Equipment Collaboration Expressions in Automatic Test of Safety Critical Systems

J.H. Lv
State Key Lab of Software Development Environment
School of Computer Science
Beijing University
Beijing 100191, China

S.L. Ma
State Key Lab of Software Development Environment
School of Computer Science
Beijing University,
Beijing 100191, China

B. Sun
China Academy of Space Technology
Beijing, 410073, China

X.J. Li
State Key Lab of Software Development Environment
School of Computer Science
Beijing University
Beijing 100191, China

Abstract--**The trustworthiness of safety critical system (SCS) is very important. To assess their trustworthiness depends on data from test. In order to ensure the reliability and validity of test data, especially for such complex SCS, development of test languages is inevitable trend for automatic test of SCS. As general test language for SCS should be independent of specific equipment, in the paper types and syntax of expressions of equipment collaboration are abstracted, evaluating rules for these expressions are designed, and related properties are proved so that to support the generality of SCS test languages.**

Keywords-trustworthiness; automatic test; equipment collaboration; operational semantics; SCS (Safety Critical System)

I INTRODUCTION

Safety critical system is the system that once failure occurs, heavy losses of property, even life and environmental destruction, will be caused, whether it is trustworthy [1, 2, 3] has become widespread concerns. While evaluation and verification of trustworthiness depend on test data, namely, by means of test data whether SCS is trustworthy is assessed [4, 5, 6]. Thus how to get creditable test data is essential. In the other hand, since SCS is mostly complex, manual testing will lead inaccuracy to test; even in some cases testing cannot be carried out in manual. Thus automation of test has become an inevitable trend in SCS testing. So automatic test is an important guarantee for validation and evaluation of SCS trustworthiness.

Test of different types of SCS involves different types of test equipment, currently; the testing mode of SCS is multi-SCS testing in parallel. In this case, different types of test equipment and SCS collaborate together to complete testing. In order to support new types of test equipment and SCS to join to the testing system when necessary, Collaboration of equipment of the testing system should be dynamical and open to decouple with special equipment. Traditionally, these high order instructions are encoded and collaborated as first order data, which brings that the testing system is tightly coupled with some equipment and SCS, and then the testing system is hard to be applied on tests of different SCS or even the same SCS with different test equipment.

As equipment collaboration in SCS test are high order collaboration systems [7]. there are various test equipment and products under test, and the contents of collaborations among these entities are mostly operations, different kinds of equipment have their own operations, if these operations are taken as first order data, the automatic test system would be tightly coupled with test equipment and spacecraft, which means that only these fixed test equipment can use the automatic test system. Thus in SCS testing, a general high order representation of operations should be abstracted to describe these operations, and equipment collaboration model need to be construct to abstract collaborations among equipment to support general collaboration of equipment.

II RELATED WORKS

Existing equipment collaboration can be divided into two kinds: indirect collaboration, which mainly used to remote access to devices in order to share equipment, in this case, kinds and numbers of equipment are rare, and equipment collaborates by users who using these equipment. The other one is simple collaboration, such as BPEL [8], WSFL [9], YSWL [10] and GSEL [11]. In this mode, the equipment is packaged as services, and collaboration among equipment is implemented by compositions of services. However, because of the lack of descriptions of equipment operations or instructions, the collaboration among equipment cannot be organized dynamically and directly.

Mcmullen D [12] gave ontology equipment models, which describe equipment in ontology to unify the definition of kinds of equipment, but functions of equipment are lack of. Kawsar F [13, 14] adopts documents to encapsulate characteristics of equipment, as a unified description of equipment, but how equipment collaborates with each other is not shown. Consortium E [15, 16] supplied an interface for equipment access, defined types for equipment, if equipment change, then the interface has to be modified. Chen C [17] based on ATLAS, put forward DDL (Device Description Language) to describe equipment. Chen F [18, 19] defined a large –scale device collaborative process DCS (Large-scale lighting device collaborative system) to define process of lighting equipment collaboration.

Based on the analysis of SCS test process, the contributions of the paper is to give and prove collaboration model of equipment in SCS test to describe high order equipment collaboration process.

