A Test Approach for Process Service by Setting Proxy Database Server of Generated Database States

Jo Yong Hui, Choe Song Hyok

Information and Science Faculty, **Kim Il Sung** University

The great leader Comrade **Kim Jong Il** said as follows.

"Scientists and technicians, basing themselves firmly on reality, must take the problems arising in practical socialist construction as the subjects of their scientific research, and solve the scientific and technological problems appearing in the application of the achievements of their work to production, in a responsible manner."

Today, large scale distributed process has become a major problem and SOA is being applied to implement it. In SOA process service subdivides business into several jobs and implements flow process which changes with the situation [1]. Thus, to establish the methods of testing process service is a main problem to improve quality of SOA application.

States in a test approaches proposed by [2] does not mean DB and they are stored in memories of container as data which provides components.

Web service test method described in [3] is the test done by request message description. And it is not useful for process services. Process test described in [1] can be only used for making request message and invocation of process service and cannot test business service.

We suggest auto-generating method and process service testing method to test DB states (structures and contents of database) of basic data service of process service. Then we simply set generated DB states using proxy database server, not setting to different database servers and suggest a method to reduce time of DB accesses with DB connection of all basic data services to proxy database server.

1. The Generation of Database States for Process Service Test

1) Auto-generation of database states using types and constraints of fields

Expr attribute of field tag represents constraints of a field of table.

Here we write it as regular expression.

Expr = ">=1990/1/1 & <=2015/1/1 &" | "김*" | "김* & 박*"

*: means all input.

김*: means all input started with "김".

김* & 박*: all input started with "김" or "박"

>=1990/1/1 & <=2015/1/1: all date between 1990.9.1 to 2015.1.1

Using above regular expressions we auto-generate following records of table.

① Set values which satisfy its constraint.

② Set values on the boundaries of constraint.

According to the types of the field, analyze its constraints and generate input scope. If its type is integer or real number, input scope is a number interval. If constraint is null, use byte boundary or 2 bytes boundary.

③ Set previous value of boundary of constraint to the field.

2) Test of process services

Definition 1. Process Service Graph Gp

Gp = (Start, Sp, Tp, End)

P: Process Service

Sp: Set of services invoked by P

Start: the start node of P and is contained in Sp

End: the end node of P and is contained in Sp

Tp: Set of transactions of P, one transaction start between two services

Definition 2. Process Service Testcase TCp

TCp = (Gp, Dp, TargetPathp)

Gp: Graph of P

Dp: Set of DB states which is used in connection of servicesin Sp of Gp

Dp = { x | x = DB State of y, y∈Sp of Gp }

TargetPathp: path to be compared after test for P.

In order to test using test case above, we generate all Tp covering all running path of P by TargetPathp of TCp.

Definition 3. Database State Set DBSp

DBSp: All possible sub sets of DBS1 ×DBS2 × DBS3 ×…× DBSn

DBSi: { ri1, ri2, …, riki }

ri1, ri2, …, riki: Records of table which Si connects.

ki: Number of records of table which Si connects.

P: Process Service

n: count of DB services in Sp.

Si: ith DB-Service of Sp

Algorithm Stochastic generation of TCp

①Set CurrentNode as Start. Set DBp as DBSp, Set index as 1.

②If CurrentNode is End(sn), go to Step ⑦, else goto step ③.

③For all edges e in Tp joined to CurrentNode, it repeats from step ④ to step ⑥.

④If Se, end node of e is DB service go to step ⑤ else go to step ③.

⑤Calculates m, count of edges in Tp of Gp started from Se.

⑥Let k is count of members of Tp.

Select subset from DBSp as probability of m/ k. Let selected subset is selectedDBSp. Here Set DBp as intersection of DBSp and selectedDBSp. Increase index and Set CurrentNode as Sindex. Goto ②.

⑦If Coutn of DBp is bigger than GenCount, goto ①, else goto next step.

