构件可测试性标准

本节描述了构件的运行时可测试标准[1]。

我们知道运行时测试会以意想不到的方式干扰系统状态或资源可用性，因为系统的生产状态和数据将与测试混合。更糟糕的是，测试操作可能触发系统边界之外的事件，可能以难以控制或无法恢复的关键方式影响系统环境，例如在测试部分战斗系统时发射导弹。

运行时测试中存在干扰的事实需要一个指示器来指示系统相对于运行时测试的弹性有多大，或者换句话说，运行系统上的测试可能导致的不利影响。 IEEE的可测试性标准定义可以改写为反映这些要求，如下：

运行时可测试性定义：运行时可测试性是（1）系统或组件在不受到广泛影响的情况下促进运行时测试的程度; （2）允许在运行时执行哪些测试而不会对运行系统产生广泛影响的规范。

下面我们将介绍影响运行时可测试性的主要因素。

测试灵敏度（Test Sensitivity），它表征了作为测试的一部分执行的哪些操作以不可接受的方式干扰正在运行的系统或其环境的状态。在本节中，我们将描述影响组件测试灵敏度的四个主要因素：具有内部状态的组件，组件的内部/外部交互，资源限制和系统可用性。

组件状态（Component State）。知道组件是否具有某种外部状态（即输入的结果不仅取决于输入本身的值，而且还取决于过去输入的值）是测试灵敏度的重要因素。在传统的“离线”测试中，这很重要，因为调用顺序会对测试的预期结果产生影响。在运行时测试的情况下，了解组件是否具有状态非常重要，原因有两个。首先，因为如果处理不当，运行时测试的结果将受到系统状态的影响，其次，因为系统的状态可能会随着测试调用的结果而改变。

组件交互（Component Interactions）。在许多情况下，组件将使用系统中的其他组件，或与系统边界之外的外部参与者交互。这些交互可能启动其他交互等。这意味着组件的运行时可测试性取决于它在测试期间与之交互的组件的运行时可测试性。

所有这些交互都可能通过改变协作中任何组件的状态来干扰正在运行的系统的状态。在某些情况下，这些交互中的一些将跨越系统的边界并影响其他系统的状态，这可能难以预防和修复。在最坏的情况下，相互作用将通过发送一些能够实现无法撤消的物理输出的输出来达到“外部世界”，例如发射导弹。

资源限制（Resource Limitations）。之前的两个灵敏度因素主要影响系统的功能要求。但是，运行时测试也可能会影响非功能性需求。由于运行时测试将在正在运行的系统上执行，因此这些测试的负载将被添加到由系统正常运行引起的负载中。在某些情况下，它将超过系统的可用资源，例如处理器或内存使用，时序限制，超出功耗限制。

可用性（Availability）。要进行测试的系统的可用性要求也是一个因素。存在两种可能性：如果组件仅用于测试目的（独占使用），或者用于测试和正常服务（共享使用）。在共享配置中，可以做出两个区别：阻塞和非阻塞。第一个意味着在执行测试时生产操作将被阻止或拒绝，从而削弱组件提供的服务的可用性。如果组件具有高可用性要求，则无法在此情况下执行运行时测试。在第二种情况下，测试调用可以与生产调用交错，并且组件能够区分测试和生产请求。

运行时可测试性度量。

最终，阻碍运行时测试的所有测试灵敏度因素将阻止测试工程师评估某些特性或需求，否则这些特性或需求可在无限资源的理想条件下执行并完全控制运行系统。这是测量标准使用的主要思想，以获得系统的运行时可测性测量（RTM）的数值测量。

设M\*是我们想要测试的所有那些特征或需求的数量，并且Mr是相同的测量，但是减少到可以在运行时测试的特征或需求的实际数量，即Mr ≤ M\*。系统的运行时可测性测量（RTM，Runtime Testability Measurement）定义为M \*和Mr之间的商

更具体的，我们需要量化Mr和M\*。

本测量标准引入组件交互图（CIG，Component Interaction Graph）作为量化基础，即组件的接口调用图。CIG被定义为有向图CIG =（V，E）。顶点集V =Vp U Vr，由提供的和所需顶点的集合形成，其中每个顶点表示某个组件的接口的方法。E中的边是从顶点创建的，这些顶点对应于用于组件间依赖性的接口所提供的顶点的所需接口，和从提供到用于组件内依赖性的所需接口。即Vp代表组件提供的接口，Vr代表组件需要（或依赖）的接口，E代表接口间的依赖关系。

其中，每个顶点都用测试信息进行注释，这意味着在执行运行时测试时是否可以遍历这样的顶点，如下所示：

(2)

其中，如果该接口可以被运行时调用，也即是该接口运行时测试不会对系统产生影响，标记为0，否则标记为∞。

边信息可以通过对组件源代码的静态分析，或通过提供某种模型来获得，例如状态或序列图。对于某些顶点无法进行信息注释的情况，应采取保守的方法，为其分配无限权重∞。

覆盖标准（Coverage Criteria）。我们将根据现有的准则中提出的两个充分性标准来衡量系统的运行可测性：

1）全顶点充分性标准要求在组件的所有提供的和所需的接口中执行每个方法，这转换为遍历我们模型的顶点vi∈V，至少一次。

2）全上下文依赖标准要求在每个可能的上下文之间测试顶点的调用。如果存在来自vi的到达vj的调用序列，则顶点vj是依赖于vi的上下文。对于这些依赖性中的每一个，所有可能的路径（vi，vi + 1，...，vj）被认为是可行的，并且需要进行测试。

RTM的值(Value of RTM)。

我们将假设交互开始于我们想要覆盖的顶点（对于所有顶点覆盖）或路径的第一个顶点（对于所有上下文依赖性覆盖）。因为CIG中的边表示可能发生或不发生的交互（没有任何控制流信息），我们不能假设在尝试覆盖路径时，只会遍历路径中的顶点。在最坏的情况下，交互可以通过从交互开始的顶点可到达的所有顶点传播。

因此，为了估计覆盖从顶点vi开始的顶点或vi上下文相关路径的最坏情况惩罚，计算必须考虑从vi可到达的所有顶点，我们将其表示为Pvi。

对于我们想要覆盖的每个顶点vi或路径（vi，vj，vk，...），我们计算一个损失值T（vi），类似于单个顶点的损失值：

通过将那些特征为T（vi）≠∞的特征考虑为可测试，可以分别为所有顶点和全上下文依赖覆盖重写等式(1)，

[1] González A, Piel É, Gross H G. A model for the measurement of the runtime testability of component-based systems[C]//2009 International Conference on Software Testing, Verification, and Validation Workshops. IEEE, 2009: 19-28.