Implementation and Analysis of Autonomic Management
SaaS ER System

Nadir K. Salih*1, Tianyi Zang*2
Electrical and computer engineering department, Karary University, Sudan1
School of Computer Science and Engineering, Harbin Institute of Technology, China2

*Correspondences should be addressed
nadircom2006@gmail.com1      tianyi.zang@hit.edu.cn 2

ABSTRACT: To provide a single Instance of a software service to multiple users is great thing about SaaS for software vendors, is that multiple-users can access the same instance of a software service. This means that software vendors can save a lot of the money and the resources by running just a single instance of a software service, which can then be accessed by many users. But the big challenge face the SaaS application is administrators need to be able to not only monitor overall health of the application but also the performance issues related to a specific tenant. To solve this problem we follow autonomic MAPE loop in our SaaS model which include three management levels. By keep alive models with implementations we monitor prior and posterior knowledge. And we have used PRISM as model checker to dynamic analysis QoS requirements. However to obtain good performance in execution state we have design autonomic algorithm that plan to select suitable solution for the problem. We have defined the functional a nonfunctional requirements of SaaS application by electronic health record as case study. The result is implementing new SaaS system manage without user intervention. Finally we present a conclusion and the future work.

Keywords: autonomic management, SaaS application, case base reasoning, electronic health record.

I. Introduction

Medical and health service processes needs to facilitate better convenience for resident’s medical treatment. Establish and improve the health of residents file and the promotion of the regional health information network construction in the same city or region it is quite important. To design an application of the regional health care system. We need for each resident has a card use for their information and the promotion of the regional health information network construction in the same city or region it is quite important. To design an application of the regional health care system. We need for each resident has a card use for their information. To achieve this, we have designed a system that provides electronic medical records for residents. This system is designed to be accessible to medical professionals and residents alike. The system has several layers, including a provider layer, a tenant layer, and a user layer. The provider layer is responsible for managing data for all tenants, and the tenant layer is responsible for managing data for a single tenant. The user layer is responsible for managing data for an individual user. This system has several advantages over traditional paper-based medical records. For one thing, it is more convenient for both providers and patients. Patients can access their medical records at any time from any location, and providers can access patient records easily. For another thing, it is more efficient in terms of cost. According to the World Health Organization, implementing electronic medical records can save 30% of the total cost of healthcare. Finally, it is more accurate in terms of data. Electronic medical records are less likely to have errors than paper-based medical records. We have implemented this system using Java and Oracle database. The system has been tested and validated with real-world data. The results show that the system is effective and reliable. We have also compared our system with other electronic medical record systems and found that our system has better performance and efficiency. In conclusion, we have designed and implemented an electronic medical record system that is effective, reliable, and efficient. We believe that this system will have a significant impact on the way healthcare is delivered in the future.
administrator for all patient belong to unique hospital, and in user-level we have groups of patients every group has the same services and typical characteristics for application. Computation independent model (CIM) [3] for SaaSHER: here to obtain the requirements of the SaaSEHR system we used activity diagram to represent business process for applications as appeared in figure 2.

**Fig 2 Computation Independent Model SaaSEHR**

Platform Independent Model (PIM) [4] for SaaSEHR: To implementation SaaS system we have described class, operations and associations between classes that defined the SaaSEHR before coding. In a manner that is entirely independent of any programming language or operating system see figure 3.

**Fig 3 Platform Independent Model SaaSEHR**

Platform Specific Model (PSM) for SaaSEHR: here we transform PIM to PSM by represent rule and constrain and class in programming language.

**III. Enhanced -SaaS Applications**

In this term of implementation case study we have described our enhanced in SaaSEHR example. As depicted in figure 4 bellow the architecture of SaaSEHR have two main components, autonomic management and manage SaaSEHR element.

**Fig 4 Autonomic Management Architecture of SaaSEHR**

### a. Autonomic Management

Autonomic management which it showed our contributions in this paper and it represented business process layer. We have started from monitoring service of SaaSEHR for three levels patient, hospital and provider. We can monitor internal component of application by reflection concept this for functional property. However the nonfunctional property can monitor by keep alive models with Implementations which compare between prior and posterior knowledge. It update models better capture real system behaviors and give the models evolve at run time following the changes in the environment. Our self-configuration can dynamic configure layers application of provider, hospital and patient levels by deployment process in java program. We have applied dynamic analysis to quality of the services requirements in availability of care for patient and resources management for hospitals. And performance of care services from anywhere in any time. We can dynamically diagnosis the problem and solve it or from data analysis can detect or predict failure. We used case base reasoning in our new algorithm to evolution SaaSEHR. Just we put forms for patient, hospital and provider to pass some parameters it look like new case for any service or problem the system automatically will replay. By our algorithm in CBR we realized plan state. Finally is the execution of the best solution from plan state. We explain autonomic MAPE loop as follow:

