Evaluating Data Integrity in the Cloud using the UPPAAL

Sachi ishida[1] and oshiyu i Shin awa[2]

[1]*Fujitsu FSAS Inc., 13-2 Nakamaruko, Nakahara-ku, Kawasaki, Kanagawa, Japan*

[2]*Graduate School of Science and Technology, Ryukoku University, 1-5 Seta Oe-cho Yokotani, Otsu, Shiga, Japan*

eywords Cloud Co puting UPPAAL Transaction Processing Data Integrity

A stract There are several considerations when i ple enting a transaction processing syste in cloud environ ents li e oogle App Engine AE One of the ost critical ones is the data integrity since the cloud provides us with li ited capa ility for it Therefore we need to evaluate the applications and the cloud platfor carefully fro the data integrity viewpoint This paper presents a odel ased data integrity evaluation ethod using the UPPAAL odel chec er In order to a e the odel reusa le we uilt it as a set of application independent functional odules On the other hand the application uni ue functionalities are to e included in the odel as UPPAAL functions written y the C-li e UPPAAL language The data integrity evaluation is perfor ed in two different ways One is a si ulation ased ethod in which the odel is e ecuted y the UPPAAL si ulator to o tain the resultant varia le values The other is a veri cation ased ethod in which the given integrity constraints are e a ined y the UPPAAL veri er using full state space search of the odel

1 INTRODUCTION

Data integrity is one of the ost critical concern for distri uted and concurrent syste s especially for those in cloud environ ents e g oogle App Engine AE Sanderson 2009 A a on e Services A S van liet and Paganelli 2011 or I lue i I 2015 One of the typical syste s is data ase transaction processing and the data integrity eco es crucial issue to a e such syste s ro ust ishida and Shin awa 2014

Therefore the evaluation of the data integrity fro oth application and platfor viewpoints for transaction processing in the cloud see s i portant to the spread of cloud co puting owever there are several dif culties in evaluating and validating this data integrity for transaction processing The a ove dif culties are ainly caused y the different principle of the data integrity fro traditional transaction processing which is adopted y the cloud

This new and different principle is referred to as ASE standing for **B***asically* **A***vailable* **S***oft state* and **E***ventually consistent* Pritchett 2008 The asic differences etween the ASE and the traditional principle ACID [1] ray and euter 1993 oth of which are a set of properties to e satis ed in order to guarantee the data integrity in transaction processing are

1 hile the ACID restricts the concurrent data ase accesses within a critical section the ASE allows ar itrary concurrent data ase accesses fro any transaction y **B***asically* **A***vailable* property

2 hile the ACID postulates the transparent replication of the data ases the ASE tolerates the non-transparent replication y **S***oft state* property

3 hile the ACID ai s at the data integrity at every instant the ASE tries for achieving the data integrity within so e duration y **E***ventually consistent*

According to the a ove differences etween these two principles na ely ASE and ACID we need a different approach to evaluating the data integrity in the cloud

Since this evaluation ust e perfor ed efore syste i ple entation we need a precise odel that re ects the cloud platfor echanis i ple enting the ASE principle along with the detailed application logic that deter ines the data values The reason why is that the data integrity of transaction processing is affected y oth of the

owever ost odeling tools are speciali ed to a speci c aspect of a syste e g software speci cation languages li e Spivey 2008 D

[1] ACID is standing for **A***tomicity* **C***onsistency* **I***solation* and **D***urability.*

Nishida, S. and Shinkawa, Y.
Evaluating Data Integrity in the Cloud using the UPPAAL.
DOI: 10.5220/0006005103040309
In *Proceedings of the 11th International Joint Conference on Software Technologies (ICSOFT 2016) - Volume 1: ICSOFT-EA*, pages 304-309
ISBN: 978-989-758-194-6

Fit gerald et al 2004 and so on which are speciali ed to the functional aspect syste odeling tools li e nite state achines Petri ets [2] eisig 1985 SDL Thiel 2001 and so on which are speciali ed to the ehavioral aspect and architecture oriented odeling tools li e U L class diagra s loc diagra s and so on which are speciali ed to the structural aspect

On the other hand it is desira le to e press ultiple aspects of a syste si ultaneously in a single odel for accurate evaluation of the data integrity For this purpose we use the UPPAAL odel chec er David et al 2015 as a odeling and evaluation tool since it can e press the ehavior of a syste as a set of ti ed auto ata connected through co unication channels along with the functional and data structure speci cations using a C-li e language provided y the tool

