软件体系结构的行为求精

李长云，文志华，廖立君

（湖南工业大学 计算机与通信学院，湖南 株洲 412008）

摘 要：在D-ADL规约框架下，给出了软件体系结构的行为求精形式化规则：构件的行为求精应符合进程观察弱模拟关系，连接件的行为求精应符合进程分支弱模拟关系。

关键词：软件体系结构；求精；π演算

中图分类号：TP311 **文献标识码**：A **文章编号**：1673-9833(2007)05-0021-04

The Behavior Refinement of Software Architecture

Li Changyun，Wen Zhihua，Liao Lijun

（School of Computer and Communication，Hunan University of Technology，Zhuzhou Hunan 412008，China）

Abstract：On the basis of D-ADL formal specification，the principles of behavior refinement are formally defined as follows , the behavior refinement of components should satisfy the relation of observation weak simulation and the behavior refinement of connectors should satisfy the relation of branching weak simulation during the process.

Key words：software architecture；refinement；π calculus

在面向动态演化的软件模型SASM[1，2]中，元层RSAS居于核心地位。RSAS源于设计过程中的软件体系结构，是一个由多层抽象程度不同的体系结构描述形成的体系结构空间，层间存在求精关系，不同层次体系结构信息应该保持一致，高层次体系结构是低层次体系结构的抽象、约束，低层次体系结构是高层次体系结构的具体化和实现。因此，严格地给出体系结构求精准则成为指导RSAS的构造和检测RSAS层间一致性的关键所在。鉴于体系结构研究的一个核心在于行为描述和分析，因此，行为求精是研究体系结构求精的重点。

1 D-ADL语言简介

为刻画运行时体系结构，文献[3]中提出了基于高阶多型π演算的动态体系结构描述语言D-ADL。一方面，D-ADL遵循Darwin[4]、Wright[5]等给出的、已被广泛认同的体系结构描述框架，围绕体系结构实体（如构件、连接件和配置等）进行体系结构建模，如图1；另一方面，D-ADL把高阶多型π演算[6]作为行为语义基础，凭借高阶多型π演算，描述动态系统的特征，D-ADL允许构件、连接件和配置产生变更，并使得对体系结构模型的自动化分析、仿真、检查成为可能。

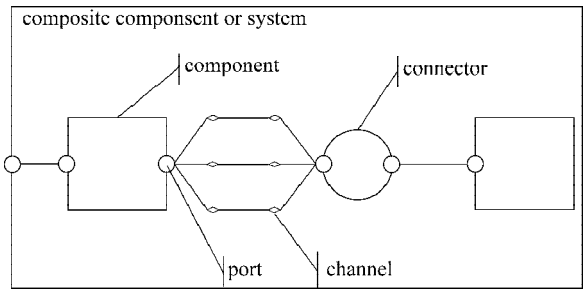

图1 D-ADL描述框架

Fig. 1 The description framework of D-ADL

收稿日期：2007-08-04

基金项目：国家自然科学基金资助项目（60773110），湖南省教育厅优秀青年科研基金资助项目（06B023），湖南工业大学成人教育研究基金资助课题（05B06）

作者简介：李长云（1971-），男，湖南耒阳人，湖南工业大学教授，博士，主要研究方向为软件体系结构和软件自动化.

D-ADL 拥有丰富的数据类型，这些数据类型主要分为两大类别：常规类型（OrdinaryType）和体系类型（ArchitectureType）。常规类型主要容纳一般意义的数，体系类型用于描述体系结构元素。体系类型中最重要的是进程类型（ProcessType），表示体系结构中的各种行为和动作序列。构件计算和连接件路由行为都是进程，例如，构件计算被定义为：

computation ::= *OrdinaryType variant assignment*. computation | prefix. computation |
if *boolean* **then** { computation$_1$ } [01 **else** { computation$_2$ }] |
choose { computation1, ⋯ ,computation$_{n-1}$ **or** computation$_n$ } |
unobservable. computation | **inaction** | **replicate**

prefix ::= **via** *portname^channelname* **send** *OrdinaryTypeValue* |
via *portname^channelname* **receive** *OrdinaryTypeVariable*

在高阶多型π演算中，Abstraction 是一种特殊的进程，是一般进程的参数化抽象。高阶多型π演算中的 Abstraction 概念被 D-ADL 所直接使用。在 D-ADL 中，构件、连接件和体系结构风格都被模型化为高阶π演算中的 abstraction 类型。

