Rendszerintegráció és -felügyelet laboratórium (VIMIM309)

Megbízható üzenetküldés IBM WebSphere MQ alapon

Mérési segédlet

Készítette: Hegedüs Ábel

Utolsó módosítás: 2013. február 25.

Verzió: 1.2

Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

1 Bevezető

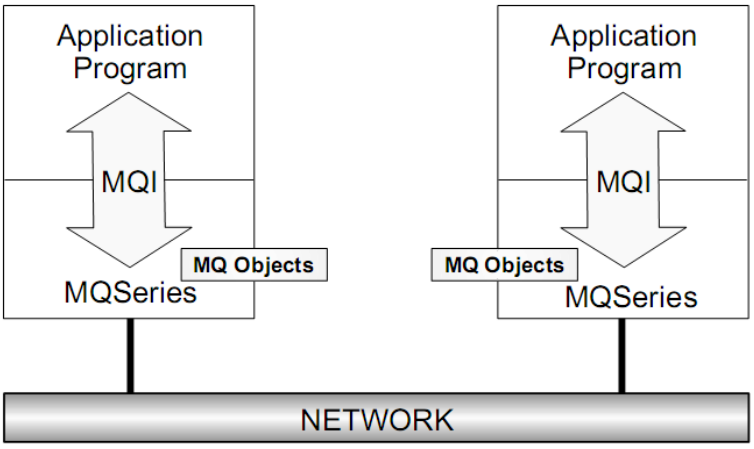
A labor során a méréseket végző hallgató a gyakorlatban is megismerkedik a rendszerintegráció és rendszerfelügyelet során használatos módszerekkel és eszközökkel. Végigköveti egy elosztott alkalmazás megvalósításának és felügyeletének legfontosabb lépéseit, ipari környezetben használt integrációs köztes réteg (middleware) technológiák és felügyeleti eszközök használatával. A mérések a következő témakörökhöz kapcsolódnak:

1. Munkafolyamatok megvalósítása Java nyelven
2. Megbízható üzenetküldés IBM WebSphere MQ alapon
3. Kommunikáció JMS és JMX technológia segítségével
4. OSGi szolgáltatások fejlesztése
5. Modell alapú eszközintegráció elosztott környezetben (SDE)
6. Felügyeleti adatok vizuális elemzése
7. Rendszerfelügyelet támogatás komplexesemény-feldolgozással

A jelen mérés során az üzenetsorok alkalmazásával kell munkafolyamatot megvalósítani, az IBM WebSphere MQ köztes réteg megbízható üzenetküldési szolgáltatásait felhasználva. A hallgatók az előző mérésen készített Java alapú munkafolyamatot alakítják üzenet alapú vezérléssűvé, ezáltal elsajátítva az üzenetsorokhoz kapcsolódó alapvető ismereteket.

2 A mérés célja

Az üzenetsorokat programok közötti kommunikációra használják. Az alkalmazáson belüli programok alkalmazás-specifikus adatok (üzenetek) sorokba írásával és sorokból olvasásával kommunikálnak anélkül, hogy privát, dedikált, logikai kapcsolat lenne közöttük. Az MQSeries az IBM ipari köztes réteg megoldása üzenetküldéshez és sorok használatához. Használatával lehetővé válik olyan programok kommunikációja, amelyek különböző komponenseken vagy hálózaton helyezkednek el.



1. ábra MQSeries futásidőben

Ahogy az ábrán látható, a programok a Message Queue Interface-en (MQI) keresztül Queue Manager-ekkel (MQM) kommunikálnak, amelyek az üzenetsorokat kezelik.[1]

Az üzenetküldés alkalmazásával a programok üzenetek formájában küldenek adatot egymásnak, nem közvetlen hívásokkal. A sorok használatakor a kommunikáló programoknak nem kell konkurensen futniuk.