⑧For all members Dp in DBp, execute service process with Dp and then set

TargetPathp by real path throughout process.

⑨ Add Tcp = (Gp, Dp, TargetPathp) to TCpList.

Input: Gp, DBSp, n

Output: TCpList (list of testcases)

2. Experiments and Results

In order to show advantages of this approach, we test broadcasting business process. Among several business processes, we consider one, which transmits media from edit to broadcast. The business process for this is represented as following Fig.

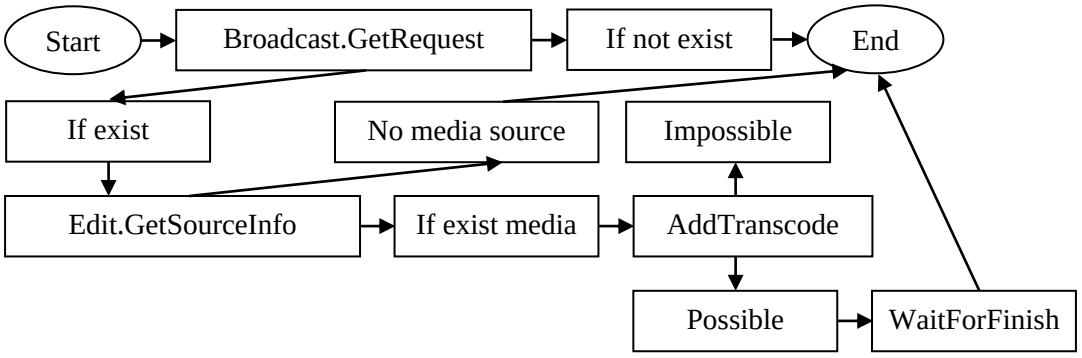

Figure. Business process from edit to transmit

Here "Broadcast.GetRequest" S1 ,"Edit.GetSourceInfo" S2 and "AddTranscode" S3 are DB services. "WaitForFinish" S4 is NonDBService which is called after transmit.

S3 is data write service, so we consider S1 and S2. To generate records of broadcasting table, we set condition expression of every field.

We select following 50 members by using algorithm.

DB1 = empty

DB2 = {(R11, R21)}

DB3 = {(R12, R26)}

DB4 = {(R16, R23)}

DB5 = {(R11, R21), (R11, R23)}

…

DB48 = {(R12, R21), (R14, R26), (R15, R27), (R16, R28)}

DB49 = {(R14, R25), (R13, R28), (R11, R26), (R12, R24), (R12, R28), (R13, R24)}

DB50 = all

The sets and testcase list of database states are represented as following.

DBp = {DB1, DB2, DB3,…, DB50}

TCpList = {(Gp, DB1, TCp1), …, (Gp, DB50, TCp50)}

Some of the paths throughout process after execution with above database are following.

Start-> S1 ->End : DB state of S1 is empty.

Start-> S1->End : DB state of S1 contains a record.

Start-> S1-> S2->End : DB state of S1 contains a record with "MediaStatus" field as 0.

Start-> S1-> S2->End : DB state of S1 contains 2 records and DB state of S2 is empty.

Start->S2->S2->S3->S4->End: DB state of S1 contains 2 records and DB state of S2 contains a record with random values.

Conclusion

We suggest a test approach for process service using generated database states. It is able to improve quality in implementing distributed system based on SOA.

Proxy database server reduces much time than other SQL servers. Using proxy database server, for 25 testcases the generation takes 11.5s and the test takes 15.5s. For MS SQL or Oracle the test takes 4.7s. In cases of using remote DB server it takes more than 6.7s

References

[1] M. Nicolai et al., SOA in Practice, O'Reilly Media Inc., 81~98, 2007.

[2] Dorothy Graham et al., Experiences of Test Automation, Pearson Education Inc., 33~48, 2010.

[3] Charitha Kankanamge et al., Web Services Testing with soapUI, Packt Publishing Ltd., 7~30, 2012.

Keywords : proxy database, process service, SOA