#### 1. Monitoring Services

Monitor the overall system which shall include the monitoring of the requests from users, sensing the availability of the services, addition of new services, removal of services, etc. KAMI: keep alive models with implementations it is model adaptation in run time by learning from monitored data. It models at run time for early discovery/prediction of violations of assumptions made at development time implementation adaptation. We monitor the actual behavior a statistical (Bayesian) approach estimates the updated DTMC matrix (posterior) given run time traces and prior transitions. We defined down to the following updating rule from DTMC or CTMC to denoted prior transition \( \left( m_{s,s'} \right)_{s,s'} \) for monitoring from state \( i \) to state \( j \). let \( d \) is the number of running instance of system. sinti initial state for every \( h = 1,...,d \) the \( h \)th Instance execute the transition from state \( i \) to state \( j \) \( N_{i_j} \) times. There is transition matrix \( M \) is random of updating each \( m_{s,s'} \) using runtime data represented to update prior distribution of \( M \) depend on \( \left( m_{s,s'} \right)_{s,s'} \) by computing posterior conditional probability of \( M \) given runtime data:

\[
p(m(s)_{i,j} | N_{i_j}^{(h)}, m(s)_{i,j}) \text{, \( i,j \in s, 0 \leq h \leq d \)} \quad \ldots \ldots \ldots \ldots \quad (1)
\]

For update \( M \) need likelihood function and prior distribution of \( M \), likelihood function of collected data is:

\[
p(x^{(1)}_1, \ldots, x^{(d)} = S_{nd}) = \prod_{i=1}^{d} \prod_{j=1}^{k} m_{i,j}^{N_{i_j}^{(h)}} \quad (2)
\]
\[ C_N^{i+1} = C_N^i + N_i \] ...........................(3)
\[ m_i^{N_i} = m_i^0 + \sum_{h=1}^{d} N_i^{(h)} \] ...........................(4)
Where \( N_i = \sum_{h=1}^{d} \sum_{j=1}^{r} N_{i,j}^{(h)} \)

For that posterior means of \( m_{i,j}^{N_i} \):
\[ m_{i,j}^{N_i} = \frac{C_i^0}{C_i^{(i)}} \times m_{i,j}^0 + \frac{N_i}{C_i^0} \sum_{h=1}^{d} N_{i,j}^{(h)} \] ...........................(5)
\[ \frac{N_i}{C_i^0} \times \frac{\sum_{h=1}^{d} N_{i,j}^{(h)}}{N_i} \] : it represented perior knowledge
\[ \frac{m_i^0}{C_i^0} \] : it represent posteriori knowledge
\[ \frac{m_{i,j}^0}{C_i^0} \] : It called smoothing parameters, they quantify confidence in priori knowledge

From our case study SaaSEHR we have defined availability requirement by DTMC in figure 5. And we determined the prior knowledge by this requirement:
R1: Probability of meet doctor failure from patient web site \( p_1 = 0.05 \)
R2: Probability of select suitable doctor failure from hospital web site \( p_2 = 0.04 \)
R3: Probability of accounting process failure from provider web site \( p_3 = 0.03 \).

Fig 5 Availability Requirement by DTMC

We represented prior knowledge for SaaSEHR performance by CTMC in figure 6.

Fig 6 Performance Requirement by CTMC

We defined some services in request of a time to indicator for performance requirement as follow:
R4: probability of meet doctor in urgent event not more than 1 second \( p_4 = 0.95 \)
R5: probability of doctor decision in urgent event not more than 10 seconds \( p_5 = 0.90 \)
R6: probability of reporting for hospital event not more than 5 second \( p_6 = 0.98 \)

All this representation to monitoring availability and performance requirements services of SaaSEHR case study in three levels of our model. We depend on prior and posterior knowledge which it defined the change can be occur. From the next stage we analyzed this knowledge to obtain the validation of services.

2. Analysis Validity of Services

In analysis state we have represented availability service requirement in PCTL and performance in CSL. At first we defined availability requirement as in following description:
\[ P_{0.05} = [ \delta s = 17 ] \] The probability of eventually reaching state 17 meet doctor failure from patient web site which corresponds to successful completion of session is greater or equal 0.05
\[ P_{0.04} = [ \delta s = 18 ] \] The probability of eventually reaching state 18 select suitable doctor failures from hospital web site which corresponds to successful completion of session is greater or equal 0.04
\[ P_{0.03} = [ \delta s = 19 ] \] The probability of eventually reaching state 19 accounting process failure from provider web site which corresponds to successful completion of session is greater or equal 0.03.