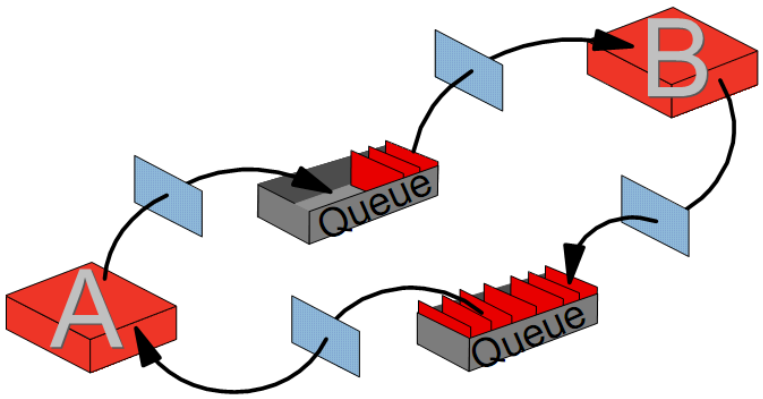
[1] MQSeries Primer: http://www.redbooks.ibm.com/redpapers/pdfs/redp0021.pdf

Aszinkron üzenetküldés esetén a küldő program nem vár válaszüzenetet, hanem továbbhalad, míg szinkron üzenetküldéskor megvárja a választ a végrehajtás folytatása előtt.

2.1 MQSeries

Az MQSeries felhasználható kliens-szerver és elosztott környezetben is, míg az MQSeries-t használó programok többféle programnyelven is készülhetnek (ezek közül a Java az egyik).

Használatakor a programozó nem a cél alkalmazás nevét köti meg, ahova üzenetet küld, hanem egy cél üzenetsort, és minden üzenetsor alkalmazáshoz van rendelve. Egy alkalmazásnak több bemeneti és több kimeneti sora is lehet. A programozónak azzal sem kell törődnie, hogy a cél program elfoglalt vagy egyáltalán elérhető-e, az MQSeries elintézi a célállomáshoz való szállítást, amikor az elindul, ha ez szükséges. Az alkalmazások futhatnak folyamatosan, vagy triggerelhetők is, amikor egy vagy adott számú üzenet érkezett.



2. ábra Üzenetek és sorok

Az ábrán két, egymással kommunikáló program látható, ahol az egyik kimeneti sora a másik bemeneti sora és fordítva. A téglalapok a sorok és programok között az MQI-t, vagy API-t reprezentálják, amelyen keresztül az MQM-el kommunikálnak a programok.

2.2 Üzenetek

Az üzenetek két részből állnak: az adat, amelyet az egyik program a másiknak küld és az üzenet leíró vagy fejléc. A fejlécben találhatók az üzenet azonosítója és egyéb vezérlési információk.

Az MQSeries képes az üzenetek szegmentálására és csoportosítására. A szegmentálás használható a nagyméretű üzenetek felbontására, ha nem férne bele az üzenetsorba, míg az üzenetek csoportosításával kis üzenetek kapcsolhatók össze, a hálózati forgalom csökkentése érdekében.

Az MQSeries négyféle üzenetet ismer:

- Datagram: információs üzenet, amelyre nem várnak választ
- Resuest: olyan üzenet, amelyre választ várnak
- Reply: egy Request üzenetre adott válasz
- Report: egy eseményt leíró üzenet, például hiba jelzés vagy megerősítés egy üzenet megérkezéséről.

Megkülönböztetünk továbbá perzisztens és nem-perzisztens üzeneteket. A perzisztens üzenetek átvitele garantált, még rendszerösszeomlás után is, míg nem-perzisztens üzenetek esetén ilyen garancia nincs.

A fejléc a következő információkat tartalmazza:

Version	Return address
Message ID / Correlation ID	Format
Persistent / non-persistent	Sender application and type
Priority	Report options/ Feedback
Date and time	Backout counter
Lifetime of a message	Segmenting grouping information

A version a használt MQSeries verziójától és a használt platformtól függ. A ID-k az egyes kérés és válasz üzenetek azonosítására valók. A priority segítségével az üzenetek feldolgozási sorrendjét lehet befolyásolni. A Date and time az üzenet küldésének időpontja, a lifetime megadja, hogy mennyi idő után legyen érvénytelen az üzenet. A return address megadja, hogy hova kell válaszolni egy kérésre, ez reply-to sor és egy reply-to queue manager adatokat tartalmaz.