动态演化可分为两种类型：非预设的和预设的。非预设意指在原始设计中没有预料到的新需求，直到系统投入运行以后才发现的、必须对系统配置进行修改和调整的情况。D-ADL 本身并不直接支持非预设演化。预设的演化意指软件的动态改变都是软件设计者能够预先计划和设想的，可实现为系统的固有功能，成为应用的一部分。因此，动态体系结构语言必须提供某种机制来表示预设的演化行为。在 D-ADL 中，D-ADL 把这种演化行为从计算行为中分离出来，显式、集中地表达。D-ADL 中预设的演化行为规约为：

choreographer ::= attach.choreographer|detach.choreographer|create.choreographer |
destroy.choreographer | **inaction** | **replicate**

attach ::= **attach** *ComponenInstanceName^PortName* **to** *ConnectorInstanceName^EportName*|
attach *ComponenInstanceName^PortName^ChannelName*
to *ConnectoInstanceName^EportName^EchannelName*

detach ::= **detach** *ComponenInstanceName^PortName* **from** *ConnectorInstanceName^EportName* |
detach *ComponenInstanceName^PortName^ChannelName*
from *ConnectoInstanceName^EportName^EchannelName*

create ::= **new** [01 *ComponentInstancName*:]*ComponentName*([0+ *Actual-parameter*])|
new [01 *ConnectorInstancName*:]*ConnectorName*([0+ *Actual-parameter*])] |
new [01 *PortName*:]*PortType name*|
new [01 *PortName^ChannelName*:] *ChannelType name*

destroy::=**delete** *ComponentInstanceName*|**delete** *ConnectorInstanceName*|
delete *PortName*| **delete** *PortName^ChannelName*

attach 起建立新连接的作用，detach 起解除连接的作用。create 使用 new 关键字，动态创建新的构件实例和连接件实例，以及新的端口和通道；destroy 使用 delete 关键字，动态删除构件实例和连接件实例，以及端口和通道。

2 行为求精规则

体系结构求精包含以下 3 个方面：1）结构求精；2）行为求精；3）属性求精。结构求精是指在结构方面对求精对象进行细化；行为求精意指对求精对象的行为进行求精，详细化内部行为过程，添加行为活动；属性求精意指在求精对象的约束基础上添加新的属性。一般地，实际的求精是 3 个方面的组合。本文仅讨论行为求精。

2.1 基本思想

在 D-ADL 中，构件（连接件）的行为基于π演算理论，其中单一构件（单一连接件）的行为规格被 D-ADL 直接描述，复合构件（复合连接件）的行为来

自其成员行为的并发组合。如果可从前者求精成后者，则其行为之间存在着什么样的关系和限制，这就是行为求精研究的核心。

π演算理论研究核心是行为等价理论，即，如何判定2个并发执行并与环境交互的系统是否具有相等行为（Equivalent Behavior）。如果2个系统行为相等，那么，从任何环境中取出其中1个而放入另外1个，环境不会感知替换发生。π演算提供两种行为等价理论：强等价关系和弱等价关系。

π演算的行为分为两类：一类是不可见内部动作 τ；另一类是可见的外部行为。外部行为描述进程与环境交互的能力。强等价关系既考虑系统的内部行为，又考虑系统的外部行为。弱等价关系是一种不考虑系统的内部行为的行为等价理论，它判定2个具有不同内部结构因而具有不同内部行为的系统是否等价。应用π演算时，感兴趣的总是系统的外部行为。因此，弱等价关系具有更重要的实际意义。

从外部观察者角度来看，求精对象和求精结果的行为应该保持一致。但是，行为求精关系并不是严格的弱等价关系。在 π 演算中，行为等价理论的基本技术是互模拟（Bisimulation），弱等价关系就是弱互模拟。而环境对求精对象的行为期望应该被求精结果所维持，但求精结果可能表现出一些新的行为。换句话说，求精结果可以弱模拟求精对象的行为，但求精对象无需弱模拟求精结果的行为。这种弱模拟关系在构件行为求精和连接件行为求精上又有不同的具体体现，下面分别讨论。

2.2　构件行为求精

对构件的行为求精，着眼点在于外部观察行为应该前后保持一致。作为一个构件，其所能被外部观察到的行为是发生在接口处的交互活动。因此，我们认为单一构件计算规格中只有定义在端口（或通道）处的接收（receive）或发送（send）数据活动才是可以被观察到的，其它活动都被视作内部活动 τ。在复合构件中，只有发生在外部端口（或外部通道）处的交互活动才能够被外部观察到。当然，外部端口（或外部通道）如果通过Configuration段的rely关键字被映射到内部构件的端口（或通道），则这个内部端口（或内部通道）上发生的交互活动也能被外部观察到。因为映射关系相当于把内部端口（或内部通道）暴露为外部端口（或外部通道）了。