Dynamic analysis output for availability indicator by probability failure of services as depict in figure 7 we found:
R1: requirement of meet doctor service failure if equal to 0.0150 is less than prior \( R_1 = 0.05 \) this mean is valid value for inputs.
R2: requirement of select suitable doctor service failure is equal to 0.006 is less than prior \( R_2 = 0.04 \) this mean valid value of input.
R3: requirement of accounting process service failure is equal to 0.035 is more than prior \( R_3 = 0.030 \) this mean invalid value of input it can prediction of violation for requirement.

Fig 7 Analysis of Availability Requirements

Performance requirements have been mapped into CSL representation as follow:
\[ P_{0.95} = [ \delta s_{1} = 11 ] \] The probability of meet doctor in urgent event eventually reach state 11 within 1 second is not more than or equal 0.95.
\[ P_{0.90} = [ \delta s_{10} = 9 ] \] The probability of doctor decision in urgent event eventually reach state 9 within 1 second is not more than or equal 0.90.
\[ P_{0.98} = [ \delta s_{5} = 10 ] \] The probability of reporting for
hospital event eventually reach state 10 within 5 second is not more than or equal 0.98. R4: requirement of patient urgent event in 1 second probability is 0.90 less than prior requirement is 0.95 this mean valid value for inputs. R5: requirement of doctor decision in 10 second probability is 0.5 is less than prior requirement is 0.90 this mean valid value for inputs. R6: requirement of provider reporting in 5 second probability is 0.15 is less than 0.98 as prior requirement is mean valid value of inputs. Here we observed no violated in performance requirement all services not exceeded the range of time put as requirement at first. The output dynamic analysis by PRISM for services performance requirement in three levels of our model as depict in figure 8:

3. Plan to Suitable Solution
Our CBR algorithm as depict in listing 1 it has case base contain n cases and case, case of new problem (Cnp), and similarity measure (SM) as input. The outputs of algorithm are nearest neighbor (Nn), and select solution for case Cnp.
We started by initial retrieval: starting with some attributes for case and compute similarity: that weight sum of attribute retrieval case is the same. Adaptation rule: rule obtains from engineering session with expert appraises. By additional feature cause the different between subject and retrieved cases. That will adjust to better reflect property value for final selection. The aggregate selection cases: combine to produce estimate of the value of subject, result is justification reliable value.

Listing 1 Case base reasoning algorithm:

- **Input:** CB containing n cases, Cnp: case new problem.
- **SM:** similarity measure.
- **Output:** Nn nearest neighbor, select solution for case Cnp.

For each \( C_j \in CB \)

\[ \text{simi} = \text{find similarity}(C_{np}, SM) \]

\[ w = \text{all simi cases}. \]

\[ \text{retrieve } w. \]

End for

\[ N_n = \text{find nearest solution} \]

Revision(\( N_n \))

Retention

\[ CB = CB + N_n \]

End.

By our new algorithm we have realized the autonomic behavior by 4R at first retrieve the similar cases for our problem. In the second step it reuses the cases to propose the solution. The third step confirm the solution by revise the similarity cases. Finally step will be retaining the solution in case base because it can helpful to solve other problem in future.
The SaaSEHR will help the patients to select the suitable hospital by use our self-management algorithm to realize the self-optimization for patient-level. For example we have some attributes for some cases that can let the system to choose optimal hospital according to the patient requirements if we take the costing is require.
From algorithm we have self-optimization service in user level by gave the customer suitable cost to his case that enters to system. After we selected similar cases by algorithm.
Hospital level solution by different similar functions: we defined optimal solution by accuracy and RMSE. That realized the best two similar functions are selected case 1 is suitable costing solution to new case see data in figure 9.

![Fig 9 Accuracy and RMSE for Similar Function in User Level](image_url)
Our algorithm gave similar diagnosis to self-optimization diagnosis services for doctor. We let the SaaSEHR selected optimal diagnosis by used accuracy and RMSE as depicted in figure 10.

In our example we found the best two similar function are typical gave one optimal solution is case 5, solution problem for new case is Stomach trouble. We have made self-healing by our algorithm in provider or administrator level by select optimal solution of problem. For example we have 5 cases of failure denoted the reason of failure and solutions as mentioned have new case of failure require the solution.

