3: Fragment of software division into SEs

Fig.3 demonstrates a fragment of SMS division into SEs. Context dependency and Adaptation rules are defined for the selected highlighted SEs. Context elements affecting the execution of SE should be identified. Both – personal and situational context can be taken into account. For instance, lessee type can affect the execution of several SE related to rent objects. If number of lessee agreements >10 and total rented area > 10000 m^2, then lessee profile is "active". If Context-dependency rules are defined manually then the initial rule weights are also defined. Context dependency rules examples:

1) E-service: Real estate rent object list⇒ lessee profile ('active')
2) List column: area⇒ lessee profile ('active')
3) Search field⇒ time spent in object list(>180 sec)
4) Publish online procedure⇒ unsuccessful searches per hour (>10)

Adaptation rules examples:

1) IF lessee profile='active' THEN highlight rent objects in list WHERE rent object area≥30000 m^2
2) *IF lessee profile='active' THEN automatically order list by area column descending*
3) IF time spent in object list >180 sec THEN highlight search field
4) *IF unsuccessful searches per hour>10 THEN automatic e-mail/text notification to RENT user/-s.*

When Adaptation rule is executed, users' response is expected, for instance, if lessee profile is 'active' and rent object list is automatically ordered by area descending, then the e-service user is expected to apply for object with a large area or at least to open the detailed information of object with large area. If CASEA systems feedback monitoring process identifies that the user executes the expected action than the corresponding association rule weight is increased. The execution of expected action is fixed in audit protocols, thus CASEA system can compare the users executed action with predefined expected action.

6 CONCLUSIONS

In this paper we have introduced a conceptual architecture of CASEA which allows to adjust enterprise applications at the SE level based on contextual information. The adaptation process is carried out in a form of rules execution thus allowing to account for human factors. This paper demonstrates the work in progress and marks the future tasks and challenges. Approach conceptually described in this paper will be further evolved and applied for enterprise applications that are used by municipalities of Latvia. Context data will be mainly extracted from users' action logs, error protocols and also calendar events. Future work includes the following aspects: (1) To analyze the SEs context dependencies based on municipalities systems users' action logs; (2) to expand the method for defining context dependency and adaptation rules; (3) to develop the method for adaptation process validation and evaluation of its effectiveness. (4) to develop the CASEA module; (5) to validate and evaluate the CASEA module and adaptation approach which is conceptually described in this paper.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the research project "Competence Centre of Information and Communication Technologies" of EU Structural funds, contract No. 1.2.1.1/16/A/007 signed between IT Competence Centre and Central Finance and Contracting Agency, Research No. 1.6 "Development of Software Adaptation Algorithms and Module Based on Context Information Extracted from Users Action Logs".

REFERENCES

Alferez, G.H., Pelechano, V., Mazo, R. and Salinesi C.,

2014. Dynamic Adaptation of Service Compositions with Variability Models. Journal of Systems and Software, 91, 24-47.

Andersson, J., de Lemos, R., Malek, S., Weyns, D. 2009. Reflecting on self-adaptive software systems. In: Proc. of the ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems, pp. 38–47.

Ashbrook D., Starner T., 2003. Using GP to Learn Significant Locations and Predict Movement Across Multiple Users. Personal and Ubiquitous Computing, 7(5), 275-286.

Calisir, F. and Calisir F., 2004. The relation of interface usability characteristics, perceived usefulness, and perceived ease of use to end-user satisfaction with enterprise resource planning (ERP) systems. Computers in Human Behavior, 20, 4, 505-515.

Carvalho, J.E.S., Santoro, F.M. and Revoredo, K., 2015. A method to infer the need to update situations in business process adaptation. Computers in Industry, 71, 128-143.

Dey, A.K., 2000. Providing architectural support for building context-aware applications. PhD thesis, Georgia Institute of Technology.

Dey, A.K., 2001. Understanding and Using Context. In Personal Ubiquitous Computing, vol 5(1), pp. 4–7.