2.3 Queue manager

Az MQSeries központi része az üzenetsor kezelő (MQM), amely a futásidejű része az MQSeries-nek. Feladata az alkalmazásokhoz tartozó sorok és üzenetek kezelése. Ő szolgáltatja az MQI interfészt az alkalmazásoknak a kommunikációhoz.

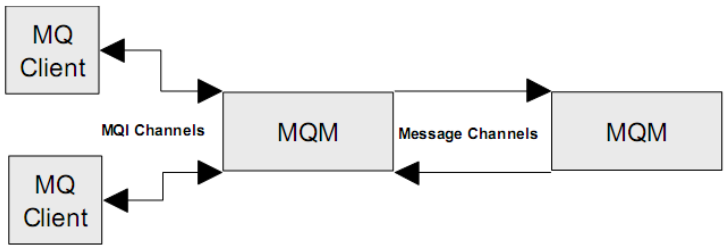
Egy program küldhet üzenetet olyan alkalmazásnak, amely ugyanazon a gépen fut, és olyannak is, amelyik egy távoli gépen. Távoli gépre küldött üzenetek kezeléséhez az MQM csatornákat használ.

Az MQM funkciói a következők:

- Sorok és üzenetek kezelése.
- MQI interfész szolgáltatása.
- Létező hálózati megvalósítások használatával az üzenetek más MQM-ekhez való eljuttatása.
- Adatbázisok és sorok frissítése kétfázisú commit használatával.
- Üzenetek szegmentálása és összerakása, ha ez szükséges. Üzenetek csoportosítása és szétválogatása.
- Egy üzenet több célponthoz való elküldése a hálózati forgalom csökkentésével.
- Adminisztratív funkciók sorok létrehozásához és törléséhez, sorok tulajdonságainak megváltoztatására, és az MQM működésének irányítására. Az MQSeries 5.1-es verzió grafikus interfészt is biztosít, más platformokon parancssoros felületek vagy panelek használhatók.

Az MQM objektum a következő típusú objektumokat használhatja:

- Üzenetsorok: üzenetek tárolása
- Folyamat definíciók: alkalmazások leírása (név, elérési út) triggereléshez
- Csatornák: logikai kommunikációs kapcsolat.
    - Üzenetcsatorna: két MQM közötti egyirányú csatorna üzenetek átvitelére
    - MQI csatorna: kliens és szerver közötti interfész csatorna



3. ábra Üzenet és MQI csatornák

2.4 Üzenetsor

Az MQSeries többféle üzenetsor típust ismer, ezek:

- Local queue: valódi sor, amelyhez alkalmazások kapcsolódnak, az adott MQM tulajdona.
- Cluster queue: olyan local queue, amelyet egy fürtben lévő több MQM ismer.
- Remote queue: más MQM tulajdonában lévő sor helyi definíciója, amely csak üzenetküldést enged meg.
- Transmission queue: speciális local queue, amely köztes lépcső egy remote queue-hoz. Ezen keresztül lehet üzenetet küldeni más MQM-eken található sorokba.
- Dynamic queue: az alkalmazás által definiált local queue, az MQM törölheti, ha az alkalmazás befejezte futását.
- Alias queue: sor definíció, amely segítségével egy sorhoz több név rendelhető különböző tulajdonságokkal.
- Model queue: attribútumok gyűjteménye amelyet dynamic queue-k létrehozásakor használnak
- Initiation queue: Trigger üzenetek tárolására használt sor, két MQSeries alkalmazás figyeli a sort, az alkalmazás indító és a csatornaindító.
- Reply-to-queue: olyan sor, ahova válaszüzeneteket vár egy alkalmazás, a kérés üzenet fejlécében kerül átadásra.
- Dead-Letter queue: Idekerülnek a nem elküldhető üzenetek. Például ha tele van vagy nem létezik a célsor, nem lehet üzenetet elhelyezni a célsorba, túl nagy az üzenet, stb.
- Repository queue: fürtözött MQM-ek által használt sor.

