一种 SOA 软件系统可信性评价方法研究

赵会群 孙 晶
(北方工业大学计算机系 北京 100144)

摘 要 结合计算机系统可信性评价需求,研究基于 SOA 软件系统可信性评价方法.首先引入 SOA 软件代数模型,给出 SOA 软件可信范式,为从软件体系结构角度研究可信性评价问题奠定基础.给出 SOA 软件元素(服务组件和服务组合)可信属性的定义以及可信性定义;证明可信范式下 SOA 软件系统的 Markov 特性,提出 SOA 软件系统的综合可信性评价模型.通过一个案例解释 SOA 软件系统可信性模型的应用.最后通过与同类研究比较,总结本文的特点和贡献.

关键词 面向服务的体系结构;软件可信性评价;半 Markov 模型
中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2010.02202

A Methodological Study of Evaluating the Dependability of SOA Software System

ZHAO Hui-Qun SUN Jing
(*Department of Computer Science, North China University of Technology, Beijing* 100144)

Abstract In order to meet increasing need for the method of measuring the dependability of computer system, this paper makes the effort to the methodological research of evaluating the dependability of SOA software system: An algebra model of SOA software system and its formal pattern for design a dependable SOA software system are introduced at first, all of the works give a fundamental theory for evaluating the dependability of SOA software system; Define the dependability of each attributes for typical SOA software element, the service component and the service operation, and the dependability of an element that compose with the each attributes. It argues that the SOA software system bears the Markov property if it is deigned under the dependable formal pattern. Based on this argument, it proposes a dependability model for SOA software system. In order to promote the application of the model, a case study is demonstrated. After comparing the difference profiles with other research work it concludes the advantage of the new dependability model.

Keywords service oriented architecture; software dependability evaluation; semi_Markov model

1 引 言

所谓面向服务的体系结构(Service Oriented Architecture,SOA)是一种充分利用 Internet 技术,满足企业对不断增长的业务运营模式需求的应用框架,该模式需要具有灵活、安全和无逢地处理异构、异质的内、外资源的能力[1].由于 SOA 具有广泛的应用前景,所以 IBM、Microsoft、Gartner 等 IT 领军企业先后提出了自己的解决方案[2-4].一些国际标准化组织和研究机构也先后提出了 SOA 参考模型,如 OASIS(Organization for the Advancement

收稿日期:2010-06-08;最终修改稿收到日期:2010-08-26.本课题得到国家自然科学基金(61070030)、北京市属高校学术创新团队建设计划项目(可信物联网关键技术研究及应用示范系统研制)资助.**赵会群**,男,1960 年生,博士,教授,研究领域为软件体系结构、服务计算和软件评价.E-mail: zhaohq6625@sina.com. **孙 晶**,女,1968 年生,硕士,副教授,研究兴趣为软件测试和决策支持系统.

of Structured Information Standards)提出的 SOA 参考模型 1.0[5]、OSOA(Open Service Oriented Architecture)提出的 SOA 编程模型 SCA(Service Component Architecture)等[6]. Web Service 作为一种新型网络信息服务技术,逐渐成为 SOA 软件开发的主要技术支撑之一[7]. 在 Web Service 模式下,用户可以通过基于 WSDL(Web Service Describing Language)[8]的 UDDI(Universal Description, Discovery and Integration)[9]技术请求服务;服务提供商也可以通过 UDDI 发布服务. 虽然 SOA 为企业信息化提供了新的应用模式,但由于 SOA 软件系统开发方式的开放性,得到系统的可信性遇到巨大的挑战.

所谓计算机系统可信性(Dependability)是对系统所提供服务的信任度,应该具有如下属性[10]:

可用性(availability). 系统随时可用;

可靠性(reliability). 系统服务具有连续性;

防危性(safety). 不会给环境带来灾难性的后果;

保密性(confidentiality). 不会发生非法的信息泄露;

完整性(integrity). 不会发生不适当修改信息的现象;

可维护性(maintainability). 进行修改和进入正常态的能力.

在上述属性中,有时把可用性、完整性和保密性综合考虑,统称为安全性(security).

计算机系统可信性研究已有较长的历史,但主要集中在可靠性和安全性研究方面. 可靠性研究起步早、成果丰富,如计算机系统(软件和硬件)的可靠性评价模型,提高计算机系统可靠性的容错技术和测试技术等[10-11];计算机系统的安全性研究主要表现在恶意软件的入侵防御以及信息保密等环节[12-13]. 对软件系统可信性评价研究主要集中在软件系统可靠性度量方面,采用的方法往往直接使用计算机硬件系统的评价模型,如 Jelinski 和 Moranda 提出的基于指数失效时间模型,Goel 和 Okumoto 提出的非齐次泊松过程的软件可靠性模型等等[11]. 然而,软件系统中各成分还不能像硬件那样彼此可拆分,其耦合度一般比较高,所以不能直接根据各个构成元素的可信性评价系统的可信度.

本文研究 SOA 软件系统可信性评价方法. 首先引入 SOA 软件体系结构的代数模型,这是作者提出的一种 SOA 软件系统模型(详细内容参见文献[14]),其中给出了多个 SOA 软件可信范式. 本文在此基础上证明了基于 SOA 软件可信范式的软件系统满足 Markov 特性,从而提出用 Markov 过程模型度量 SOA 软件系统可信性的方法,为从软件体系结构角度研究可信性评价问题奠定基础.

本文第 2 节介绍 SOA 软件代数模型;第 3 节给出 SOA 软件可信性的数学定义;第 4 节证明可信范式下 SOA 软件系统的 Markov 特性后,提出 SOA 软件系统的综合可信性评价模型;在上述研究基础上,第 5 节给出 SOA 软件系统可信性评价的案例,解释其模型的具体应用;最后通过与同类研究比较,总结本文的特点和贡献.

