Web 服务组合失效检测框架①

Failure Detection Framework for Web Service Composition

陈丽金 周 娅 (桂林电子科技大学 计算机与控制学院 广西 桂林 541004)

摘 要： 随着 Web 服务组合的不断发展，失效检测器成为构建可靠的 Web 服务组合环境所必需的基础组件之一。对基于 Web 服务组合的失效检测机制进行研究，设计了主从式失效检测算法并对其进行理论分析，构造 Web 服务组合失效检测框架，为 Web 服务组合监测提供有效的手段，提高系统可用性。

关键词： 失效检测器 Web 服务组合 失效检测算法

通过 Web 服务组合动态生成新的应用系统，以满足企业的动态需求，已成为 Web 服务技术不断向前发展的技术动力和研究热点。目前，Web 服务研究仍有许多关键问题尚待解决。在许多关键领域，软件的设计者和开发者需要考虑系统失效检测的问题。对于许多关键性业务领域来说，Web 服务的可靠性和可用性问题严重妨碍了其在更多领域内的应用。其中，Web 服务的可用性是非常重要的一个方面，失效检测是增强 Web 服务可用性的核心技术。对于推动 Web 服务的应用而言，Web 服务失效具有重要意义[1]。

1 相关工作研究

文献[2]将检测服务结合到面向服务架构 QoS 管理中，通过监测服务反馈的信息，诊断服务能够检测状态的变化，并利用一种基于图模型的方法对状态变化原因进行推断。文献[3]提出一种基于移动代理的流程故障处理方法，该方法将移动代理分布到系统中，由这些代理监控流程运行的状态，并在必要的时候由这些代理进行追踪和错误处理。文献[4]提出容错 SOA 的概念，利用 SOAP 的优点提出 FT-SOAP 框架，利用 Web 服务复制和日志的方法来实现故障恢复机制，其失效检测机制通过简单的超时机制来判断失效。文献[5]提出了一种基于反射技术的 Web 服务失效处理方法，通过在 Web 服务客户方和服务器方配置反射层，检测造成服务失效的各种状态，以提高 Web 服务的适应性和健壮性。文献[6]提出了能够适应网络状况变化的自适应的失效检测算法，可以实现较小的检测时间，但是不能有效减少由于丢包产生的误判，错误率会比较高。

2 失效检测器的基本概念

失效检测(failure detection)是 Web 服务组合系统可用性保障的基本技术，它对运行时系统的存活状态进行检测。提供失效检测功能的组件称为失效检测器[7](failure detector,简称 FD)，失效检测器是失效恢复、动态重启、可靠性通信、集群管理等功能的基础。在 Web 服务组合中需要提供良好的失效检测机制来及时检测服务器以及其中服务组件和应用组件的失效情况，从而提高系统的可靠性和可用性。

2.1 失效检测的基本模型

push 模型也称作心跳策略：由被检测对象按照一定的时间时隔定期地向失效检测器发送心跳信息，通告它们依然存活。如果失效检测器超过某一期限没有收到心跳消息，失效检测器则认为其失效。该模式被检测对象向失效检测器“推”(push)失效事件，我们也称又为“推”模式，其消息交换如图 1 所示。

pull 模型(也称作“ping”策略)：失效检测器定时发送查询消息检查被检测对象的状态，被检测对象收到查询消息后返回应答消息，如果失效检测器超过一定时间间隔没有收到应答消息，则意味着被检测对

① 基金项目:广西青年科学基金(桂科青 0832101)
收稿时间:2009-02-19

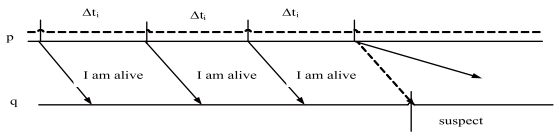

图 1　push 模型的监控消息

象失效。该模式失效检测器从被检测对象中“ 拉 ”(pull)失效事件，我们又称之为 “ 拉 ” 模式，其消息交换如图 2 所示。

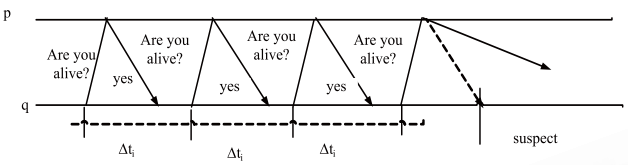

图 2　pull 模型的监控消息

两种检测模型各有优劣。pull 模型需要失效检测器发送查询消息，并等待被检测对象返回应答，这样导致失效检测时间较长；同时需要两倍数量的消息数目。然而，pull 模型是一种主动检测方式，可以只在需要的时候才发起检测，而且可以保证每个检测消息及其延时对检测结果的影响是独立的。