The rest of the paper is organi ed as follows In section 2 we introduce the asic concepts of the data integrity in the cloud along with the transaction ehavior following the ASE principle Section 3 shows how transaction processing in the cloud is odelled using the UPPAAL Section 4 discusses an evaluation and validation ethod for the data integrity using the UPPAAL

2 TRANSACTION PROCESSING IN THE CLOUD

Data integrity in transaction processing has een hitherto relying on the ACID principle that is guaranteed y a transaction processing onitor TP under which they are running One of the ac grounds of the ACID is that the seriali ed e ecution of transactions always aintains the data integrity Therefore at the i ple entation level the TP isolates and seriali es the critical sections of each transaction y *locking* echanis In addition the ACID i plicitly presu es the transparent replication or synchronous replication of data ases to reali e C Consistency property of it

This approach could cause the reduction of data ase availa ility along with the perfor ance degradation of transaction processing In cloud co puting the ACID principle eco es a urden too uch to guarantee the high availa ility scala ility and sta le perfor ance of a syste Therefore ore light-weight echanis to aintain the data integrity is desired in the cloud

[2]E cept for higher order Petri ets li e Coloured Petri ets ensen and ristensen 2009

The ASE principle is a newly introduced principle to co pro ise the con icting re uire ents that is availa ility and integrity in the cloud In order to i prove the availa ility the ASE principle does not seriali e the critical sections of each transaction and allows the non-transparent or asynchronous replication For aintaining the data integrity in such an environ ent a TP following the ASE principle provides us with version infor ation instead of a loc ing echanis in order to deter ine whether the referred data are valid If so e of the referred data are invalid the relevant transaction a orts the data ase updates This echanis is nown as opti istic loc ing

efore discussing the data integrity of ASE transactions [3] we need to de ne the concept of data integrity rigorously in order to evaluate it effectively The ter integrity or data integrity is used differently in various conte ts For e a ple it focuses on the relationships etween directories and le allocation infor ation e g i-node in the case of U I at the operating syste level while it eans the referential integrity that re uires the e istence of speci c ey values at the D S Data ase anage- ent Syste level

On the other hand at the application level there are no co only recogni ed de nitions since it depends on the se antics of the data rather than their structure Therefore it see s ore dif cult to e press the data integrity at this level than the for er two levels In order to deter ine whether an application can e perfor ed in the cloud in the for of a transaction we have to evaluate the data integrity at the application level in this circu stance

Conse uently we rst need to de ne rigorously the concept of data integrity at the application level using a uni ed notation The data integrity at the application level can e de ned as a set of constraints or rules on data ase occurrences One of the ways to e press these constraints is to use predicate logic for ulae Shin awa 2012 In order to co pose these logic for ulae we rst have to de ne the language $\mathcal{L}$ and the structure $\mathcal{S}$ to provide the synta and se antics of the for ulae

The language $\mathcal{L}$ stipulates the usage of sy ols regarding constants varia les functions predicates and logical operators In the data integrity evaluation the $\mathcal{L}$ deals with data ase related atters Therefore each sy ol for a varia le or constant represents an entity or its value in the data ases As for functions and predicates there are two inds of the that is data ase oriented and application oriented The for-

[3]Transactions to e run under the control of a TP i ple enting the ASE principle

er ones are the functions or predicates de ned in a data ase anipulation language li e S L On the other hand the latter ones are those used in a speci c application do ain e g production control product anage ent or custo er anage ent applications

Therefore we are to prepare the $\mathcal{L}$ as co posed of two parts na ely the application independent part and application dependent part hile the for er part can e reused a ong the different application do- ains the latter need to e uilt every ti e a new application is dealt with On the other hand the structure $\mathcal{S}$ consists of the do ain of discourse $\mathcal{D}$ and the interpretation I All the o ects that are referred to fro the functions and predicates or assigned to vari- a les and constants ust e the ele ents of the a ove $\mathcal{D}$ In our case this $\mathcal{D}$ includes

1 all the data ase instances D $_i$

2 all the data ase records $r_j^{(i)}$ in each D $_i$ and

3 all the attri ute values $a_k^{(ij)}$ in each $r_j^{(i)}$

The interpretation I aps each sy ol in the $\mathcal{L}$ to an actual entity de ned over the $\mathcal{D}$ So e of functions and predicates are prede ned in a data ase anipulation language e g S L Other sy ols in the $\mathcal{L}$ are de ned during the odeling process discussed in the succeeding sections