2　SOA 软件代数模型

下面介绍 SOA 软件代数模型. 由于篇幅有限,本节采用非形式化的方式给出 SOA 软件中服务组件、服务组合运算、SOA 软件代数模型的描述性定义以及服务组合运算范式等,更详细的研究参见文献[14].

在 SOA 软件代数模型中,用形式化方式给出了 SOA 服务的定义. 其将服务解释成网构组件,该组件由接口和实现构成. 其中,服务实现是软件程序,接口是服务与外部接触点的集合,而每一个接触点是由〈服务标示、服务功能、外部请求、行为描述、所产生消息集合、行为约束、非功能描述〉.

在 SOA 软件代数模型中还定义了多种服务组合运算,包括“激发”、“使用”、“反馈”、“协同”、“并行”、“重复”、“选择”. 其中,“激发”运算刻画了基于消息的组件组合行为,用 $A \oplus B$ 或 $x \oplus y$ 表示,x 和 y 分别是组件 A、B 的活动,下同;“使用”运算用来描述环境相关的组件组合行为,用 $A \otimes B$ 或 $x \otimes y$ 表示;“反馈”运算用来描述组件“激发”和“使用”组合下对组合结果的需求,用 $A \copyright B$ 或 $x \copyright y$ 表示,$\copyright$ 或者是 $\oplus$,或者是 $\otimes$;“协同”运算用来描述相互配合工作的服务组合行为,用 $A \ominus B$ 或 $x \ominus y$ 表示,$-$ 表示协同条件集合;“并行”运算用来描述多个服务同时工作的服务组合方式,用 $A \parallel B$ 或 $x \parallel y$ 表示,如果并行服务来自于同一个服务组件,也称“并发”;“重复”运算用来描述多个服务循环执行的服务组合行为,用 $A \cdot B$ 表示,当 $A=B$ 时,称 A 重复执行,记为 $x_1 \cdot A$.“选择”运算用来描述服务可替换工作的服务组合行为,用 $A+B$ 或 $x+y$ 表示. 在上述定义基础上,进一步把服务组合解释为服务组合运算的实现,用〈组合标示,服务角色,组合行为,服务消息集合,

服务条件,非功能说明〉表示.其中,"角色"用于描述服务组合运算的具体行为过程,包括〈角色标示,活动描述,活动产生的事件集合,活动的约束条件〉.

在 SOA 软件代数模型中,进一步明确了 SOA 作为代数系统的特征,并给出了 SOA 软件代数模型的定义,把 $S=\langle C,O\rangle$ 称为 SOA 的代数模型,也称代数表达式,C 和 O 分别是服务组件集合和服务组合运算集合.在代数模型定义基础上,进一步定义了 SOA 同构、同态以及 SOA 服务相等、进化等概念,从而建立了完备的 SOA 软件代数模型理论体系,为进一步用代数方法解决 SOA 服务软件性质奠定了基础.

SOA 软件代数模型下的可信软件范式包括:

(1)服务组件依赖第一范式,即"若任意一个组件都至少与另一个组件发生组合";

(2)服务组合第二范式,即"SOA 软件中存在多个核心服务组件",所谓的核心服务组件是指所有服务组件都与其发生服务组合的服务组件;

(3)SOA 软件可信范式 DNF,即在满足第一范式的基础上,所有的服务依赖均为"激发运算组合".

在进一步明确 SOA 服务组合代数表达式标准型后,可以给出 SOA 软件可信性范式.

SOA 服务组合代数表达式都可以写成下面的标准型.如任意一个 $S=\langle C,O\rangle$ 的代数表达式的标准型可以写成:

$$S=a_1\cdot C_1+a_2\cdot C_2+\cdots+a_k\cdot C_k$$

或简写成

$$S=a_1C_1+a_2C_2+\cdots+a_kC_k \tag{1}$$

其中,C_i 可以是经过其它组合运算后得到的服务组件.

(1)SOA 安全范式,即 $S=a_1\cdot C_1+a_2\cdot C_2+\cdots+a_k\cdot C_k$.其中 a_i 是明确的、为了实现防危性保证的活动;

(2)服务可达性范式,即式(1)为非递归表达式.

为了更好地辅助 SOA 软件建模与可信性分析,给出以下协同服务组合运算规则,这些规则参考了进程代数中算子的运算规则:

(1)协同组合运算扩展规则.设 $F=a_1f_1+a_2f_2+\cdots+a_kf_k$;$G=b_1g_1+b_2g_2+\cdots+b_kg_k$ 是两个服务组件,那么

$$\begin{aligned}F\ominus G=&(a_1f_1\ominus G)+\cdots+(a_mf_m\ominus G)+\\&(b_1F\ominus g_1)+\cdots+(b_nF\ominus g_k)+\\&\Sigma(a_i\ominus b_j)f_k\ominus g_k\end{aligned} \tag{2}$$

a_i,b_j 为协同服务活动对,$a_i\ominus b_j$ 记为 ε;$a_i\cdot f_i$ 简写为 a_if_i;$\Sigma(a_i\ominus b_j)\cdot f_k\ominus g_k$ 表示多项选择.

(2)隐藏内部协同常量规则.$\varepsilon\cdot F=F$;$a\cdot\varepsilon\cdot F=a\cdot F$.

(3)限制规则.在式(2)中,如果把常量 $a_i(i=1,2,\cdots,m)$ 或 $b_j(j=1,2,\cdots,n)$ 列入限制范围,那么式(2)的扩展式中 $a_if_i\ominus G$ 和 $F\ominus b_jg_j$ 可以消去.