Hallsteinsen, S., Geihs, K., Paspallis, N., Eliassen, F., Horn, G., Lorenzo, J., Mamelli, A., Papadopoulos, G.A., 2012. A development framework and methodology for self-adapting applications in ubiquitous computing environments, Journal of Systems and Software, 85, 12, 2840-2859.

Jokste, L., 2015. Towards a Model of Context-aware Recommender Systems. In Proceedings of the CAiSE 2015 Forum at the 27th International Conference CAISE, vol 1367, pp 145-152.

Ke C.K and Liu D.R., 2011. Context-based knowledge support for problem-solving by rule-inference and case-based reasoning. International Journal of Innovative Computing, Information and Control, 7.

Kiefer, C., Bernstein, A. and Tappolet, J., 2007. Mining Software Repositories with iSPARQL and a Software Evolution Ontology. In Mining Software Repositories, ICSE Workshops MSR ’07.

Kolski, C., Le Strugeon, E., Tendjaoui, M. 1993. Implementation of AI techniques for “intelligent” interface development. Engineering Applications of Artificial Intelligence, 6, 295-305

Lavie, T. And Meyer, J., 2010. Benefits and costs of adaptive user interfaces, Int. J. of Human-Computer Studies, 68, 508-524.

Liu, D.R., Ke, C.K. and Wu M.Y., 2008. Context-based Knowledge Support for Problem-solving by Rule-inference and Case-based Reasoning. In Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, pp 32015- 3210.

Loukil, S., Kallel, S., Jmaiel, M., 2017. An approach based on runtime models for developing dynamically adaptive systems. Future Generation Computer Systems, 68, 365-375.

Macias-Escriva F.D., Haber R., Toro, R. and Hernandez V., 2013. Self-adaptive systems: A survey of current approaches, research challenges and applications. In Expert Systems with Applications, 40(18), 7267-7279.

Pernici, B., 2007. Adaptive Information Systems. Conceptual Modeling in Information Systems Engineering. Eds. J.Krogstie, A.L. Opdahl and S.Brinkkemper, pp.295-304.

Pils, C., Roussaki, I. and Strimpakou, M., 2006. Location-Based Context Retrieval and Filtering, vol. 3987 of the series Lecture Notes in Computer Science, pp 256-273.

Rubens, N., Kaplan, D. and Sugiyama, M., 2011. Recommender Systems Handbook: Active Learning in Recommender Systems (eds. P.B. Kantor, F. Ricci, L. Rokach, B. Shapira). Springer, pp 735-767.

Salehie, M., Tahvildari L., 2012. Towards a goal-driven approach to action selection in self-adaptive software. Software Practice & Experience, 42, 2, 211-233.

Shakshuhi, E.M., Reid, M., Sheltami, T.R., 2015. An Adaptive User Interface in Healthcare, In: Procedia Computer Science, vol 56, pp 49-48.

Singh, A. and Wesson, J., 2009. Improving the Usability of ERP Systems through the Application of Adaptive User Interfaces. In Proceedings of the 11th International Conference on Enterprise Information Systems (ICEIS), vol SAIC.

Šūpulniece, I. and Grabis, J., 2015. Conceptual Model of User Adaptive Enterprise Application. CSIMQ, 3, 84-96

Uchibayahi, T., Bernady, O.A. and Shiratori N., 2012. Towards an Adaptive Workflow with Multi-Agents in a Semantic Grid. In 12th International Conference on Computational Science and Its Applications, pp 20-25.

Tan, P.N., Steinbach M. and Kumar V., 2006. Introduction to data mining: Association Analysis: Basic Concepts and Algorithms, pp 327-414.

Voigtmann, C., Lun L.S. and Klaus D., 2011. A Collaborative Context Prediction Technique. In: Vehicular Technology Conference, IEEE, pp 1-5.

Yan, H., Zheng, D. and Wang, J., 2012. Research of Quality Based Autonomic Context Processing for Pervasive Applications. In 7th International Conference on Computer Science & Education, pp 1234 – 1238.

Yang W., Liao, Q., Zhang, C., 2013. An Association Rules Mining Algorithm on Context-Factors and Users’ Preference. In International Conference on Intelligent Human-Machine Systems and Cybernetics, 190-195.