Using the a ove language $\mathcal{L}$ and the structure $\mathcal{S}$ each constraint to e press an integrity rule is represented y a standardi ed logic for ula PC F Prene Con unctive or al For

$$Q_1 \cdots Q_n (\bigvee_j \bigwedge_i P_{ij} (t_1^{(ij)} \cdots t_{m_{ij}}^{(ij)}))$$

where Q_i is a varia le with the uanti er $\forall$ e g $\forall x_i$ P_{ij} is a predicate and $t_k^{(ij)}$ is a ter co posed of vari- a les constants and functions Schoening 2008

There are several inds of constraints regarding data integrity e g restrictions on data values e - istence of a record with so e speci c ey or constraints on the values derived fro a set of records owever any inds of those constraints can e e - pressed y the a ove predicate logic for ulae in the for of PC F

3 MODELING THE TRANSACTION PROCESSING WITH THE BASE PRINCIPLE

Once the rules or constraints for data integrity are e - pressed in the for of predicate logic for ulae the ne t step is to odel the transaction processing with the ASE principle which updates the data ases in the cloud For this odeling we use the UPPAAL odel chec er or the UPPAAL in short The UP- PAAL e presses a syste as a set of nite ti ed au- to ata with varia les along with the functions that anipulate the

Each ti ed auto aton consists of states *locations* in ter s of the UPPAAL and arcs *edges* in ter s of the UPPAAL that represent the state transitions oolean e pressions with cloc type varia les can e used as ti e constraints which are associated with any a ove stated *location* or *edge* These ti ed au- to ata are de ned as para eteri a le te plates and ust e instantiated y the syste de nition

In order to a e the odels reusa le these te - plates should e appropriately odulari ed In our approach the ehavior of the transaction processing in the cloud is categori ed into ve types na ely Initiali ation Scheduling Thread Data ase and eplication Each odule wor s as follows

1 The Initiali ation odule sets up the data ases to e used during the si ulation The data ases are e pressed as three-di ensional integer arrays The rst di ension represents the replication nu er the second represents the record or row nu er and the third represents the attri utes in the data ase sche a

2 The Scheduling odule sends a transaction to one of the Thread instances to process it A transaction is e pressed in the for of integer array each ele ent of which represents an argu ent or para eter to the transaction These integer arrays co pose a two di ensional transaction list [4]

3 The Thread odule perfor s the functionality of each transaction The functionality is deter- ined y the transaction type and the speci ed argu ents in the transaction list The data ase update re uests fro a transaction are routed to the Data ase odule through a UPPAAL channel

4 The Data ase odule is to e instantiated as any as data ase replications Each instance reads and updates a speci c replication of a data ase e pressed in the for of an integer array

5 The eplication odule tries to eep the replicated data ases identical in an asynchronous way

[4]Since the UPPAAL allows only ed si e for arrays and each transaction type could re uire the different nu - er of argu ents an individual two-di ensional array is de ned for each transaction type independently

i ple enting the **S***oft state* property This od-ule is instantiated only once and deals with all the data ases and their replications

In addition to the a ove odules we have to prepare several functions to a e the odel e ecuta le and veri a le These functions are written y a C-li e UPPAAL uni ue language hile the odel structure is co on a ong application do ain these functions are application uni ue and ust e uilt for each application do ain

Figure 1 through Figure 5 show an e a ple of the a ove UPPAAL odules As stated a ove the structure of their ve odules can e co only used a ong different application do ains including function na es and channels associated with edges and locations in the odel owever the i ple entation of these functions and other supple ental functions are differently uilt a ong different application do-ains

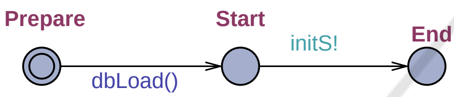

Figure 1 Initiali ation odule

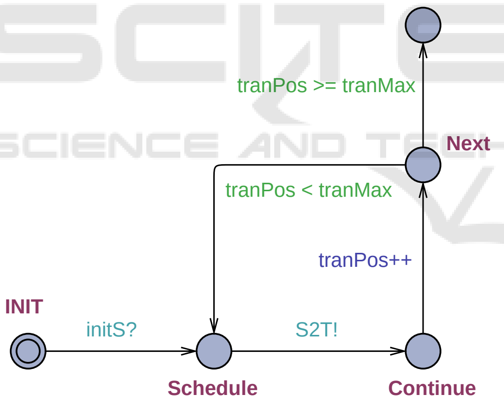

Figure 2 Scheduler odule

For e a ple the function d Load in Figure 1 represents a function that initiali e all the data ases in the syste and the na e is co on for all applications owever its i ple entation usually different a ong the depending on the structure and usage of the data ases Figure 6 shows a sa ple i ple entation of the d Load function for a si pli ed li-rary application

hen e ecuting the odel these odules are instantiated through the syste de nition as shown in Figure 7 In this e a ple three concurrent threads and three data ase replications are assu ed