命题. 设 $a_i(i=1,2,\cdots,m)$ 和 $b_j(j=1,2,\cdots,m)$ 是可信协同常量对,则在式(2)中 $a_if_i\ominus G$ 和 $F\ominus b_jg_j$ 可以消去.

3 SOA 软件可信性概念

本节首先讨论 SOA 构成元素的安全、可靠、防危、可维护等可信属性的数学定义,在此基础上讨论 SOA 软件系统的可信性定义.

首先给出 SOA 中服务组件或服务运算的安全性定义.

定义 1. 令 $\forall c_k\in C,S=\langle C,O\rangle$,$F_k$ 为执行 c_k 前排除"隐患"的度量.让

$$Se=\begin{cases}f_0, F_k\geq f_0\\ f_1, f_0>F_k\geq f_1\\ f_2, f_1>F_k\geq f_2\\ \vdots\\ f_n, f_{n-1}>F_k\geq f_n\end{cases} \tag{3}$$

则把 SOA 元素 c_k 的安全性定义为执行 c_k 前能够排除高于某种程度隐患度量的概率,记为 $P\{Se=f_k\}=P_{Se}$.

在上述定义中,$S=\langle C,O\rangle$ 元素可以为服务组件,也可以为服务运算.这里"被排除"是指不会再次发生此类安全"隐患",而"隐患"是指 SOA 软件代码中的设计缺陷,这些缺陷可能会造成软件运行时出现错误.软件缺陷是错误发生的根源,但并非每一个缺陷都产生错误.这里只关注可以产生软件系统的不可用、信息泄露和信息不完整这三类错误的缺陷.所谓"隐患"的度量在不同关注点上,有其不同的解释.在关注可以产生软件系统可用性缺陷时,可以把度量解释为软件的缺陷数量;在考虑可以产生信息泄露的缺陷时,可以把度量理解成排除漏洞数量,或者安全措施的等级(如排除"病毒"的数量)等;在讨论数据完整性时,可以把度量定义成纠正不完整或不一致数据的记录数等等.在关注所有类型缺陷时,可以用加权平均值作为度量.

下面定义 SOA 软件元素的可靠性.

定义 2 令 $\forall c_k\in C,S=\langle C,O\rangle$,$W_k$ 表示执行 c_k

时错误发生数量的度量. 让

$$R=\begin{cases} w_0, W_k \le w_0 \\ w_1, w_0 < W_k \le w_1 \\ w_2, w_1 < W_k \le w_2 \\ \vdots \\ w_n, w_{n-1} < W_k \le w_n \end{cases} \tag{4}$$

则把 SOA 元素的可靠性定义为执行 c_k 后不发生高于某种程度错误度量的概率，记为 $P\{R=w_k\}=P_R$.

下面给出 SOA 软件元素的防危性定义.

定义 3.　令 $\forall c_k \in C, S=\langle C,O\rangle$，$L_k$ 表示执行 c_k 失败后代价的度量. 让

$$Sa=\begin{cases} l_0, L_k \le l_0 \\ l_1, l_0 < L_k \le l_1 \\ l_2, l_1 < L_k \le l_2 \\ \vdots \\ l_n, l_{n-1} < L_k \le l_n \end{cases} \tag{5}$$

则把 SOA 元素的防危性定义为执行 c_k 后不发生高于某种程度代价失败的概率，记为 $P\{Sa=l_k\}=P_{Sa}$.

下面给出 SOA 软件元素可维护性定义.

定义 4.　令 $\forall c_k \in C, S=\langle C,O\rangle$，$M_k$ 为 c_k 中可修复错误数量的比率. 让

$$M=\begin{cases} m_0, M_k \ge m_0 \\ m_1, m_0 > M_k \ge m_1 \\ m_2, m_1 > M_k \ge m_2 \\ \vdots \\ m_n, m_{n-1} > M_k \ge m_n \end{cases} \tag{6}$$

则把 SOA 元素的可维护性定义为：c_k 中高于某种程度错误修复率的概率，记为 $P\{M=w_k\}=P_M$.

在上述定义中，并没有区分服务组件和服务运算. 实际上，从 SOA 软件角度二者都可以理解成一段代码，其可信属性可以不加以区别.

SOA 软件的设计缺陷是运行错误的根源，但并非每一个缺陷都会产生错误，通过增加可信手段可以减少错误的发生，如对软件进行测试等；运行错误又是 SOA 软件运行结果失败的起因，但并非所有的错误都会导致运行结果失败，可以通过软件容错等可信技术降低导向失败的程度. 缺陷、错误和失败发生在 SOA 软件生命周期中的不同阶段，安全性、可靠性、防危性和可维护性是对不同阶段可信行为质量的度量.

在上述定义基础上，可以定义 SOA 软件元素的可信性.

定义 5.　令 $\forall c_k \in C, S=\langle C,O\rangle$，它的可信性为

$$D_{C_k}=\alpha_1\times P_{Se}+\alpha_2\times P_R+\alpha_3\times P_{Sa}+\alpha_4\times P_M \tag{7}$$

其中，$\Sigma\alpha_i=1 (i=1,2,3,4)$，$\alpha_i$ 称为可信加权系数.

在上述定义中，可以根据实际关注的某种可信属性，调整相应的 α_i 值. 虽然 D_{C_k} 由代数表达式求得，但仍然具有概率属性. 当 P_{Se}、P_R、P_{Sa} 和 P_M 取得最大值 1 时，$D_{C_k}=1$；当 P_{Se}、P_R、P_{Sa} 和 P_M 取得最小值 0 时，$D_{C_k}=0$.