2.2 主从式失效检测器的设计模型

本失效检测系统采用了主从式失效检测机制确保失效检测的可靠性，正常状态下检测系统运行的是主检测器，当主检测器检测到系统某资源失效时启动从失效检测器进行确认，只有主从检测器均返回失效结果时，才认为该资源失效。失效检测的系统开销主要是由主检测器消耗的，因此在设计失效检测器时应该尽量使主检测器的开销最小，所以主失效检测器使用 push 模型，从失效检测器使用 pull 模型。

3　失效检测算法

3.1 算法基本思想

每个被检测进程 p 周期性地向失效检测进程 q 发送不同序号的心跳消息，心跳消息的序号总是递增的。失效检测器对收到的心跳消息的历史纪录做统计、分析，并计算出符合确认度下一个心跳消息到达的时间上限；如果超时未收到心跳消息，根据确认度要求失效检测器主动发送消息询问被检测进程是否失效。失效检测器算法主要步骤如下:

```
For process p:
For all i>0,at time△ti* i,
  send mpi to q ,
    i=i+1;
If ( upon receive mqi from q do)
  send mai to q   endif;
For process q:
  f = ?1 //freshness point
  upon receive mpi from p at time Tcurrent do;
  if f==-1 then f=Tcurrent;
  else △T= Tcurrent - f, f =Tcurrent,
  append △T to W endif;
  if (△T >E(TD) )
    {execute TS_transition,
    send mqi to p,
    if(upon receive mai from p and Twait<Tado)
      execute ST_transiton,
    else execute SF_transition   end if;
    }
```

以上算法中，主失效检测器由进程 p 每间隔 ti 周期性地向进程 q 发送消息 mpi(i 为检测消息的序号)；进程 q 负责接收消息，进程 q 收到消息之后，首先将消息 mpi 的接收消息时间($T_{current}$- f)保存到滑动窗口 W 中。当进程 q 超过 E(TD)没有收到进程 p 的发送消息 mp_i 时，即认为该系统资源失效，则启动从失效检测器进行确认。从失效检测器主要由进程 q 发送查询信息 mq_i 给进程 p，p 在收到该消息后发回应答消息 ma_i，以表明自己处于正常状态，如果进程 q 没有收到应答消息 ma_i 则认为进程 p 失效。其中，T 表示进程处于正常状态，S 表示进程处于失效状态，F 表示进程处于失效状态，则 TS-transition 表示检测器的输出由 T 变为 S，ST-transition 表示检测器的输出由 S 变为 T，SF-transition 表示检测器的输出由 S 变为 F。

3.2 计算阈值 T_D^U

T_D 指失效进程 p 从失效发生时间开始，到进程 q

上的失效检测器开始怀疑 p 的时间，这一指标保障失效检测器输出的完整性要求[7]。T_D^U 是一个失效检测器与被检测对象之间的服务质量的基本评价指标之一，T_D^U 是检测时间 T_D 的上界，限定了检测速度，保证了失效检测的完整性级别，是一个十分重要的评价指标。

图 3 介绍了主从式失效检测器的执行过程。进程 p 定时向进程 q 发送心跳消息，进程 q 分别在 T_0、T_1、T_2 时刻收到前 3 个心跳消息，并预测下一个心跳消息将在 T3 时刻到达。而实际上进程 p 在 T_6 时刻失效，对于进程 q 的时钟而言是 T_5 时刻，则进程 q 在 T_3 时刻未收到应该到达的心跳消息，开始怀疑进程 p。则实际检测时间为：T_3-T_5，即 T_3-T_4+ η，η 为心跳消息的时间延迟。通常通常令 t=T_3-T_4，T_D^U = t+E(η)，E(η)为平均心跳消息延迟。进程 q 超过 T_D^U 时间没有收到进程 p 的心跳消息，则主动发送问询消息，查看进程 p 是否失效。如果进程 q 在发送问询消息后，超过 T_7-T_3+ T_D^U 时间没有收到应答消息，则确认进程 p 失效。

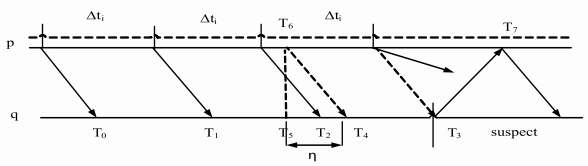

图 3　失效检测执行过程

3.3 算法证明

在 Web 服务组合失效检测中，失效检测器为应用程序提供失效检测等价于◇P[7]类的失效检测器，即失效检测器需达到强完整性和强准确性级别。通过以下分析说明本失效检测器等价于◇P 类失效检测器。