III TYPES OF EQUIPMENT COLLABORATION IN AUTOMATIC TEST OF SCS

Compared with the common types, such as Integer, Boolean, Char, Real, and Arrays, there are specific types for equipment collaboration in SCS test languages, such as types of test data, test equipment and SCS under test.

Test equipment consists of test devices, test system, and other test resources. Due to various types and numbers of test equipment in testing process, this equipment is hierarchically managed according to their functions so as to manage and control this equipment clearly.

The top layer is called communication middleware of testing process (MTP), which controls and manages test equipment to be accessed by testing tasks transparently. MTP is in fact a gateway of test equipment at application level. The middle layer is called device application level (DAPP), which provide unified access to heterogeneous test equipment. DAPP is in fact a gateway of test equipment at equipment level. The bottom layer is test equipment level (TE), which finds the access route to possible physical equipment. By interactions with MTP, DAPP and TE, the testing tasks can access the needed physical test equipment.

Test equipment type is defined as follows.

Dev = (DevId, DevKind, DevInfo, DevOpS, DAPPName)

Where DevId is test equipment identification; DevKind is test equipment class; DevInfo is test equipment working status; DevOpS is test equipment operation instruction set; and DAPPName is device-level gateway of test equipment.

Based on the above type definitions, variables or constants of SCS and test equipment can be described in SCS test languages. The syntax format is the same as description format of common languages.

IV EXPRESSIONS OF EQUIPMENT COLLABORATION IN SCS TEST

The following part introduces expressions of basic equipment collaboration, equipment collaboration atom and equipment collaboration.

The definition of basic equipment collaboration expression, TestP, is as follows.

TestP ::= DevRequest | DevData | DevColGuard | DevColData | Judge (Num×Val) | Wait (Num)

Where

DevRequest::= DevG(DevName, DevOpId, ParamV, Ack, SCSId)

DevData ::= DevV(DevName, ParamName, Var, Ack)

DevColGuard::= DevColG(DevName, SCSId, OpId,ParamV, Ack)

DevColData::= DevColV(DevName, SCSId, ParamName, Var, Ack)

Judge (ParamName,ParameV), are used to determine whether a parameter is compliant with the standard. Wait (Time), are used to wait a certain time. For example, Wait (n) means waiting a period of time n.

Since SCS are complex systems, SCS are tested hierarchically. Test cases gradually are refined to the end test units, called as test atoms. These atoms have separate functions and can be reused frequently. Test atom encapsulates expression sequences of basic test equipment collaboration.

Atom = Precond → AName(Time-Restriction) [TestP]

Normally, test atoms have pre-conditions. The test atom expressions cannot be evaluated until the pre-conditions are satisfied. Meanwhile, as in real-time characteristics, each test atom has time-restriction, which means that whether the result of an atom is valid also depends on whether its time-restriction is satisfied.

In equipment collaboration atom expression, each atom has its time-restriction. When these atoms consist of equipment collaboration expression by composition, as real-time constraints involved, compound operations of equipment collaboration atoms are influenced by time-restriction. Thus special compound operations are defined as time-restriction sequence, time-restriction parallel, time-restriction selection and time-restriction loop. Thus equipment collaboration expressions are as follows.

ColExp::= Skip | Atom | ColExp;t ColExp

| ColExp ||t ColExp | (⟨BoolExp→ColExp⟩)

| (BoolExp→ColExp, ColExp)

Where, Skip is empty expression. Time-restriction sequence ce1; t ce2 means that if ce1 satisfies the time restriction and its result is true, ce2 will be evaluated. Otherwise the expression will return false. Time-restriction parallel, ce1 ||t ce2, means that only if both ce1 and ce2 satisfy the time restriction and results are true, the value of equipment collaboration expression is true. Otherwise the execution will be suspended and return false. Time-restriction selection, b→ce1, ce2, means that if b is true, ce1 will be evaluated and returned the result of ce1. Otherwise, ce2 will be evaluated and returned the result. Time-restriction loop, ⟨b→ce⟩, means that either b or ce is false, the loop will be broken and return current expression value.

V EVALUATING OF EXPRESSIONS OF EQUIPMENT COLLABORATION IN SCS TEST

Here the abstract execution machine [20] of equipment collaboration expressions, namely evaluation rules of test equipment collaboration expressions, is defined to strictly describe evaluation process of equipment collaboration expressions in SCS automatic testing.

