# تحقيق نظام ملـفّات مـوزّع
# على منصّة عمل مجموعة الغرض الموزّع "Jgroup"

*م. علي اسماعيل ** أ.د رضوان دنده *** د. قاسم قبلان

(الإيداع:26 كانون الأول 2017، القبول: 29 كانون الثاني 2018)

**الملـخّص:**

تهدف تقنيات بناء الأنظمة الموزّعة إلى تحقيق تفاعل بين مكوّنات التطبيق المتوضّعة على أجهزة متعدّدة والمتّصلة مع بعضها عبر شبكة اتصال. ونظراً للاعتماد المتزايد على هذه الأنظمة في النشاطات اليومية؛ توجّب على تقنيات بنائها تقديم خدمات متوفّرة وموثوقة. لم تتمكّن RMI وCORBA من تحقيق ذلك؛ لعدم دعمها نمط التفاعل one–to–many في اتصالاتها.

تزوّد منصّة عمل مجموعة الغرض الموزّع Jgroup خدمات معتمدة من خلال دمجها لتقنية مجموعة الغرض (Object Group) الذي يدعم نمط التفاعل (one–to–many) مع نموذج الغرض الموزّع Java RMI، فقدّمت بذلك نظام اتصالات مجموعة يعتمد تقنية واحدة (RMI) في جميع تفاعلاته؛ سواء الداخلية لتحقيق التنسيق بين أغراض مجموعة المخدم أو الخارجية اللازمة لاتصال الزبون مع مجموعة الغرض.

تحقّق هذه المقالة نظام ملفات موزّع على Jgroup يستفيد من المزايا التي تقدّمها هذه المنصّة في إجراء العمليات الأساسية (استعراض جميع الملفات، تحميل ملف من النظام، رفع ملف، تعديل ملف).

يقيّم البحث أداء النظام من خلال قياس زمن التأخير اللازم لرفع ملفات (upload) بحجوم مختلفة من الزبائن؛ وذلك مع تزايد عدد المخدّمات المتوفّرة ضمن مجموعة غرض Jgroup. كما يقيّم الأداء من خلال قياس زمن التأخير اللازم لتحميل ملف (download) وذلك مع تزايد عدد طلبات الزبائن للحصول عليه.

تظهر النتائج التي تم التوصّل إليها تزايد زمن التأخير اللازم لرفع ملفات من قبل الزبائن مع تزايد عدد المخدمات المتواجدة ضمن مجموعة الغرض، كما تبيّن عدم تأثر زمن التأخير اللازم لتحميل ملف مع تزايد عدد طلبات الزبائن.

**الكلمات المفتاحية:** النظام الموزّع؛ مجموعة الغرض؛ أنظمة اتصالات المجموعة؛ Jgroup؛ استدعاء الطريقة عن بعد؛ نظام ملفات موزّع.

* طالب دراسات عليا (دكتوراه) – قسم النظم والشبكات الحاسوبية– كلية الهندسة المعلوماتية– جامعة تشرين– اللاذقية .

** أستاذ– قسم النظم والشبكات الحاسوبية– كلية الهندسة المعلوماتية – جامعة تشرين– اللاذقية.

*** مدرّس– قسم النظم والشبكات الحاسوبية– كلية الهندسة المعلوماتية – جامعة تشرين– اللاذقية.

# Implementation of Distributed File System on Object Group Platform "Jgroup"

*Eng. Ali Esmaeel ** Dr. Radwan Dandah ***Dr. Kasem Kabalan

(Recived:26 December 2017, Accepted: 29 January 2018)

**Abstract:**

Distributed systems building techniques aims to interact the application components which are placed on different machines and connected through a network. The increasing reliance on these systems in day-to-day activities requires that their techniques to provide available and reliable services. Neither RMI nor CORBA can achieve this requirement; because they cannot support one-to-many interaction.

Jgroup provides dependable services through its integration of object group paradigm with the distributed object model of Java RMI, so it offers a Group Communication System depends on RMI in all its interactions; whether internal for coordination between object group replicas, or external for communicating clients with object group.

This paper implements a distributed file system benefits of all provided facilities of Jgroup in performing the main operations (list all files, download a file, upload a file, and modify a file).

This research evaluates performance of the system through measurement of delay required to upload files of different sizes from clients; with increasing the number of servers.

In addition to evaluation of performance through measurement of delay required to download a file with increasing the number of clients' requests.

Results show that the delay for uploading a file increases with increasing number of servers in object group, and the delay for downloading a file doesn't affected by increasing the number of clients' requests.

**KEYWORDS:** Distributed System; Object Group; Group Communication System; Jgroup; Remote Method Invocation; Distributed File System.

---

* Postgraduate Student, Department of Systems and Computer Networks, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.