(1)强完整性。在任意一次运行中，每一个发生失效的进程最终都会被所有正确的进程永远判定为失效。

证明：假设在一个正确的进程 q 检测进程 p。需要证明：如果 p 发生失效，那么存在某一时刻对于任一的 T>t，失效检测的结果都将判定 p 失效。在 p 失效之后，将不会发出任何消息，也就是说，进程 q 此后将收不到任何消息，因此，E(TD)将不会发生变化。假设 mp_i 是 q 在时刻 tlast 从 p 收到的最后一个应答消息，则 i 值将永远一个常数。此后，在任何查询时刻 $t_{current}$>t_{last}，都有△T=$T_{current}$ − f，随着 $t_{current}$ 的增长，△T 将严格单调递增。因此，对于应用程序 A，在任意时刻 tcurrent>tlast 查询，都会将 p 判定为失效进程。

(2)强准确性。在任意一次运行中，在某个时刻 t 之后，每个正确的进程都不会被错误认为发生失效。

证明：同上面的证明进行同样的假设，需要证明：在某个时刻 t 之后，应用程序 A 将不会错误地认为 p 发生失效。因为此属性具有最终(eventual)性。假设 mpi 是 p 发出的第 i 个消息，发送时刻为 ts，进程处理消息时间和消息的传递时间都有上界，分别记为 tp 和 tt。因此，对于任意检测消息 mpi (j≥i)，其检测时间≤tp+tt。那么，对于任意的查询时刻 tcurrent>ts，△T 存在一个上界。只要 T_D^U > $T_D^{'}$，那么对于任意的查询时刻 $t_{current}$> t_s +t_p+t_t，A 都不会怀疑进程 p。

4　Web服务组合失效检测框架

基于以上失效检测的原理，根据 Web 服务组合的失效检测需求，本文设计了一个失效检测框架，框架的结构如图 4 所示。在整个架构由服务流程监测器(Service Process Monitor,SPM)、策略管理器(Policy Manager,PM)和故障调节器(Fault Adjuster,FA)构成，其中，失效检测流程监控处理是一个相当重要的环节。

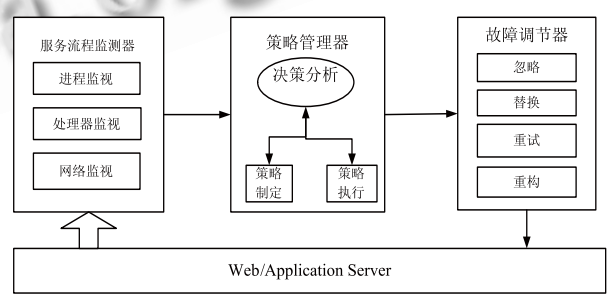

图 4　失效检测框架结构

服务流程监测器(SPM)主要负责监测与 Web 服务组合运行相关的进程、处理器和网络状态，根据主从式失效检测机制动态掌握被监测组件的存活状态。

策略管理器(PM)主要由决策分析、策略制定和策

略执行等模块组成。决策分析模块：它是应用决策分析理论，分析服务流程监测器的结果来制定错误处理策略。策略制定模块：实现由决策分析模块制定的错误处理策略，按照策略描述语言(PDL)的格式规范化该策略。策略执行模块：调用故障调节器的相关接口实现错误处理。

故障调节器(FA)主要由忽略、重试、替换和重构操作组成，根据策略管理器的决策选择相应的恢复策略进行处理。

5 结语

随着 Web 服务组合的不断发展，针对提高 Web 服务组合可用性问题，构造了 Web 服务组合失效检测框架。本失效检测框架是借鉴公布式系统不可靠错误检测器研究的思想，采用主从式失效检测机制设计失效检测算法，适应复杂的网络生态环境，增强了系统的可用性，为流程执行时业务功能与服务质量的保障提供了一种有效的机制。

参考文献

1 付晓东,邹平.一种规则驱动的 Web 服务组合例外处理方法.计算机应用, 2007,27(8):1984 - 1990.

2 Wang GJ, Wang CZ, Chen A, et al.Service level management using QoS monitoring, diagnostics, and adaptation for networked enterprise systems. Proceedings of the 9th IEEE International EDOC Enterprise Computing Conference. Washington DC: IEEE Computer Society, 2005.239 - 250.

3 Cao JN, Yang J, Chanw T. Exception handling in distributed workflow systems using mobile Agents. Proc. of the IEEE International Conference on e-Business Engineering. Washington: IEEE Computer Society, 2005.48 - 55.

4 Liang D, Fang C, Chen C, Lin F. Fault-tolerant Web service. Proc. of the 10th Asia-Pacific Software Engineering Conference(APSEC'03), ChiangMai, hai-land, 2003.310 - 319.

5 徐新卫,周良,丁秋林.Web 服务失效处理的反射中间件技术应用与实现, 2007,29(8):1371 - 1376.

6 Chen W, Toueg S, Aguilera MK. On the quality of service of failure detectors. IEEE Transactions on Computers, 2002,51(5):561 - 580.

7 董辉.失效检测器在拜占庭容错复制系统中的应用[硕士学位论