2.5 Queue Manager létrehozása

Egyszerre több MQM is létrehozható és futtatható egymás mellett. A létrehozáshoz a

```
crtmqm [/q] MYQMGR
```

parancs használható, ahol a /q paraméter opcionális és az alapértelmezett MQM kijelölésére alkalmas.

Az MQM neve érzékeny a kis és nagybetűkre. Az MQM-ek létrehozásához és kezeléséhez Windows –on grafikus interfész is elérhető.

A dead-letter queue nem jön létre automatikusan, ehhez az MQM-et a következő paranccsal kell létrehozni:

```
crtmqm /q /u system.dead.letter.queue MYQMGR
```

Az MQM elindítása a

```
strmqm
```

paranccsal történik.

2.6 Kliensek és szerverek

Az MQSeries különbséget tesz kliensek és szerverek között. A kliensek lehetnek vékony- (vagy MQSeries) vagy vastagkliensek. A vastag kliensek rendelkeznek saját MQM-el.

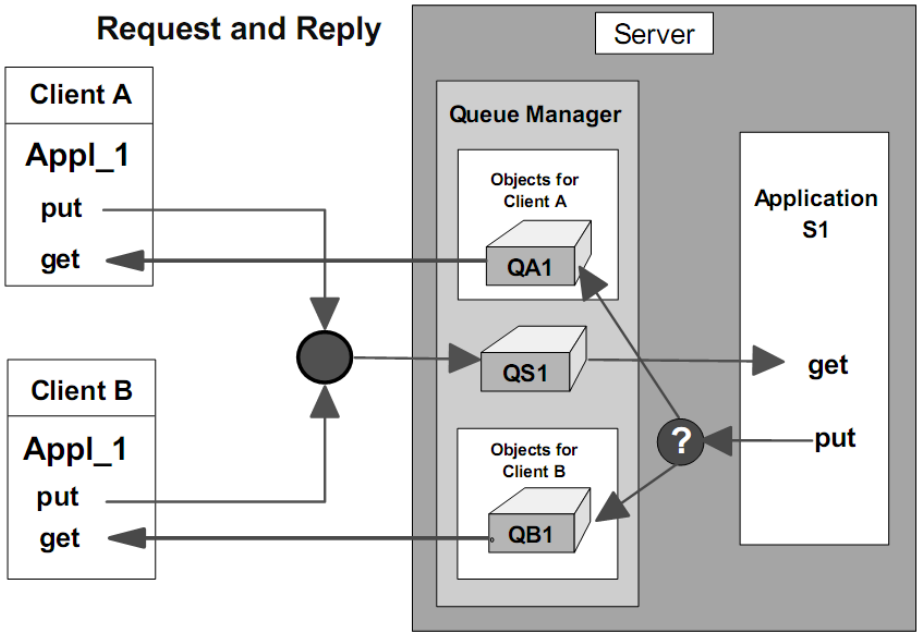
A következő három fajta munkaállomás létezik:

- MQSeries kliens: nincs saját MQM-je, egy szerveren található MQM-en osztozik más kliensekkel.
- MQSeries szerver: kliens és szerver is lehet, köztes csomópontként viselkedik kliensek között.

- Leaf node: egy felhasználó által használt csomópont, amelynek van saját MQM-je, de nem viselkedik köztes pontként más klienseknek. Más néven vastagkliens.

2.7 Üzenetküldés és fogadás

A kliens által elindított program a következőképpen helyezhet el üzenetet egy sorba az MQI hívásával. Először csatlakozni kell a szerveren futó MQM-hez, utána meg kell nyitni az üzenetsort, el kell helyezni az üzenetet a sorban, le kell zárni a sort és végül le kell kapcsolódni az MQM-ről.



4. ábra Kliensek és szerver kommunikációja

A szerveren a következő objektumokra van szükség a kérés fogadásához: egy szerver kapcsolat csatorna, egy local queue, amelybe a kliens az üzenetet teszi, egy initiation queue, amelybe a trigger üzenet a sor megnyitásakor érkezik, a folyamat definíció, amely megadja melyik programot kell elindítani a trigger esemény hatására, végül pedig azok a sorok, amelyekbe a válasz üzeneteket kell tenni.