** Professor, Department of Systems and Computer Networks, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.

*** Lecturer, Department of Systems and Computer Networks, Faculty of Information Engineering, Tishreen University, Lattakia, Syria.

## 1- المقدمة:

تتعاون مجموعة الأجهزة المكوّنة للنظام الموزّع فيما بينها لتزويد تطبيق معيّن؛ بحيث تظهر إلى مستخدمي هذا التطبيق كنظام وحيد مترابط [12]. يدعم النظام الموزّع قابلية التوسّع (Scalability) من خلال سماحه بإضافة أجهزة جديدة، كما يتفوّق على نظيره المركزي في تحقيقه التوافرية ودعمه التسامح مع الخطأ.

تعتبر RMI (Remote Method Invocation) [10] إحدى التقنيات المستخدمة في بناء الأنظمة الموزعة، فهي تسمح لأغراض جافا المتوضّعة على أجهزة مختلفة بالتفاعل مع بعضها البعض كما لو أنّها على جهاز واحد. يستدعي غرض جافا طرائق على غرض المخدّم (غرض مخصّص لتزويد مجموعة من الخدمات)، وتعالج RMI عملية تنظيم (Marshaling) تشمل تجميع وتغليف للبارامترات الممررة إلى الطريقة المستدعاة والقيم المعادة منها.

تقدّم لغة البرمجة جافا أيضاً تقنية أخرى وبمستوى عالي لبناء الأنظمة الموزعة تعتبر امتداداً لتقنية RMI؛ وهي Jini [11]. تزوّد هذه التقنية خدمة الاستعلام (Lookup)؛ وتنهي حاجة الزبون إلى معرفة مكان تواجد الخدمة من خلال تزويدها لآلية الاكتشاف (discovery mechanism). تسمح آلية الاكتشاف للخدمات بإيجاد مواضع خدمة الاستعلام وتسجيل وكلاء لها ضمنها، كما تسمح لطالبي الخدمة –الزبائن– بتحديد مواضع خدمة الاستعلام والاتصال معها للحصول على وكيل الخدمة المراد استخدامها.

تساهم CORBA (Common Object Request Broker Architecture) [10] في بناء الأنظمة الموزّعة، من خلال سماحها للبرامج المطوّرة بلغات مختلفة (مثل Java،++C ، C وغيرها)؛ المتنوّعة في تحقيقاتها (Implementations)؛ والعاملة في مواقع مختلفة أن تتصل مع بعضها البعض بسهولة، كما لو أنها في موضع واحد. تمثّل المنحى العام لزيادة توافريه الخدمة بإنشاء نسخ متعدّدة (Replication) من الخدمة نفسها وتوزيعها جغرافياً على عدة أجهزة، ويتطلب ذلك دعماً من النظام لنمط التفاعل one-to-many. لا تزوّد التقنيات السابقة هذا النمط من التفاعلات؛ في حين تزوّده أنظمة اتصالات المجموعة (Group Communication Systems)[7] . تستخدم هذه الأنظمة نموذج مجموعة الغرض (Object Group Pattern) [9]، يقوم هذا النموذج بتجميع مجموعة من أغراض المخدم منطقيّاً في مجموعة، ويصبح بإمكان الزبون أن يتفاعل بشكل شفّاف مع هذه المجموعة كما لو أنّها مخدّم وحيد مفرد.

تتميز Jgroup [1] عن غيرها من أنظمة اتصالات المجموعة بدمجها لنموذج مجموعة الغرض مع نموذج الغرض الموزّع RMI، فقدّمت بذلك نظاماً متكاملاً يعتمد تقنية واحدة (RMI) في جميع تفاعلاته، سواء الداخلية لتحقيق التنسيق بين نسخ الخدمة أو الخارجية اللازمة لتحقيق اتصال الزبون مع مجموعة المخدم.

طُوّرت Jgroup إلى منصّة العمل Autonomous Replication Management Jgroup/ARM [6]، والتي تسمح بإنشاء نسخ المخدّم على الأجهزة آلياً دون تدخل بشري، وتحافظ على عدد محدد وثابت من النسخ في بيئة الهدف من خلال مراقبتها عبر مكوّن مدير النسخ (Replication Manager). طوّر ميلينغ المنصة ARM إلى منصة جديدة Distributed Autonomous Replication Management Jgroup/DARM)) [5]، يتم فيها توزيع النسخ بطريقة موزّعة دون الحاجة إلى المكوّن المركزي المتمثّل بمدير النسخ المستخدم في ARM.