4　SOA 软件系统可信性模型

以下分单域和多域两种情况讨论 SOA 软件系统的可信模型. 单域（所有元素来自于相同的领域 Domain）多以“使用”、“激发”、“重复”和“选择”运算形式完成服务组合；而在多域（元素来自于不同的领域）环境下需要使用“协同”和“并行”等运算实现域间的服务组合.

4.1　单域可信性模型

第 2 节中给出了 SOA 软件可信性范式 DNF. 对满足 DNF 范式的 SOA 软件，由于所有的元素是逻辑独立的，所以单域软件系统满足半 Markov 过程的条件. 为此，我们给出如下定理.

定理 1. 如果一个单域 SOA 软件系统 $X(t)$ 满足 DNF 范式，对 $0=t_0\le t_1\le\cdots\le t_n\le T$ 以及 $X_i=X(t_i)$ 的所有可能状态 $a_1, a_2, \cdots, a_n$，有 $P\{X_n=a_{in}, T_n-T_{n-1}\le t \mid X_{n-1}=a_{in-1}, \cdots, X_1=a_{i1}\}=P\{X_n=a_{in}, T_n-T_{n-1}\le t \mid X_{n-1}=a_{in-1}\}$ 成立，并且与 n 无关.

证明.　假设单域 SOA 软件系统 $X(t)$ 有 k 个服务组件构成. 把每一个组件看成系统 $X(t)$ 在某一时刻的一个运行状态，$X(t)$ 在 $t_i (i=1,2,\cdots,n)$ 发生状态转移，记 $X_i=X(t_i)$，$X(t)$ 在某一个状态的逗留时间为 $T_{i+1}-T_i$，在证明中为了书写方便，暂不写逗留时间；此外，另设一个失效状态 F 和一个成功状态 S，不失一般性把状态记为 $a_1, a_2, \cdots, a_n (n\ge k+2)$，而对 $\forall i\in\{1,2,\cdots,k\}$，第 i 次运行记为 $a_{i1}, a_{i2}, \cdots, a_{in}$. 下面用归纳法证明定理的正确性.

(1) 当 $n=3$ 时，系统 X 运行状态序列为 a_{i1}, a_{i2}, a_{i3}；如果它们为组件，它们一定是 $a_{i1}\oplus a_{i2}$，又 $a_{i2}\oplus a_{i3}$，而 a_{i1} 与 a_{i3} 无关；否则，如果 a_{i1} 与 a_{i3} 有关，就有在同一时刻 $a_{i1}\oplus a_{i2}$、$a_{i2}\oplus a_{i3}$ 和 $a_{i1}\oplus a_{i3}$，这时状态序列应为 $a_{i1}, a_{i2}, a_{i3}, a_{i2}, a_{i1}, a_{i3}$ 或 $a_{i1}, a_{i3}, a_{i1}, a_{i2}, a_{i3}$，这与 a_{i1}, a_{i2}, a_{i3} 是系统 X 的一次执行状态序列矛盾. 如果 a_{i1}, a_{i2}, a_{i3} 中 a_{i3} 是 F 或 S，显然成立；若 a_{i1}, a_{i2} 之一是 S 或 F，则与 a_{i1}, a_{i2}, a_{i3} 是 X 的一次运行状态序列矛盾. 所以当 $n=3$ 时定理成立.

(2)假设当 $n=K-1$ 时定理成立,即有 $P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}=P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2}\}$ 成立,去证明 $n=K$ 时有 $P\{X_k=a_{ik}\mid X_{k-1}=a_{ik-1},\cdots,X_1=a_{i1}\}=P\{X_k=a_{ik}\mid X_{n-1}=a_{ik-1}\}$.

$$
\begin{aligned}
&P\{X_k=a_{ik}\mid X_{k-1}=a_{ik-1},\cdots,X_1=a_{i1}\}=\\
&P\{X_k=a_{ik},X_{k-1}=a_{ik-1},\cdots,X_1=a_{i1}\}/\\
&P\{X_{k-1}=a_{ik-1},\cdots,X_1=a_{i1}\}=\\
&P\{X_k=a_{ik},X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}/\\
&P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}=\\
&P\{X_k=a_{ik},X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}/\\
&P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2}\}
\end{aligned}
\tag{8}
$$

对 $P\{X_k=a_{ik},X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}$,若 $X_k=a_{ik}$ 与 $X_{ij}=a_{ij}$ $(1\leq j\leq k-2)$ 相关,就有 $a_{ij}\oplus a_{ik}$ 和 $a_{ij}\oplus a_{ij+1}$,X 的第 i 次运行的状态序列为 $\cdots a_{ij},a_{ik},a_{ij+1}\cdots$ 或是 $\cdots a_{ij},a_{ij+1},a_{ik}\cdots$,这与 $a_{i1},a_{i2},\cdots,a_{ik-2},a_{ik-1},a_{ik}$ 是 X 的第 i 次运行的状态序列矛盾.所以只能有 $a_{ik-1}\oplus a_{ik}$,这样式(8)为

$$
\begin{aligned}
&P\{X_k=a_{ik}\mid X_{k-1}=a_{ik-1}\}\times\\
&P\{\ X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2},\cdots,X_1=a_{i1}\}/\\
&P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2}\}=\\
&P\{X_k=a_{ik}\mid X_{k-1}=a_{ik-1}\}\times\\
&P\{\ X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2}\}/\\
&P\{X_{k-1}=a_{ik-1}\mid X_{k-2}=a_{ik-2}\}=\\
&P\{X_k=a_{ik}\mid X_{k-1}=a_{ik-1}\}.
\end{aligned}
$$

当 $n=k$ 得证.根据(1),(2)的证明有定理 1 成立.

证毕.

