ПОДОБРЯВАНЕ НАДЕЖДНОСТТА НА ВРЪЗКИТЕ С БАЗИ ДАННИ В DeTC

Доц. д-р Асен Рахнев, ФМИ, ПУ „Паисий Хилендарски", Пловдив
Гл. ас. д-р Олга Рахнева, УХТ, Пловдив
Никола Вълчанов, ФМИ, ПУ „Паисий Хилендарски", Пловдив

IMPROVING DATABASE CONNECTIONS RELIABILITY IN DeTC

Assoc. Prof. Asen Rahnev, PhD, FMI, Plovdiv University, Plovdiv
Ch. Ass. Olga Rahneva, PhD, UFT, Plovdiv
Nikola Valchanov, FMI, Plovdiv University, Plovdiv

Abstract

This paper analyses connectivity problems in systems for electronic testing, and the causes for connectivity failures. The paper describes means to overcome connectivity failures, based on the experience of thousands of performed tests. Examined models are disconnected database access model, and implementation of an intermediate layer, responsible for connectivity and synchronization of local databases with server ones.

1. Въведение

Разпределеният клъстер за електронно тестване - DeTC (Distributed e-Testing Gluster) се разработва [1-4] като част от разпределения център за електроннно обучение – DeLC (Distributed e-Learning Center) [5] и представлява една отворена инфраструктура, която предлага услуги за електронно тестване, локализирани върху различни сървъри, с възможност за частична и автоматично контролирана интеграция (само в рамките на предварително дефинирани виртуални структури). Възлите на DeTC могат да бъдат различни университети, факултети, филиали, училища, институции и др., в които се предлага затворен цикъл на електронно тестване. DeTC се разработва като съвместен проект между Университета в Лимерик – Ирландия, катедрите Компютърни системи и Компютърни технологии на Пловдивския университет, катедра Информатика и статистика на Университета по хранителни технологии.

При DeTC данните могат да бъдат получавани от разнородни източници (web service, database server, COM object, server application (чрез remoting, TCP sockets и т.н.), текстов файл - XML, CSV или с друг специфичен формат и др.). Клиентските приложения, на които студентите полагат тестовете, осъществяват достъп до данните през модули, които имплементирайки общ интерфейс, позволяват еднакъв начин на работа, независимо от типа на източника. Това налага изследване на надеждността на връзките с базата данни.

В тази работа се изследват проблемите на свързаността в системи за електронно тестване и причините за загуба на свързаност. Разпределеният клъстер за електронно тестване DeTC се използва от няколко години за провеждане на изпити чрез тестове в Пловдивски Университет „Паисий Хилендарски", Университета по хранителни технологии, Европейския колеж по икономика и управление, Филиалите на Пловдивския Университет в Смолян и Кърджали, както и в други учебни заведения. В резултат на натрупания опит от проведените хиляди тестове, в работата се предлагат средства за преодоляване на загубата на свързаност в DeTC. Разглеждат се разкачен достъп до базите данни, създаването на междинен слой, отговарящ за свързаността и синхронизация на локалните данни със сървърните.

2. Проблеми на свързаността

Системите за електронно тестване обикновено са с централизирана база данни. Терминалите, на които студентите полагат изпитите се свързват със сървър и съхраняват там информацията за дадените отговори. Често срещана практика е приложения, работещи с отделен доставчик на данни да разчитат на постоянно отворени връзки към доставчици на данни. Този метод на работа крие проблеми с поддържането на връзките отворени.

При добре изградена и подържана компютърна мрежа свързаността на отделните станции е надеждна. В подобна среда поддържането на постоянно отворени връзки към източник на данни не е проблем. Но в случай на неизправна мрежа връзката между приложението-клиент и сървъра може да бъде загубена за кратък период от време. Тези ситуации могат да настъпят при проблем с хардуера, операционната система или дори софтуера на устройството, разпределящо трафика на информация в мрежата. В зависимост от източника на данни тези ситуации могат да доведат до срив на системата. В такива случаи се създава излишно напрежение, тъй като единственото решение е студента да положи на ново изпита.

В случаите, когато системата получава данните от сървърно приложение (като комуникацията се извършва през TCP socket) или СУБД (MS SQL Server, Oracle, DB2, mySQL, Firebird и т.н.) подобна опасност от срив е напълно реална.