The definition of MState is the state pattern of abstract machine which evaluate the equipment collaboration expressions in general SCS test languages. It consists of test device collaboration expression Tescoexp, external environment ExeEnv, and test environment TestEnv. The expression is as follows.

MState = <Tescoexp, ExeEnv, TestEnv>

External environment ExeEnv = (Env, Timer). Where Env is general programming language statement execution environment, Env=Var→Val; Timer is the real time clock.

According the layered structure of equipment, there are three layers for evaluation process of basic equipment collaboration expressions. They are equipment collaboration task layer, test equipment layer and the layer of SCS.

(1)Evaluating rules of equipment collaboration task layer

$$<Judge(v,\upsilon),(\sigma,\tau), (\chi,\rho),(\delta,\gamma)>\Rightarrow<\sigma(v)=?\upsilon, (\sigma,\tau),(\chi,\rho),(\delta,\gamma)>$$

$$<Wait(n),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<\kappa(n,\tau(time),\tau),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>$$

$$\kappa(n, ct, \tau)=\tau(time)-ct< n \rightarrow\kappa(n,ct,\tau) ,true$$

$$\frac{check(d,g,as, \chi) \rightarrow false}{<DevG(d,g,as,ack,pid),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<false,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

$$\frac{check(d,g,as, \chi) \rightarrow true}{\begin{array}{l}<DevG(d,g,as,ack,pid),\\(\sigma,\tau),(\chi,\rho),(\delta,\gamma) >\Rightarrow(true,(\sigma\{true/ack\},\tau),<(g,as,pid),(\chi,\rho),(\delta,\gamma) >^{d})\end{array}}$$

$$<DevV(d,di,pn,v,ack),(\sigma,\tau),(\chi,\theta,\rho),(\delta,\gamma)>\Rightarrow(true,(\sigma\{true/ack\},\tau),<(di,pn,v),(\chi,\rho),(\delta,\gamma) >^{d})$$

$$\frac{check(p,g,as, \delta) \rightarrow false}{<DevColG(d,pid,g,as,ack),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<false,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

$$\frac{check(p,g,as, \delta) \rightarrow true}{\begin{array}{l}<DevColG(d,pid,g,as,ack),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow\\(true,(\sigma\{true/ack\},\tau),<(pid,g,as),(\chi,\rho),(\delta,\gamma) >^{d})\end{array}}$$

$$\begin{array}{l}<DevColV(d,pid,pn,v,ack),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\\\Rightarrow(true,(\sigma\{true/ack\},\tau),<(pid,pn,v),(\chi,\rho),(\delta,\gamma) >^{d})\end{array}$$

(2) Rule of device layer:

$$<(g,as,pid), (\chi,\rho),(\delta,\gamma) >^{d} \Rightarrow ((\chi,\rho'),<s, \gamma>^{pid})\quad (s,\rho_{di}')=g(as,\rho_{di})$$

$$<(di,pn,v),(\chi,\rho),(\delta,\gamma) >^{d} \Rightarrow((\chi,\rho\{\rho di(pn)/v\}),(\delta,\gamma))$$

$$<(pid,g,as), (\chi,\rho),(\delta,\gamma) >^{d} \Rightarrow ((\chi,\rho),<(g,as), (\delta,\gamma)>^{pid})$$

$$<(pid,pn,v),(\chi,\theta,\rho),(\delta,\gamma) >^{d} \Rightarrow((\chi,\theta,\rho),<(pn,v), (\delta,\gamma)>^{pid})$$

(3) Rules of SCS operations.

$$<s, (\delta,\gamma)>^{pid} \Rightarrow(\delta, \Omega(pid,s,\gamma))$$

$$<(g,as), (\delta,\gamma)>^{pid} \Rightarrow(\delta,\gamma_{pid}')\quad \gamma_{pid}'= (g(as), \gamma_{pid})$$

$$<(pn,v), (\delta,\gamma)>^{pid} \Rightarrow(\delta, \gamma\{\gamma_{pid} (pn)/v\})$$