由定理 1 得知,满足 DNF 的单域 SOA 软件系统运行 $X(t)$ 满足半 Markov 过程条件.所以可以利用 Markov 模型计算单域 SOA 软件系统的可信性.

推论 1. 设一个单域 SOA 软件系统 $X(t)$,如果 $X(t)$ 满足 DNF 范式,则 $X(t)$ 的可信性可以由下列公式计算.

$$
\begin{pmatrix}C_{1n}\\C_{2n}\\\vdots\\C_{nn}\end{pmatrix}=\begin{pmatrix}Q_{11}&Q_{12}&\cdots&Q_{1n}\\Q_{21}&Q_{22}&\cdots&Q_{2n}\\\vdots&\vdots&Q_{ij}&\vdots\\Q_{n1}&Q_{n2}&\cdots&Q_{nn}\end{pmatrix}\times\begin{pmatrix}C_{1n}\\C_{2n}\\\vdots\\C_{nn}\end{pmatrix}\tag{9}
$$

其中:$Q_{ij}(t)=P_{ij}\times d_i(t)$,$P_{ij}$ 为服务组件 $i\sim j$ 的转移概率与组合运算可信性的乘积,即 $P_{ij}=t_{ij}\times o_i(t)$;$d_i(t)$ 为服务组件的可信性,即 $P\{D=d_i\mid T_{K+1}-T_K\leq t\}$ 为组件 i 的可信性,在式(9)中设 C_{1n} 是开始状态,C_{nn} 是成功状态.

4.1 多域可信性模型

在多域 SOA 软件系统中,域间子系统的服务组合采用“协同”或“并行”运算进行.根据 SOA 软件代数模型中定义,“并行”是“协同”的特例,所以只讨论“协同”组合即可.

“协同”组合的物理实现是通过相互交换信息的协议来完成,而协议的本质是对等体的相互“激发”,所以多域 SOA 软件系统 $X(t)$ 也有 Markov 过程特性.

定理 2. 设 $X(t)$ 是一个多域 SOA 软件系统,如果域间可信子系统的服务组合采用“协同”运算进行,则对 $0=t_0\leq t_1\leq t_2\leq\cdots\leq t_n\leq T$ 以及 $X_i=X(t_i)$ 的所有可能状态 $a_1,a_2,\cdots,a_n$,有 $P\{X_n=a_{in},T_n-T_{n-1}\leq t\mid X_{n-1}=a_{in-1},\cdots,X_1=a_{i1}\}=P\{X_n=a_{in},T_n-T_{n-1}\leq t\mid X_{n-1}=a_{in-1}\}$ 成立,并且与 n 无关.

证明. 设 $F=a_1f_1+a_2f_2+\cdots+a_kf_k$;$G=b_1g_1+b_2g_2+\cdots+b_kg_k$ 是不同域的两个可信服务.根据 SOA 软件代数模型中的“协同组合运算扩展规则”有

$$
\begin{aligned}
F\ominus G=&(a_1f_1\ominus G)+\cdots+(a_mf_m\ominus G)+\\
&(b_1F\ominus g_1)+\cdots+(b_nF\ominus g_k)+\\
&\Sigma(a_i\ominus b_j)f_k\ominus g_k
\end{aligned}
\tag{10}
$$

由于 F 与 G 均为可信服务,所以满足代数模型中的所有可信范式.如果把 a_i、b_j 解释成执行可信行为的活动对,则根据“限制规则”和“隐藏内部协同常量规则”,式(10)可以写成

$$F\ominus G=\Sigma f_k\ominus g_k.$$

又根据“协同”组合$\ominus$的定义,有 $\exists x\in Publ(f_k)$,$\exists y\in Extn(g_k)$,$[(pre\text{-}cond(g_k))\wedge(y)\Rightarrow((pre\text{-}cond(f_k))\wedge(x)]$,反之 $\exists x\in Extn(f_k)$,$\exists y\in Publ(g_k)$ 使得 $[(pre\text{-}cond(f_k))\wedge(x)\Rightarrow((pre\text{-}cond(g_k))\wedge(y)]$,可得

$$
\begin{aligned}
f_k\ominus g_k=&[(pre\text{-}cond(g_k))\wedge(y)\Rightarrow\\
&((pre\text{-}cond(f_k))\wedge(x)]=\\
&[(pre\text{-}cond(g_k))\wedge pre\text{-}cond(f_k))\wedge((y)\Rightarrow(x))]
\end{aligned}
$$

或者

$$
\begin{aligned}
f_k\ominus g_k=&[(pre\text{-}cond(f_k))\wedge(x)\Rightarrow\\
&((pre\text{-}cond(g_k))\wedge(y)]=\\
&[(pre\text{-}cond(f_k)\wedge pre\text{-}cond(g_k))\wedge((x)\Rightarrow(y))].
\end{aligned}
$$

又根据“激发”组合$\oplus$的定义,有 $\exists x\in Extn(f_k)\wedge\exists y\in Publ(g_k)$ 使得 $[pre\text{-}cons(f_k)\wedge pre\text{-}cons(g_k)\wedge(x\Rightarrow y)]\wedge[Msgs(x)\rightarrow Msgs(y)]$.

所以 $f_k\ominus g_k\wedge((Msgs(x)\rightarrow Msgs(y))\vee(Msgs(y)\rightarrow Msgs(x)))=f_k\oplus g_k\vee g_k\oplus f_k$.

因此,当采用消息传递方式协同时,多域 SOA 软件系统可以由激发组合表示,即 $X(t)$ 满足 DNF 范式.根据定理 1,定理 2 得证. 证毕.