在D-ADL中，单一构件的行为专注于业务计算功能，而复合构件的行为分为两种：一是表示其处理业务逻辑的计算功能，这种行为来自其成员行为的并发组合；一是用于预定义的演化行为。从单一构件求精到复合构件，表现在行为方面的关联就是复合构件的计算功能对单一构件的计算功能的观察弱模拟。

在π演算中，观察弱模拟意指对不可观察的活动序列进行抽象，并跟踪每一个可观察的活动。要严格定义π演算进程间的观察弱模拟关系，需先定义π演算进程间的经历关系。

定义 1（经历关系（Experiment Relations））

对任一可观察活动 α，→表示进程间的迁移关系，定义关系$\Rightarrow$和关系$\overset{\alpha}{\Rightarrow}$：

1）$P \Rightarrow Q$表示存在0个或多个约简序列$P \rightarrow \cdots \rightarrow Q$，即为→的传递自反闭包；

2）设 $\alpha = \alpha_1 \cdots \alpha_n$，那么，$P \overset{\alpha}{\Rightarrow} Q$表示$P \Rightarrow \xrightarrow{\alpha_1} P_1 \cdots \Rightarrow \xrightarrow{\alpha_n} P_n \Rightarrow Q$。

定义 2（观察弱模拟（Observation Weak Simulation））

设某进程集合上的二元关系 ε，对所有 $P \varepsilon Q$，如果有 $P \overset{\alpha}{\Rightarrow} P'$，则存在 Q'，并且 $Q \overset{\alpha}{\Rightarrow} Q'$ 和 $P' \varepsilon Q'$ 成立，则称 ε 为观察弱模拟。

构件求精表现在行为方面的关联就是行为之间的观察弱模拟关系。

规则 1　　设进程 P_A 表示构件 A 的计算行为，进程 P_B 表示构件 B 的计算行为，若构件 A 是对构件 B 的求精，则 $P_A \varepsilon P_B$ 成立，即 P_A 观察弱模拟 P_B。

2.3　连接件行为求精

分支弱模拟概念是分支弱等价关系的基础。分支弱模拟意指抽象不可观察的活动序列，在跟踪每一个可观察的活动的同时，尽可能保持进程的所有分支轨迹。进程间的分支弱模拟定义如下。

定义 3（分支弱模拟（Branching Weak Simulation））

设某进程集合上的二元关系 γ，对所有 $P \gamma Q$ 且 $P \xrightarrow{\alpha} P'$，则有：$(((\alpha = \tau) \wedge (P' \gamma Q)) \vee (\exists Q', Q'' : ((Q \Rightarrow Q'' \xrightarrow{\alpha} Q') \wedge (P \gamma Q'') \wedge (P' \gamma Q'))))$，则称 γ 为分支弱模拟。

连接件虽然可视作一种特殊的构件，但与一般意义的构件不同的是：构件专注于计算，连接件旨在建立构件间的交互以及支配这些交互规则，而交互规则的核心体现为路由行为。因此，构件求精过程需要维持外部观察行为的一致，表现为求精结果对求精对象的观察弱模拟。而连接件求精过程应该保持路由行为的一致，具体来说，就是求精结果能够分支弱模拟求精对象的路由行为。

规则 2　　设进程 P_C 表示连接件 C 的行为，进程 P_D 表示连接件 D 的行为，若连接件 C 是对连接件 D 的求精，则 $P_C \gamma P_D$ 成立，即 P_C 分支弱模拟 P_D。

和构件求精相同，只有发生在接口处的交互活动才可以被观察，其它的活动视作内部活动 τ，对外不可见。复合连接件中的某些内部端口（或内部通道）被映射到外部端口（或外部通道），相当于把内部端口（或内部通道）暴露为外部端口（或外部通道），则这