Временната загуба на свързаност между две станции в една мрежа може да бъде причинена и от хардуерен и от софтуерен проблем. Някои от най-често срещаните причинители на подобни проблеми са:

a. Хардуер - мрежовата карта на даден компютър или слота на дънната платка, в която е поставена въпросната мрежова карта е в неизправност;

b. Кабел - мрежовият кабел е прекъснат или на определено място не дава добър контакт;

c. Софтуер - firewall, който следи пакетите предавани в мрежата и блокира тези, които смята за атака;

d. Операционната система - рестартирането на компютъра обявен за Master Browser в мрежа от компютри с ОС Windows, например може да доведе до временна липса на свързаност;

e. Устройството, което осигурява свързаността на компютрите - софтуерът на устройството в определена ситуация може да допусне грешка при работата си;

f. Спиране на електричеството.

При внезапно спиране на електричеството системата няма възможност да съхрани отговорите, дадени до момента. В такива случаи единственото решение е изпитът да бъде положен отново. В останалите случаи подобно прекъсване налага разпадане на връзката между connection-обектите на приложението-клиент и СУБД. [6] А именно сървърът на базата данни спира да подържа в pool-а си инстанция на връзката с клиентското приложение. Различните database server-и имат различна имплементация на проверката дали една връзка е активна или не. В общия случай тази проверка се извършва на определен интервал. Цикълът на живота на един connection обект, създаден в СУБД започва при отварянето на връзката между приложението-клиент и сървъра. Тогава този обект се създава и се използва за първи път, след което се подържа в паметта (connection pool) докато системата не реши, че той вече не е нужен. Това решение се взима посредством механизъм, който проверява дали СУБД все още може да достъпи приложението-клиент. При тази проверка СУБД изпраща на приложението-клиент Keep Alive пакет, с който проверява свързаността. Ако този пакет се върне – connection обект-а продължава да стои в паметта, в противен случай за определен период от време СУБД ще се опитва да достъпи приложението-клиент. Ако за този период приложението все още не е достъпно - паметта, заемана от connection обект-а се освобождава. Добрите СУБД имат възможност за настройка на интервала, през който се прави тази проверка или цялостно изключване на проверката. Когато проверката бъде изключена connection обектите се пазят в connection pool-а толкова време, колкото е посочени при отваряне на самата връзка.

Повечето разработени библиотеки за работа със СУБД разбират, че връзката се е разпаднала едва след като се направи опит да се достъпят данни от източника. В този случай през времето, когато е липсвала свързаност, сървърът е направил проверка и е установил, че приложението-клиент не е достъпно. В следствие на това е унищожило connection обекта. Когато библиотеката направи опит да достъпи базата данни през тази връзка – СУБД връща грешка. При добро администриране на СУБД този проблем може да бъде заобиколен, но системи, които разчитат на данни от различни източници трябва да предвидят механизъм за справяне с този проблем, за да са максимално независими и защитени от грешки.

При работа със sockets обаче при разпадане на връзката socket-обекта веднага разбира за загубата на връзка със сървъра.

Подобни ситуации могат да доведат до загуба на данни, а тя винаги е съпътствана с проблеми.

3. Разкачен достъп до бази данни

В DeTC (когато източникът на данни е СУБД) този проблем намира решението си, като се използва разкачен достъп до базата данни (при всяка заявка да се отваря нова връзка, която да се затваря след изпълнение).

Предимствата на този начин на работа са:

a. Данните се извличат веднъж от базата данни, след което се пазят във паметта. Това позволява тези данни да бъдат бързо достъпвани от

компоненти или изпращани на други приложения, без да се налага повторна заявка към базата данни;

b. Класовете, които съхраняват извлечените данни обикновено структурират информацията в таблици и предоставят описание на връзки между тях. Това позволява индивидуална работа с определена таблца или с навигиране из данните, следвайки връзки от тип parent-child;

c. Информацията може да бъде събрана от няколко места. Веднъж получена тя може да бъде манипулирана все едно е дошла от един източник (могат да се поставят ключове, да се правят връзки с други таблици и т.н.).