推论 2. 设 $X(t)$ 是一个多域 SOA 软件系统,如果 $X(t)$ 中的每个单域子系统均为可信服务,则

$X(t)$ 的可信性为

$$\begin{pmatrix} C_{1n} \\ C_{2n} \\ \vdots \\ C_{nn} \end{pmatrix} = \begin{pmatrix} Q_{11} & Q_{12} & \cdots & Q_{1n} \\ Q_{21} & Q_{22} & \cdots & Q_{2n} \\ \vdots & \vdots & Q_{ij} & \vdots \\ Q_{n1} & Q_{n2} & \cdots & Q_{nn} \end{pmatrix} \times \begin{pmatrix} C_{1n} \\ C_{2n} \\ \vdots \\ C_{nn} \end{pmatrix} \quad (11)$$

其中：$Q_{ij}(t)=P_{ij}\times d_i(t)$，$P_{ij}$ 为服务组件 $i\sim j$ 的转移概率与组合运算可信性的乘积，即 $P_{ij}=t_{ij}\times o_i(t)$；$d_i(t)$ 为服务组件的可信性，即 $P\{D=d_i \mid T_{K+1}-T_K\leq t\}$ 为组件 i 的可信性，在式(11)中设 C_{1n} 是开始状态，C_{nn} 是成功状态，C_{in} 可能属于不同的单域子系统.

5　可信性模型的应用

本节通过一个案例解释 SOA 软件可信模型的应用. 由于篇幅有限，这里仅给出模型应用过程中参数的获取方法.

5.1　案例设计

设计一个由多家旅行社构成的飞机订票业务服务系统，如图 1 所示. 其中，Traveler 为被服务的旅行者、Airline 表示航空公司、$\text{Agent}_1\sim\text{Agent}_4$ 表示提供飞机订票服务的旅行社，这些旅行社之间已经建立了业务往来关系，用箭头表示，如 Agent_3 可以代理 Agent_1 提交的订票服务请求等等. 现该服务系统已经实现了基于 SOA 的订票服务 Web 系统，该系统由 5 个 Web Service 子系统构成，分别称为 $\text{Agent}_1\sim\text{Agent}_4$ 和 Airline，每一个子系统都通过 WSDL 提供订票服务接口，同时把服务发布到 UDDI 上，其它子系统可以根据权限在 UDDI 上发现服务. 现各个子系统已经根据商业合同建立了相对稳定的业务逻辑，并建立了各自的 BPEL 业务逻辑描述.

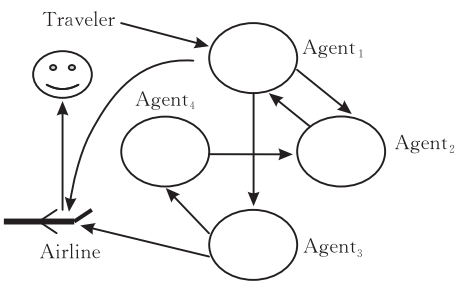

图 1　业务服务流

上述 SOA 软件系统是一个松耦合连接的基于消息的系统，符合 SOA 软件代数模型中可信范式 DNF 条件. 根据定理 2，该系统满足 Markov 特性，它的 Markov 状态转换图如图 2 所示. 在图 2 中 C_1、C_2、C_3、C_4 和 C_5 分别表示 Agent_1、Agent_2、Agent_3、Agent_4 和 Airline；C_5 是成功状态，F 为增加的失败状态. 根据实际业务发生情况统计，各元素的转移概率如表 1 所示.

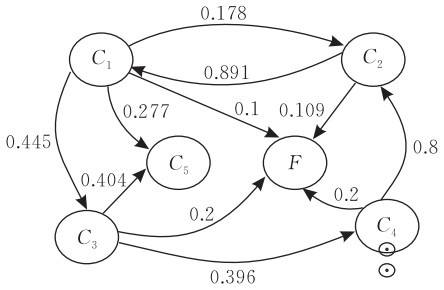

图 2　状态转换图

表 1　各个服务元素的转移概率

	概率					
	C_1	C_2	C_3	C_4	C_5	F
C_1		0.178	0.445		0.277	0.100
C_2	0.891					0.109
C_3				0.396	0.404	0.200
C_4		0.800				0.200

5.2　可信性评价

为了获得订票服务系统可信性评价所需的数据，设计实现了一个测试系统. 通过配置 Web 服务和 JUDDI 服务注册[18]，模拟各旅行社、订票代理和航空公司的服务业务；通过建立服务的 Net Beans BPEL 描述[19]到 TTCN-3[20]测试套的映射，实现各个订票服务的动态测试，从而获得各种可信性度量的数据. 由于篇幅有限，这里仅给出可靠性测试套的部分 TTCN-3 代码(见图 3)，其它工作可以类似进行.

```
testcase TC_1() runs on mtcType system sysType
  { var ptcType ptc;
    ptc := ptcType.create; //定义被测实现端口
    map(ptc:ptcPort, system:sysPort); //映射被测实现端口
    ptc.start(ptcBehavior(Request_1,Response_1));
                                  //执行测试功能
    ptc.done; }
  control
    {execute (TC_1());} //执行测试套
  }
function ptcBehavior(in wsRequestType wsRequest,
        //wsResponseType wsResponse)runs on ptcType
    {ptcPort.send(wsRequest);
    localTimer.start;
    alt { [] ptcPort.receive(wsResponse) //接收到正常反馈
          {localTimer.stop; //停止始终
            setverdict(pass); } //返回测试成功裁决
        [] ptcPort.receive //接收到非正常反馈
          {localTimer.stop;
            setverdict(fail); } //返回结果错误的裁决
        [] localTimer.timeout //在指定时间内没有回应
            {setverdict(fail); } //返回结果错误的裁决
    }}
```

图 3　可靠性测试套

经实验测得 C_1、C_2、C_3、C_4 和 L 的安全性、可靠性、防危性和可维护性度量如表 2 所示. L 表示服务连接器,既 Web Service 系统,它有相同的可信属性. 设可信性权值 α_i 分别为:$\alpha_1=0.25$、$\alpha_2=0.25$、$\alpha_3=0.25$、$\alpha_4=0.25$,则每个服务元素的可信性度量可以由式(7)求得. 为了获得相对真实的评价结果,测试过程中增加了一些故障,如修改 Web Service 接口信息,并不立即更新 UDDI 注册信息表,从而造成订票结果错误;再如,人为造成网络中断,从而造成访问不成功等等.