个内部端口（或内部通道）是可见的。

关于观察弱模拟和分支弱模拟的关系有着如下的定理。

定理 1 如果 Q 分支弱模拟 P，则 Q 也观察弱模拟 P，即，如果 $P \leqslant_b Q$ 成立，则 $P \leqslant_o Q$。

证明 设 P 和 Q 之间存在关系 ε，因为 Q 分支弱模拟 P，则 $\varepsilon \in \leqslant_b$。根据定义 3，对任意 $\alpha \in \mathrm{Act}$，如果 $P \xrightarrow{\alpha} P'$，当 $\alpha = \tau$ 时，$P' \varepsilon q$；当 $\alpha \neq \tau$ 时，$(\exists Q', Q'' : ((Q \Rightarrow Q'' \xrightarrow{\alpha} Q') \wedge (P \varepsilon Q'') \wedge P' \varepsilon Q')))$，而 $(Q \Rightarrow Q'' \xrightarrow{\alpha} Q')$ 可表示为 $Q \overset{\alpha}{\Rightarrow} Q'$，同时 $(P' \varepsilon Q')$ 成立。定义 2 的条件满足。因此，ε 也是观察弱模拟关系，即 $\varepsilon \in \leqslant_o$。

可见，对连接件行为求精的限制要比构件行为求精强烈。

3 结语

体系结构求精的本质在于，不同层次的体系结构信息应该保持一致。本文给出了行为求精的形式规则，从而不仅使得求精前后的接口保持兼容，而且能够维持求精前后的外部行为的一致性，这为软件演化过程中应用的完整性和一致性提供了理论基础。

参考文献：

[1] 李长云，李 莹，吴 健，等. 一个面向服务的支持动态演化的软件模型[J]. 计算机学报，2006，29（7）：1020-1028.

[2] 李长云. 基于体系结构的软件动态演化研究[D]. 杭州：浙江大学计算机学院，2005.

[3] 李长云，李赣生，何频捷. 一种形式化的动态体系结构描述语言[J]. 软件学报，2006，17（6）：1349-1359.

[4] Magee J，Kramer J，Giannakopoulou D. Behaviour analysis of software architectures[C]// Proc. of 1st Working IFIP Conf. on Software Architecture. Boston：Kluwer Academic Publishers，1999：35-50.

[5] Allen R. A Formal Approach to Software Architectures [D]. Pennsylvania：Carnegie Mellon University，1997.

[6] Sangiorgi D. Expressing Mobility in Process Algebras：First-Order and Higher-Order Paradigms[D]. Edinburgh：University of Edinburgh，1992.

（上接第 20 页） 盐类结晶侵蚀等因素的影响，所以笔者认为桩基础仍然能继续正常工作。

4 结论

综上所述，长期浸没于地下水位以下的方桩，由于土壤和地下水中氧气和二氧化碳浓度较低，没有发生碳化，钢筋锈蚀轻微，且在湿养环境下继续水化，强度有所提高。桩体混凝土中氯离子浓度分布与土壤的渗透性有直接关系，土壤的渗透性越弱，受到的氯离子侵蚀程度就越轻，其主要体现为减小 C_s 值。氯离子分布曲线中的上升段为硫酸盐所造成，土层的弱渗透性同样也可以减轻硫酸根对混凝土侵蚀。由此可见，长期处于弱渗透性土壤环境中且位于地下水位以下的钢筋混凝土，其受到的侵蚀会相对有所缓解。

可以认为，本文所检测的桩基础是处于比较有利的环境中的，其特点是长期位于地下水位以下和地下水中高浓度的氯离子。前者隔离了氧气，保护了钢筋，后者抑制了硫酸盐的侵蚀，保护了混凝土；同时，干湿、盐类结晶等问题也不会发生。土壤渗透性若是比较弱，对结构的保护作用就更好，但如果钢筋混凝土结构并非是长期浸没于水下的，则问题会严重得多，硫酸盐侵蚀和盐类结晶侵蚀会对混凝土造成严重影响[8]。

参考文献：

[1] 金伟良，赵羽习. 混凝土结构耐久性[M]. 北京：科学出版社，2002.

[2] 李佩珍，谢慧才. RCT——快速氯离子检测方法及其应用[J]. 混凝土，2000（12）：46-48.

[3] Ervin Poulsen. Chloride profiles： Analysis and interpretation of observations[R]. Denmark：AEC Laboratory，2004.

[4] 田俊峰，潘德强，赵尚传. 海工高性能混凝土抗氯离子侵蚀耐久寿命预测[J]. 中国港湾建设，2002（2）：1-6.

[5] 李学礼. 水文地球化学[M]. 北京：原子能出版社，1988.

[6] 俞震豫. 浙江土壤[M]. 杭州：浙江科学技术出版社，1994.

[7] 王绍东，黄煜镔，王 智. 水泥组分对混凝土固化氯离子能力的影响[J]. 硅酸盐学报，2000，28（6）：570-574.

[8] 马孝轩，仇新刚，孙秀武. 钢筋混凝土桩在沿海地区腐蚀规律试验研究[J]. 混凝土与水泥制品，2002（1）：23-24.