A választ a kliens kétféleképpen kezelheti:

- Társalgási módban az alkalmazás vár az üzenet megérkezésére a továbbhaladás előtt. Vagyis a válasz sor nyitva van és egy üzenetlekérés várakozással parancsot adott ki az alkalmazás. Az alkalmazásnak kezelnie kell azt, ha az üzenet megérkezik időben és azt, hogy az időzítés lejár és nincs üzenet.
- Igazi aszinkron módban az alkalmazás nem vár válaszüzenetre. Általában későbbi interakció (például gombnyomás) miatt kérdezi le a válasz sor tartalmát. Az üzenetet így az aktuális vagy egy másik program is feldolgozhatja.

2.8 Message Queuing Interface (MQI)

Az interfész a következő 13 API-t tartalmazza:

MQCONN	Kapcsolódás MQM-hez
MQDISC	Lekapcsolódás MQM-ről
MQOPEN	Sor megnyitása
MQCLOSE	Sor lezárása

MQPUT	Üzenet elhelyezése a sorban
MQGET	Üzenet kivétele sorból
MQPUT1	MQOPEN + MQPUT + MQCLOSE
MQINQ	Objektum tulajdonságainak lekérdezése
MQSET	Objektum tulajdonságainak beállítása
MQCONNX	Alapértelmezett vagy gyorsított kapcsolódás
MQBEGIN	Adatbázis tranzakció (munkaegység) indítása
MQCMIT	Munkaegység lezárása
MQBACK	Visszavonás

2.9 WebSphere MQ osztályok Java-hoz

Azoknak a Java osztályoknak a gyűjteménye, amelyek segítségével kapcsolódni lehet a kliensként vagy direktbe egy MQ szerverhez.[2]

A kliens többféle módon tud kapcsolódni a szerverhez: akár Java-t tartalmazó böngészőből, appletviewer-ből, egyszerű Java programból vagy alkalmazásszerverről.

Windows rendszeren feltelepített WebSphere MQ esetén a következő elérési útvonalak lehetnek fontosak:

- Terméktelepítési könyvtár: \Program Files\IBM\WebSphere MQ\java
- Mintagyűjtemény: \Program Files\IBM\WebSphere MQ\tools\Java\
- Környezeti változók:
    - CLASSPATH=mq_root_dir\java\lib\com.ibm.mq.jar; mq_root_dir\java\lib\connector.jar; mq_root_dir\tools\java\base\; mq_root_dir\java\lib\jta.jar;
    - PATH=install_dir\lib

Felhasználáshoz el kell indítani az MQM-et, majd a

```
runmqsc [QMNAME]
```

parancs végrehajtása után definiáljunk egy csatornát a következő paranccsal (vagy használjuk a grafikus interfészt):

```
DEF CHL('JAVA.CHANNEL') CHLTYPE(SVRCONN) TRPTYPE(TCP) MCAUSER(' ') +
DESCR('Sample channel for WebSphere MQ classes for Java')
```

Majd indítsuk el a listener programot a következő paranccsal:

```
runmqlsr -t tcp [-m QMNAME] -p 1414
```

Az alapértelmezett MQM használata esetén a –m kapcsoló elhagyható.

A kapcsolat teszteléséhez lépjünk át a mintagyűjtemény könyvtárba és futtassuk a mintaalkalmazást a következő paranccsal:

```
java MQIVP
```

Ez a program megpróbál csatlakozni és lekapcsolódni a megnevezett MQM-ről, megnyitni, üzenetet elhelyezni, kivenni és lezárni az alapértelmezett soron. Végül visszaad egy üzenetet, ha minden sikerült.

A parancssoron a következőket lehet tenni: TCP/IP kapcsolat esetén írjuk be a szerver host nevét, natív kapcsolat esetén hagyjuk üresen. A többi adat értelemszerűen kitöltendő.

[2] Using Java with WebSphere MQ: http://publibfp.boulder.ibm.com/epubs/pdf/csqzaw12.pdf,

A következő esetleges hibaüzenetek lehetnek:

- Unable to identify local host IP address The server is not connected to the network.
    - Connect the server to the network and retry.
- MQRC_ADAPTER_CONN_LOAD_ERROR
    - If you see this z/OS error , ensure that the WebSphere MQ SCSQANLE and SCSQAUTH datasets are in your STEPLIB statement.