表 2 各个服务元素的可信性属性值以及可信性计算结果

	安全性	可靠性	防危性	可维性	可信性
C_1	0.89	0.90	0.99	0.99	0.95
C_2	0.88	0.90	0.99	0.99	0.94
C_3	0.87	0.95	0.99	0.99	0.95
C_4	0.90	0.94	0.99	0.98	0.95
L	0.99	0.99	0.99	0.99	0.99

根据表 1 和表 2 可得可信转移概率 $Q_{ij}(t)=P_{ij}\times d_i(t)$ 的分布如表 3.

表 3 各个服务元素的可信转移概率

	概率					
	C_1	C_2	C_3	C_4	C_5	F
C_1		0.167	0.418		0.260	0.094
C_2	0.829					0.101
C_3				0.372	0.379	0.188
C_4		0.752				0.188

以 C_1 到 C_5 的可信性为例

$$\begin{cases} D_{1,5}=0.226+0.167D_{2,5}+0.418D_{3,5} \\ D_{2,5}=0.829D_{1,5} \\ D_{3,5}=0.379+0.372D_{4,5} \\ D_{4,5}=0.752D_{2,5} \end{cases} \tag{12}$$

解式(12)有:C_1 到 C_5 的可信性为 0.468.

该案例仅讨论了 C_1 到 C_5 的可信性. 当然,也可以求任何一个元素到成功的可信性,只是本例中业务流从 C_1 开始. 从可信值高低角度,该案例的可信性不高,这是因为在综合考虑系统可信性时,系统的可信性会受到各个服务本身可信属性以及状态转换率的干扰. 因此,在实际可信软件工程中,不但要增加代码的可信性,也要加强业务流程的可信管理.

6 相关研究比较及结论

前面基于 SOA 软件代数模型,给出了服务元素可信属性的定义和可信性评价模型. 该模型的客观性和实际意义可以从以下两个方面考虑.

(1) 传统的软件可信性评价方法中,如软件可靠性,是把可信性定义成:"不发生故障的概率"[10-11]. 但就目前的软件开发、验证和测试技术,还无法保证软件不发生故障,也无法准确地预知软件发生故障的概率,所以传统方法中只能用概率论模型进行预测,所采用的模型也都是直接使用计算机硬件的可信性模型.

由于传统方法中对可信性定义有不切实际的要求,所以带来可信性评价的客观性和实际意义不强. 与传统方法比较,本文首先考虑到软件故障测试的不完全性,用 SOA 软件的可信属性的实际发生值(分段函数)的概率作为可信性的度量,符合 SOA 软件分布、松耦合和质量无法集中控制的具体情况. 因此,提出的可信评价方法与传统方法比较有较好的客观性和实际意义.

(2) 从模型实际使用考虑,可信属性计算的工作量主要在 $Q_{ij}(t)=P_{ij}\times d_i(t)$ 中 P_{ij} 和 $d_i(t)$ 的计算方面. 其中 P_{ij} 可以通过测试工具获得的可信属性数据计算,本文给出了基于 TTCN-3 的可信属性数据获取方法;而 $d_i(t)$ 是 SOA 软件中各种服务的可信性度量,从服务规范性角度,服务的可信属性应该由服务方提供,是可信性模型计算的已知量. 因此,提出的可信性评价方法具有可操作性.

有关可信性基础理论研究已经有较长的历史,但主要关注计算机系统的可靠性和安全性等方面,对软件系统可信性研究更是如此. 如 Gutjahr 和 Yang[15-16] 仅给出了可靠性、防危性等部分可信属性定义,仅讨论了部分可信属性的评价问题. 本文给出了可信所有属性的定义,从概念上完善了可信性理论体系. 在对可信属性描述上也采用了与 Gutjahr 不同的方法,Gutjahr 在定义防危性和可靠性前首先定义一个失败函数 $Loss()$,在此基础上定义了软件失败的代价函数 $Cost()$,通过一个指定的阈值分类定义不同程度防危性和可靠性. 与该定义不同,本文采用了多级分类的方法区分不同程度的可信属性,各个级别可以根据需要灵活调整,更符合实际需要,而且更加严谨. Liu 和 Zhu[17] 提出一个 Web 服务 QoS 可信性评价模型,并给出了两种 Web 服务 QoS 可信性评价方法. 该方法并没有在常规的可信性分类基础上讨论可信性的评价,与本文研究的关注点不同.

分别对单域和多域 SOA 软件系统可信性模型进行讨论,在严格证明 SOA 软件的半 Markov 特性后,给出了 SOA 软件系统可信性模型. Markov 可

靠性模型在计算机硬件系统中已经被广泛使用，也有人用 Markov 模型评价软件系统的可信性，但一般不加以证明，如 Walter 和 Shiping，这是一个错误的用法. 实际上，只有在软件元素相互逻辑独立的情况下才满足 Markov 特性. 本文严格地限制 SOA 软件满足 DNF 范式正是出于这种考虑. DNF 并非苛刻的要求，事实上任何一个域内服务组合均可转换成“激发”组合[14].