A. Evaluating Rules Of Equipment Collaboration Atoms Expressions

When the pre-condition of device collaboration atom expression is satisfied, the evaluation of equipment collaboration atom expression is started. The evaluation is constrained by time restriction.

$$<p\rightarrow a(rt)[abody],(\sigma,\tau), (\chi,\rho),(\delta,\gamma)>$$

$$\Rightarrow<p, (\sigma,\tau)>\rightarrow$$

$$\begin{array}{l}TR(<abody,(\sigma a,\tau),(\chi,\rho),(\delta,\gamma)>,\tau(time,rt,\\(\sigma a,\tau),(\chi,\rho),(\delta,\gamma)),<false,(\sigma a,\tau), (\chi,\rho),(\delta,\gamma)>\end{array}$$

B. Evaluating Rules Of Equipment Collaboration Expressions

Based on rules of atom expressions, equipment collaboration expressions can be evaluated; following are evaluating rules of equipment collaboration expressions.

(1) Evaluation rule of empty expression.

$$<skip, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow <true, (\sigma,\tau),(\chi,\rho),(\delta,\gamma)>$$

(2) Evaluation rule of time-restriction serial ";t".

$$\frac{<ce1, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow<true, (\sigma',\tau), (\chi,\rho'),(\delta,\gamma') >}{<ce1;tce2,(\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow<ce2, (\sigma',\tau), (\chi,\rho'),(\delta,\gamma')>}$$

$$\frac{<ce1,(\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow<false, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) >}{<ce1;tcet2,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow <false,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

(3) Evaluation rule of time-restriction parellel "||t".

$$\frac{<cet1, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow<false, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) >}{<ce1\|_{t}ce2, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) >\Rightarrow<false, (\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

$$\frac{<ce2, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) > \Rightarrow <false, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) >}{<ce1\|_{t} ce2,(\sigma,\tau),(\chi,\rho),(\delta,\gamma) >\Rightarrow<false,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

$$\frac{\begin{array}{l}<ce1,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<true,(\sigma 1,\tau),(\chi,\rho 1),(\delta,\gamma 1)>,\\<ce2,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<true,(\sigma 2,\tau),(\chi,\rho 2),(\delta,\gamma 2)>\end{array}}{\begin{array}{l}<ce1\|_{t} ce2, (\sigma,\tau),(\chi,\rho),(\delta,\gamma) >\\\Rightarrow <true, (\sigma 1\oplus\sigma 2,\tau),(\chi,\rho 1\oplus\rho 2),(\delta,\gamma 1\oplus\gamma 2)>\end{array}}$$

$$\oplus = \lambda xy. (x\cap y)\bullet(x-(x\cap y))\bullet(y-(x\cap y))$$

(4) Evaluation of time-restriction selection (b→ce1,ce2).

$$\frac{\sigma(b) \Rightarrow true}{<(b\rightarrow ce1,ce2),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<ce1,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

$$\frac{\sigma(b) \Rightarrow false}{<(b\rightarrow ce1,ce2),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<ce2,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

(5) Evaluation rule of time-restriction loop (⟨b→ce⟩).

$$\frac{\sigma(b)\Rightarrow true, <ce,(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow <true, (\sigma',\tau), (\chi,\rho'),(\delta,\gamma')>}{<(\langle b\rightarrow ce\rangle),(\sigma,\tau),(\chi,\rho),(\delta,\gamma)>\Rightarrow<\langle b\rightarrow ce\rangle,(\sigma',\tau),(\chi,\rho),(\delta,\gamma>}$$

$$\frac{\sigma(b)\Rightarrow \text{false or } <ce,(\sigma,\tau), (\chi,\rho),(\delta,\gamma)>\Rightarrow<\text{false}, (\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}{<(\langle b\rightarrow ce\rangle), (\sigma,\tau),(\chi,\rho),(\delta,\gamma)> \Rightarrow <\text{false}, (\sigma,\tau),(\chi,\rho),(\delta,\gamma)>}$$

VI CONCLUSIONS

To satisfy the dynamics and openness of test equipment collaboration in SCS testing, the way that supports equipment collaboration in general SCS test language is given. Here types of equipment collaboration; basic collaboration expression and collaboration atom expression are defined to construct equipment collaboration expressions; and operational semantics and abstract execution machine of evaluation rules of collaboration expressions are defined to describe the process of equipment collaboration. Thus equipment collaboration can be used to design processes of equipment collaboration.