These odules operate as follows

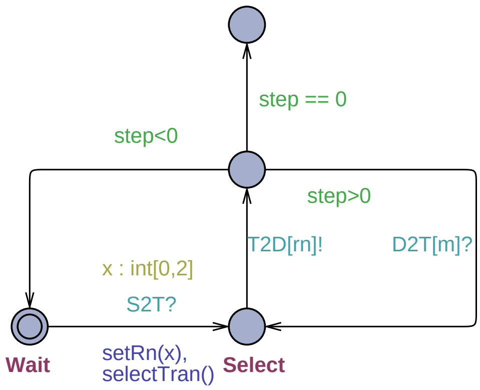

Figure 3 Thread odule

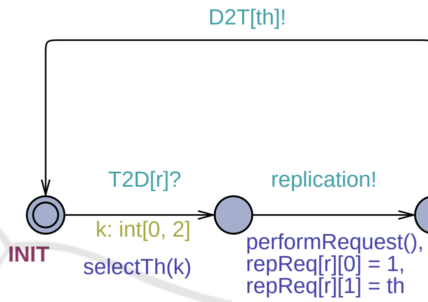

Figure 4 Data ase odule

1 Firstly the d Load function of the Initiali ation odule is invo ed to prepare all the data ases At this ti e only the associated edge is eligi le for transition since other odules are waiting for signals through the UPPAAL channels

2 After the co pletion of the d Load function the Scheduler odule is activated through the initS channel

3 The scheduler odule sends a signal to the Thread odule through the channel S2T

4 The Thread odule selects a transaction fro the prede ned transaction list y the selectTran function and sends a signal to the Data ase odule through the channel T2D channel where the is a replication nu er

5 The Data ase odule accesses and updates the data ases

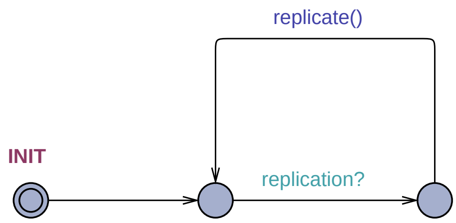

Figure 5 eplication odule

```
void d Load {
  for i int 0 2 {
    for int 0 14 {
      int search oo i oo ey
      if > 0 {
        for int 0 3 oo i oo
            i
      }
    }
  }

  for i int 0 2 {
    for int 0 8 {
      int searchAccount i account ey
      if > 0 {
        for int 0 3 account i
            account i
      }
    }
  }

  for i int 0 2 {
    for int 0 4 {
      int searchLoan i loan ey 0 loan ey
           1
      if > 0 {
        for int 0 3 loan i loan
            i
      }
    }
  }
}
```

Figure 6 d Load Function

```
// Place template instantiations here.
In Initiator
Sc Scheduler
T1 Thread 0
T2 Thread 1
T3 Thread 2
D 1 D 0
D 2 D 1
D 3 D 2
 EP eplication
// List one or more processes to be composed into
     a system.
syste In Sc T1 T2 T3 D 1 D 2 D 3 EP
```

Figure 7 Syste De nition for odule Innstantiation

6 The step 2 to 5 are repeated until all the prede ned transactions are processed

The version control and co it a ort processes are e edded in the Data ase odule as functions

4 DATA INTEGRITY EVALUATION USING THE UPPAAL

The UPPAAL odel chec er provides us with three a or functionalities The rst is a graphical odel editor with progra ing capa ility that we have used in the previous section The second is a odel si - ulator that e ecutes the odel we uild to show an instance of its ehavior The third is a odel veri- er that e a ines all the possi le ehavior whether the odel satis es the given properties written in the for of CTL Co putational Logic Tree for ulae

Therefore two alternative ways are availa le to evaluate the data integrity of transaction processing The rst is to e ecute the odel to o tain the val- ues of the varia les for the data ase records at each state transition As discussed in the previous sec- tion the data integrity is e pressed as a set of pred- icate logic for ulae in the for of PC F In the UP- PAAL odel these logic for ulae refer to the vari- a les associated with the data ase records and at- tri utes Therefore we can deter ine whether the data integrity is aintained in the transaction process- ing y e a ining the a ove varia les using a function i ple enting each constraint logic for ula Since this ethod can evaluate only one instance of the sys- te ehavior selected y the si ulation we have to perfor the si ulation for every possi le ehavior owever this possi le ehavior could e uncount- a le Therefore this ethod would e sa pling ased evaluation