通过上述研究得出如下结论：SOA 软件系统可以通过实现代码独立的服务组件组合来实现；服务组件可以有多种组合方式，满足 DNF 范式的 SOA 软件可以根据构成元素的可信性计算系统的可信性.

参 考 文 献

[1] Patrick F Carey, Bernard W Gleason. Solving the integration issue- service-oriented architecture[EB/OL]. http://www.zdnet.co.uk/tsearch/Service-Oriented + Architecture.htm, Feb. 2006

[2] David Sprott, Lawrence Wilkes. Understanding service-oriented architecture, CBDI[EB/OL]. http://www.microsoft.com/china/MSDN/library/architecture/, Jan. 2004

[3] Microsoft Whitepaper. Enabling real world SOA through the Microsoft platform[EB/OL]. http://msdn2.microsoft.com/en-us/architecture/aa948857.aspx, Sep. 2007

[4] Yefim V Natis. Service-oriented architecture scenario. Gartner Group[EB/OL]. http://www.gartner.com/resources/114300/114358/114358.pdf, Jun. 2003

[5] OASIS Committee Specification 1, 2. Reference model for service oriented architecture 1.0[EB/OL]. http://www.oasis-open.org, Aug. 2006

[6] SCA Service Component Architecture-SCA Assembly Model V1.00[EB/OL]. http://www.osoa.org/display/Main/, May 2007

[7] W3C Working Group Note. Web services architecture[EB/OL]. http://www.w3.org/TR/ws-arch/, Feb. 2004

[8] Christensen E, Curbera F, Meredith G, et al. WSDL - Web service description language, Version 1.1. Standard. W3C [EB/OL]. http://www.w3.org/TR/wsdl/, 2001

[9] Bellwood T, Clement L, von Riegen C. UDDI-Universal Discovery, Description, and Integration, Version 2.0. Standard UDDI. org[EB/OL]. http://www.uddi.org/ipubs/uddi_v3.htm, Jul. 2002

[10] Avizienis A, Laprie J-C, Randell B, et al. Fundamental concept of dependability[EB/OL]. http://www.cert.org/research/isw/isw2000/papers/56.pdf, Feb. 2000

[11] Michael R Lyu. Handbook of software reliability engineering. McGraw-Hill Companies, 1996(in Chinese)
(Michael R Lyu, 刘喜成等译. 软件可靠性工程手册. 北京：电子工业出版社，1997)

[12] Avizienis A, Laprie J-C, Randel B, et al. Basic concepts and taxonomy of dependable and secure computing. IEEE Transactions on Dependability and Secure Computing, 2004, 1(1): 11-33

[13] Jonsson E. A quantitative model of the security intrusion process based on attacker behavior. IEEE Transactions on Software Engineering, 1997, 23(4): 235-245

[14] Zhao Hui-Qun, Sun Jing. An algebraic model of service oriented trustworthy software architecture. Chinese Journal of Computers, 2010, 33(5): 890-900(in Chinese)
(赵会群，孙晶. 面向服务的可信软件体系结构代数模型. 计算机学报，2010，33(5)：890-900)

[15] Gutjahr W J. Software dependability evaluation based on Markov usage models. Perform. Evaluation, 2000, 40(4): 199-222

[16] Yang Shi-Ping, et al. Research on dependability evaluation based on Markov model. Couputer Enginnering and Application, 2003, 30(12): 40-48(in Chinese)
(杨仕平等. 基于马尔可夫模型的可信性评估研究. 计算机工程应用，2003，30(12)：40-48)

[17] Liu Guo-Qi, Zhu Zhi-Liang. Model for evaluating QoS trustworthiness of Web service. Journal of Chinese Computer Systems, 2009, 30(11): 2216-2221(in Chinese)
(刘国奇，朱志良等. 一种 Web 服务 QoS 可信性评价模型. 小型微型计算机系统，2009，11，30(11)：2216-2221)

[18] Apeche: JUDDI. http://uddi.xml.org/apache.juddi

[19] Net Beans. http://zh-cn.netbeans.org/download/6.8/ml/

[20] ETSI ES 201 873-1 V2.2.1, Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3, Part 1: TTCN-3 Core Language: 74-85.02.2003

ZHAO Hui-Qun, born in 1960, Ph. D., professor. His research interests include software architecture, service computing and software evaluation.

SUN Jing, born in 1968, M. S., associate professor. Her research interests include software testing and decision support system.

Background

Service-oriented architecture (SOA) has been widely used as a framework for structuring software systems centered around the concept of service to provide integrated services to business processes. One of the important measures of the quality of the service software is dependability that ensures the safety and reliability of the services. Although a lot of efforts have been made towards dependable SOA, few methodologies have been reported for measuring and evaluating the dependability of SOA software. The user or developer have to employ the model effected for evaluating hardware system, for example the Jelinski-Moranda model and the A. L. Goel-K. Okumoto model. However, there are quite different property on states transition between the software and the hardware system. For hardware system it satisfies the Markov property which the Jelinski-Moranda model and the A. L. Goel-K. Okumoto model based, for software system it is hard to follow.

In order to meet increasing need for the method of measuring the dependability of computer system, this paper makes the effort to the methodological research of evaluating the dependability of SOA software system. An Algebra Model of SOA software system and its formal pattern for design a dependable SOA software system are introduced.

This work have been supported by National Natural Fund of China (NSFC in short), Grant No. 61070030, Beijing Natural Science Fund, Grand No. 4062012, and Beijing Education Committee Fund for the team of academic innovation. All the work above has taken an important role on theory aspects in both Project of NSFC and the Project of Beijing Natural Science Fund.