*Setting up cron is an important step in the installation of the website and assists in the maintenance of the site's assets for search results.

Administrator level solutions by different similar functions: Our algorithm obtained similar cases for self-healing problem by different similar function.

We selected the optimal solution for healing problem by accuracy and RMSE as depict in figure 11, figure 12. In our example we made self-healing by found the optimal solution for new problem. That realized in tow similar functions to obtained case3 is the nearest solution. Exactly we realized MAPE loop of autonomic management in our SaaS model.

IV Manage SaaSEHR Elements

We have used browser in presentation layer to show three levels graphical interfaces for provider, hospital and patient for login and services inside SaaSEHR. And it included all features we added in autonomic management. We deployed our SaaSEHR application by the environment of Java is called eclipse9. To enables the creation of dynamic, platform-independent method for building Web-based we used JSP in application layer. In addition we have designed our database by concept of multi-tenancy that share schema for common data and isolated it variable.

V. Related work

For study the recent research in SaaS application we have been classified it in fourth directions

• The first direction is meta-model and modeling to evolution SaaS application. In [5] they Defined criteria for designing the process model and realized Commonality and variability modeling to maximize the reusability. Researchers in [6] they analyzed tenancy history metadata from Graphic User Interface (GUI), workflow, service, and data layer for adjusting template objects dynamically. In [7] they provided on-demand service-oriented model driven architecture to develop an enterprise mashup prototype as a practical case study. Authors in [8] they regarded PIM can be used to generate different PSMs using transformation tools to minimize the time, cost and efforts in developing cloud SaaS and enhance the return on investment. They identified technical issues and proposed their effective solution spaces in [9]. In [10] they proposed a QoS model and MCDM (Multi Criteria Decision Making) system for SaaS ERP. They empirically examined main drivers and inhibiting factors of SaaS-adoption for different application types in [11]. In [12] they studied forecasts effects expected when the SaaS model will be fully applied to the library network. And presented functional requirements and an operation model of SaaS-based library management systems. In [13] they extensible business component model named xBC is proposed for describing both the structural and behavioral properties of generic SaaS applications. For minimized the amount of source that needs to be reexamined by a transformation when the source is changed.

• The second direction is how to configuration in SaaS application model. In [14] they defined Composite configurable SaaS application packages. With variability descriptors they derived fixed and tenant-specific parts of a solution. Effectively and efficiently support configurability in SaaS software and proposed SaaS architecture in [15] by used data in xml format. In [16]
they provided information on the nature of configurability in SaaS software. To described several novel methods used in the metadata.

- The third direction is Resources management in SaaS application. In [17] they defined hybrid approach to solve placement of tenants. Selecting the optimal execution plan based on the distance of the tenant resources consumption vector. They Calculations resource requirements in [18] for multi-tenants with applied constraints in a shared application instance. To obtained Optimal placement of tenants and instances with maximum cost savings. In [19] they established formal measurements for under and over provisioning of virtualized resources in cloud infrastructures and proposed a resource allocation model to deploy SaaS applications. In [20] they built up SaaS stack to lead worker agents autonomically check the feasibility and determine the amount of resources.

- The forth directions is Database management in SaaS application. In [21] they presented a schema inheritance concept tailored to multi-tenancy. Schema considerably decreases main memory consumption and lookup times of the data. They introduced memory database as the implementation mechanism of distinguishing table instances in [22], to resolve the challenges between the business components tenant-unrelated and the global access of the data. In [23] they proposed a novel method for transferring the existing business software to integrated software that can be used in the SaaS. By used a master table and master code to implement the integrated system.

From these recent researches we have observed no work in autonomic management SaaS application. Although autonomic management it has applied in other type of applications and gave good result of increase the performance and enhance management. For that we implement new SaaS system manages autonomically.

VI. Conclusion and future work

Healthcare it is very hot research in recent development software engineering and service computing. For that we take it as case study in this research. We have used the model development driven to describe this application from specification requirements. Also we defined the classes and their association and relationship between them. All of that to validation our contributions in this paper and obtain some enhancements for SaaS system. That is it has realized in autonomic MAPE loop. Firstly we monitor prior and posterior knowledge to catch the variation in requirements. Secondly by our knowledge representation of discrete and continuous markov chain we predict and detect the violation of requirement in analysis state. Thirdly in plan state we have applied our new algorithm to select suitable solution in our SaaS system. The benefits of increase the SaaS performance obtains in execution state which execute the suitable select from plan state.

References

[15] Nitu MCSI, MIETE, Configurability in SaaS
(Software as a Service) Applications. ACM, 2009.