In the future, typing systems of equipment collaboration will be used to study the integrity of semantics of equipment collaboration expressions. And results of equipment collaboration of SCS testing will be extent to automatic testing of safety-critical systems in general.

REFERENCES

[1] K. Liu, S Z G. HAN, J. Wang.et al. Overview on major research plan of trustworthy software. Journal Academic Journal Electronic Publishing House. 2008(3): 145-151. (In Chinese).

[2] Z M. Zheng, S L. Ma, W. Li, et al. Complexity of software trustworthiness and its dynamical statistical analysis methods(In Chinese). Sci China Ser F-Inf Sci, 2009, 52(9): 1651-1657.

[3] Z M. Zheng, S L. Ma, W. Li, et al. Dynamical characteristics of software trustworthiness and their evolutionary complexity (In Chinese). Sci China Ser F-Inf Sci, 2009, 52(8): 1328-1334.

[4] John C. Knight. Safety Critical Systems: Challenges and Directions. ACM ICSE'02, 19-25, May 2002, Orlando, Florida, USA.

[5] Lon D. Gowen. Specifying and Verifying Safety-Critical Software Systems. Seventh Annual IEEE Symposium on Computer-Based Medical Systems. 235-240. 1994.

[6] Q C. Wang. Electrical Test Technology of Spacecraft. Beijing: Press of China science.2007. (In Chinese).

[7] J H. Lv, S L. Ma, X J. Li, S W. Gao. Formal Semantics Model for Automatic Test of Safety Critical Systems. Journal of Software. Vol.25(3):489-505, 2014.

[8] Andrews T, Curbera F, Dholakia H, et al. Business process execution language for web services [OL].http://www.ibm.com/developerworks/library/ws-bpel/, 2003.

[9] Leymann F. Web services flow language (WSFL1.0) [OL].http://www.ibm.com/software/solutions/webservices/pdf/WSFL. pdf . 2001.

[10] Van Der Aalst W, Aldred L, Dumas M, et al. Design and implementation of the YAWL system [A]. Proceedings of 16th International Conference on Advanced Information Systems Engineering [C] .Springer Berlin/ Heidelberg, 2004. 142 -159.

[11] Krishnan S, Wagstrom P, Von Laszew ski G. GSFL: A workflow framework for grid services [OL]. http://www-unix.globus.org/cog/papers/gsf-lpaper.pdf. 2002.

[12] Mcmullen D, Reichherzer T. Identity and Functionality in the Common Instrument Middleware Architecture [J]. Applied Ontology. 2006, 3.

[13] Kawsar F, Nakajima T, Park J H, et al. A document based framework for smart object systems[C]. Future Generation Communication and Networking, 2008. FGCN'08. Second International Conference on, 2008. IEEE, 2008: 178-183.

[14] Kawsar F, Fujinami K, Nakajima T. Potty middleware platform for smart object systems [J]. International Journal of Smart Home. 2008, 2(3): 1-18.

[15] Consortium E. ECHONET Specification ver. 2.11 Part I[J]. ECHONET Overview. 2002.

[16] ECHONET. www.echonet.gr.jp/english/8_kikaku/index.htm.

[17] C. Chen, Helal A. Device integration in SODA using the device description language[C]. Applications and the Internet, 2009. SAINT'09. Ninth Annual International Symposium on, 2009. IEEE, 2009: 100-106.

[18] F .Chen, X H. Rong, P.Deng. A Large-Scale Device Collaborative Process Design Meta-Model And Case Study[C]. The 2nd International Conference on Advanced Computer Theory and Engineering (ICACTE 2009). New York: ASME, 2009. 601-608.

[19] F .Chen, X H. Rong, P.Deng, S L.Ma. A Survey of Device Collaboration Technology and System Software [J]. ACTA ELECTRONICA SINICA, 2011, 39(2):440-447.

[20] P. J. Landin. The Mechanical Evaluation of Expressions.Computer Journal.1964, 6(4):308-320.