On the other hand the UPPAAL veri er provides us with a capa ility of full state space search against a set of CTL for ulae In order to evaluate the data integrity in this way we have to transfor a set of predicate logic for ulae into a set of CTL for ulae Unli e the predicate logic for ulae CTL for ulae can include the path operator A and E which deal with state transition paths of a syste and te po- ral operator $\Box$ and $\Diamond$ which de ne the valida- tion points of the for ulae In addition there are no uanti ers $\forall$ and $\exists$ in CTL Therefore sev- eral considerations should e ta en into account in the a ove transfor ation fro predicate logic for ulae into CTL for ulae These considerations include

1 If a property P ust always holds in a predicate logic for ulae the CTL for ula is $\mathrm{A}\Box P$

2 If a property P always i plies a property Q then the CTL for ula is $\mathrm{A}\Box(P \rightarrow Q)$

3 If a property P eventually i plies a property Q the CTL for ula is $\mathrm{A}\Box(P \rightarrow \Diamond Q)$

4 If a property P ust hold at speci c point we introduce a oolean varia le to e press the point and set it *true* at the point in the odel In this case we need to odify the odel

5 If the original predicate logic for ula includes the uanti ers $\forall$ and $\exists$ we introduce a oolean function into the odel to e a ine whether all of or so e of the varia les in the odel satisfy the for ula A odel odi cation is re uired in this case again

After the a ove transfor ation is co pleted we can evaluate the data integrity y running the veri er that the UPPAAL provides

This CTL ased evaluation see s si pler than the si ulator ased one however it perfor s full state space search and consu es huge co puting resources As a result it ta es long ti e to o tain the result In such cases we need to reduce the odel y decreasing the nu er of varia les or values to e assigned

5 CONCLUSIONS

In cloud environ ents the ehavior of transaction processing is considera ly different fro the traditional ones One of the a or reasons is that the cloud introduces a new principle for the data integrity called ASE instead of the traditional ACID In order to a e the transaction processing sta le in the cloud we need to reveal the ehavior of it clearly and evaluate the data integrity rigorously

This paper proposed a odel ased data integrity evaluation using the UPPAAL odel chec er In order to a e the odel easily understanda le and reusa le we co posed it using ve functional odules na ely Initiali ation Scheduling Thread Data ase and eplication following the ASE principle hile the odel structure can e reused a ong different application do ains we need to uild application uni ue functions for the odel

The UPPAAL provides us with two different ways to evaluate the data integrity One is a si ulation- ased evaluation which e a ines only one instance of the ehavior of transaction processing The other is a veri er- ased evaluation which e a ines full state space search to deter ine whether the given constraints are satis ed hile the latter way can evaluate the integrity ore precisely we need to transfor the original predicate logic for ulae into the CTL for ulae In addition it consu es huge co puting resources for full state space search and ta es long ti e to o tain the evaluation results

ACKNOWLEDGEMENTS

This wor was supported y SPS A E I rant u er 25330094

REFERENCES

David A Larsen and Legay A 2015 UPPAAL S C tutorial In *International Journal on Software Tools for Technology Transfer Volume 17 Issue 4* pages 397 415 Springer

Fit gerald Larsen P u her ee P Plat and erhoef 2004 *Validated Designs for Object-oriented Systems* Springer

ray and euter A 1993 *Transaction Processing: Concepts and Techniques* organ auf ann

I 2015 *Bluemix* http www i co cloud-co puting lue i

ensen and ristensen L 2009 *Coloured Petri Nets: Modeling and Validation of Concurrent Systems* Springer- erlag

ishida S and Shin awa 2014 Data Integrity in Cloud Transactions In *Proc. 4th International Conference on Cloud Computing and Services Science* pages 457 462

Pritchett D 2008 ASE An ACID alternative In *ACM QUEUE Volume 6 Issue 3* pages 48 55 AC

eisig 1985 *Petri Nets: An Introduction* Springer

Sanderson D 2009 *Programming Google App Engine* Oreilly Associates Inc

Schoening U 2008 *Logic for Computer Scientists (Modern Birkhaeuser Classics)* ir haeuser oston

Shin awa 2012 CP ased Data Integrity Evaluation for Cloud Transactions In *Proc. 6rd International Conference on Software Paradigm Trends* pages 267 272

Spivey 2008 *Understanding Z: A Specification Language and its Formal Semantics* Ca ridge University Press

Thiel A 2001 *Systems Engineering with SDL: Developing Performance-Critical Communication Systems* iley

van liet and Paganelli F 2011 *Programming Amazon EC2* Oreilly Associates Inc