2.10 Java programok írása WebSphere alapon

Egy egyszerű példaalkalmazás leírása és magyarázata megtalálható a „Using Java with Websphere MQ" leírás 7-ik fejezetében. A példaalkalmazásban részletesen be van mutatva az MQM-hez való csatlakozás, egy sor megnyitása, üzenet elhelyezése, üzenet levétele, sor bezárása és MQM-ről való lekapcsolódás.

Ugyanezen leírás 9-ik fejezete részletesen tárgyalja a Java alkalmazásokban használható MQ osztályokat, így a mérés során ezek nézegetése erősen ajánlott.

3 A mérés elvégzése

A mérés során a hallgatóknak át kell alakítaniuk az első mérésen készült programjukat olyan formába, hogy a folyamat egyes lépései közötti adatáramlás üzenetsorok segítségével legyen megoldva. Az összes csomópont külön Java folyamat legyen (nem csak thread), amelyek kizárólag a WebSphere MQ üzenetsorain keresztül kommunikálnak, nincs központi szinkronizációs komponens. A csomópontok között tényleges adatáramlás legyen, ne csak primitív token átadás.

A következő feladatokat kell elvégezni:

- MQManager grafikus interfészén keresztül MQM létrehozása és a folyamat csomópontjai közötti kommunikációhoz szükséges egyes üzenetsorok létrehozása.
- Java osztályok átalakítása, lehetőleg a kommunikáció elkülönítése (ahogy előző mérésen is). Általában egy get és egy put metódus megvalósítása szükséges, amely magába foglalja az MQM-el való kommunikációt.
- Folyamat lépéseinek összekötése a megfelelő ki és bemeneti sorok megadásával.
- Egyszerű grafikus megjelenítés, ahol látható az adott csomóponton elérhető adat és léptethető a folyamat, valamint elágazás esetén döntés kiválasztható.
- A folyamat többszörösen futtatható legyen az MQManager kezelő program használata nélkül (ne maradjon futást zavaró üzenet a sorokban).

Szorgalmi feladatok, ha az előzőek a mérés idejét nem töltik ki:

- A csomópontok folyamatos futása helyett a folyamat használjon triggerelést a csomópontok elindítására az üzenetek érkezésekor.
- Kapcsoljátok be a tracing-et és vizsgáljátok meg, hogy különböző trace szinteken milyen információk nyerhetők ki. A tracing egyébként segíthet a hibakeresésben is, a kötelező feladatok során.
- Módosítsátok az alkalmazást úgy, hogy a reply-to sorok használatával a továbbított üzenetekre visszaigazolást küld a cél csomópont. A felhasználói felületen jelezzétek, hogy egy kérés csak továbbítva lett, vagy a másik fél által a fogadás is megtörtént.

3.1 A mérés kiértékelése

A mérés után leadott jegyzőkönyvben szerepeljen a mérés elvégzéséhez szükséges lépések leírása megismételhető módon. Legyen benne a megvalósított folyamat rövid leírása, azok a műveletek, amelyeket az MQManager grafikus felületén el kellet végezni, a létrehozott üzenetsorok és paramétereik. Szerepeljen benne az, hogy hogyan és mennyire kellett az eredeti programot megváltoztatni, hogyan valósították meg az egyes csomópont típusokat. Tartalmazzon megfelelő mennyiségű képernyőképet és útmutatót ahhoz, hogy hogyan kell lefuttatni az elkészült programot.

4 Ellenőrző kérdések

A beugró kérdések kizárólag ebben a dokumentumban leírtakat fogják visszakérdezni, a két linkelt leírás a mérés elvégzéséhez szükséges pluszinformációkat tartalmazzák.

1. Mik az üzenetsor alapú kommunikáció előnyei?
2. Hogyan valósítható meg szinkron működés üzenetsorokkal?
3. Milyen feladatok végezhetők el az MQManager-el?
4. Milyen típusú sorok vannak az MQSeries-ben?
5. Milyen típusú üzenetek vannak az MQSeries-ben?