Този начин на работа по никакъв начин не забавя приложението, тъй като системите за управление на бази данни не унищожават обектите за връзка, създадени при свързване на приложението-клиент със сървъра. Тези обекти се пазят определено време след създаването им (connection pool), за да могат да бъдат преизползвани при повторно свързване на същото приложение с базата данни.

4. Междинен слой в DeTC, отговарящ за свързаността

За да може проблемите със свързаността да бъдат прихващани навреме и обработвани адекватно, в DeTC е предвиден междинен слой, намиращ се между модулите за достъп до данни и бизнес логиката. При това решение се добавя функционалност за тест на свързаността и при липса на свързаност модулът стартира процес по подновяване връзката. По този начин описания по-горе проблем може да се реши глобално за всички видове източници. Така ако разпадането на връзката е настъпило в момент, в който системата не може да продължи работа без извличането на данни от източника, то тя трябва да уведоми потребителя за липсата на свързаност и да го помоли да изчака докато сървъра бъде отново достъпен или да го прикани да преостанови работа и да потърси помощ за разрешаване на проблема. В противен случай системата може да продължи работа без да осведомява потребителя за проблема, като през това време продължава с опитите за подновяване на свързаността. По този начин ако връзката е била загубена за кратко системата ще спести излишни притеснения на изпитвания.

5. Синхронизация на локалните данни със сървърните

Проблемът, причинил загубата на свързаност между клиента и сървъра може да е лек и да бъде отстранен за минути, след което изпитът да продължи. Но винаги съществува риска той да е сериозен и да налага прекратяване на изпита. Решението, взето за превенция на такива ситуации в DeTC е да се запазва актуалното състояние на приложението, а именно системата да запаметява локално информацията, която е въведена, но все още не е обновена на сървъра. По този начин при загуба на връзка системата може да продължи работа, която не зависи от сървъра като съхранява временните данни локално, а при подновяването на свързаността информацията може да бъде синхронизирана.

Възможните варианти, предвидени за локално съхранение на информацията в DeTC са:

a. Инсталиране на локална инстанция на СУБД – данните, които не могат да бъдат обновени в отдалечения сървър се съхраняват локално. Когат свързаността бъде подновена двете СУБД биват синхронизирани;

b. Съхранение на данни и обекти (XML, двоична или друг специфичен формат) – при този начин на съхранение бизнес обектите на приложението могат да бъдат изцяло съхранени във файловата система, като при възстановяване на свързаността данните могат да бъдат възстановени и обновени на сървъра.

Съхраняването на временни данни в процеса на работа предпазва от загуба на данни не само в следствие от разпадане на връзката със сървъра, а и при токов удар или спиране на електричеството. Така подобни ситуации биха довели единствено до пауза на изпита. Когато условията за работа бъдат възстановени, студентът би имал възможност да продължи изпита от там, до където е бил стигнал, като системата може автоматично да преизчисли изгубеното време и да удължи времето за работа на студента.

6. Благодарности

Този проект е частично финансиран от Фонд „Научни изследвания“ към МОН по договор ВУ-МИ 107 / 2005

ЛИТЕРАТУРА

[1] O. Rahneva, *Testing and Assessment in Distributed Electronic Testing Cluster – DeTC*, 12th International Conference ELECTRONICS'03, Sozopol, Conference Proceedings, v. 4 (2003), pp. 214-219.

[2] O. Rahneva, *Generating Dynamic Questions in Distributed eTesting Cluster – DeTC*, ECEST'04, Bitola, v.1 (2004), pp 305-308.

[3] O. Rahneva, A. Rahnev, N. Pavlov, *Functional Workflow and Electronic Services In a Distributed Electronic Testing Cluster – DeTC* , Proceedings 2nd International Workshop on eServices and eLearning, Otto-von-Guericke Universitaet Magdeburd (2004), pp 147-157.

[4] О. Рахнева, *DeTC – Разпределен клъстер за електронно тестване*, Научно-практическа конференция "Новите технологии в образованието и професионалното обучение", София 16-17 май 2003, 84-91.

[5] Ст. Стоянов, И. Попчев, Р. Венков, Обща архитектура на системата за електронно обучение DeLC, Научно-практическа конференция "Новите технологии в образованието и професионалното обучение", София, 16-17 май 2003.

[6] MSDN – http://msdn2.microsoft.com