اقترح موقع مشروع Jgroup [1] إضافة وتحقيق مزايا جديدة في المنصة، كدمج قاعدة بيانات مع Jgroup باستخدام Hibernate، تحقيق خدمة JavaSpaces مكررة باستخدام Jgroup، تحسين خوارزمية عضوية المجموعة المستخدمة في Jgroup ومزايا أخرى. يحقّق هذا البحث نظام ملفات موزّع على Jgroup يسمح بالعمليات الأساسية (استعراض الملفات المتوفّرة، جلب ملف، إرسال ملف، تعديل ملف).

تم تنظيم هذه المقالة على النحو التالي: تقدّم الفقرة 4 توضيحاً لمراحل العمل في Jgroup مع التمييز بين أنماط استدعاء طريقة المجموعة الخارجي. تصف الفقرة 5 نظام الملفات المنشأ والعمليات الأساسية فيه ونمط الاستدعاء المستخدم في كل عملية. تعرض الفقرة 6 نتائج تقييم الأداء التي تم التوصل إليها. تختتم الفقرة 7 هذه المقالة من خلال اقتراح مجموعة من التوصيات والأعمال المستقبلية.

## 2- أهمية البحث وأهدافه:

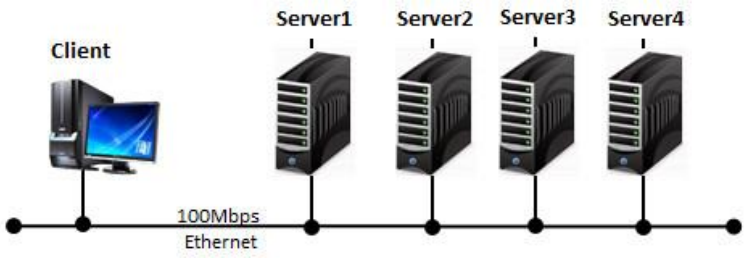
في الوقت الذي نسعى فيه إلى وجود عدد قليل نسبياً من المخدّمات لتحقيق المستويات المرغوبة من التوافرية والموثوقية؛ مع احتمالية وجود عدد كبير من زبائن الخدمة، تدعم Jgroup قابلية التوسّع (Scalability) لعدم حاجة زبائنها إلى الانضمام إلى مجموعة غرض المخدم قبل الحصول على الخدمة المرغوبة.

يهدف البحث إلى تحقيق نظام ملفات موزّع على Jgroup يستثمر جميع التسهيلات التي تقدّمها هذه المنصة كدعمها لقابلية التجزئة من خلال خدمة عضوية المجموعة القابلة للتجزئة؛ تتبّع هذه الخدمة مسار التغيّرات في الشبكة (تعطّل أحد المخدمات أو انقطاع في شبكة الاتصال) لتزوّد كل مخدّم بتقرير يسمى منظاراً (view) يحوي قائمة بالأعضاء الحاليين القابلة للاتصال والتنسيق فيما بينها. تتميز هذه الخدمة في Jgroup بأنها تحافظ على استمرارية توفّر الخدمة الموزّعة في جميع أجزاء الشبكة؛ بدلاً من محدوديتها في جزء واحد فقط.

تعالج Jgroup أيضاً جميع المشاكل الناشئة عن عملية دمج أجزاء الشبكة بعد العودة من التجزئة من خلال خدمة دمج الحالة (State Merging Service SMS) التي تسهّل عملية عودة النظام إلى حالة عامة متناسقة بين جميع المخدمات.

## 3- منهجية البحث:

تمّ تحضير بيئة الهدف في مخبر الدراسات العليا في كلية الهندسة المعلوماتية بجامعة تشرين. تتألف هذه البيئة الموضّحة في الشكل (1) من خمسة أجهزة متصلة عبر شبكة محلية (Ethernet 100Mbps). يتطلّب تشغيل تطبيق Jgroup تحميل البرمجيات الموضّحة في الجدول (1) وذلك على كل جهاز في بيئة الهدف.




**الشكل رقم (1): بيئة الهدف.**

طوّرت Jgroup ب استخدامJAVA J2SE version1.5، ويمكن تحميل هذه النسخة أو أية نسخة أحدث منها(هنا تم اختيار النسخة 1.8)، تزوّد الحزمة Apache Ant بنسختها 1.8.4 وسيلة سهلة وبسيطة لترجمة واختبار وتشغيل تطبيقات Jgroup، بينما تستخدم الأداة log4j بهدف تصحيح الأخطاء التي يمكن مواجهتها خلال تشغيل التطبيقات، بعد تحميل هذه الأدوات على الأجهزة في بيئة الهدف؛ يصبح بإمكاننا تشغيل تطبيقات Jgroup، وتحقيق نظام ملفات موزّع على Jgroup وإجراء الاختبارات المطلوبة.

**الجدول رقم (1): البرمجيات المطلوبة لتشغيل تطبيقات Jgroup**

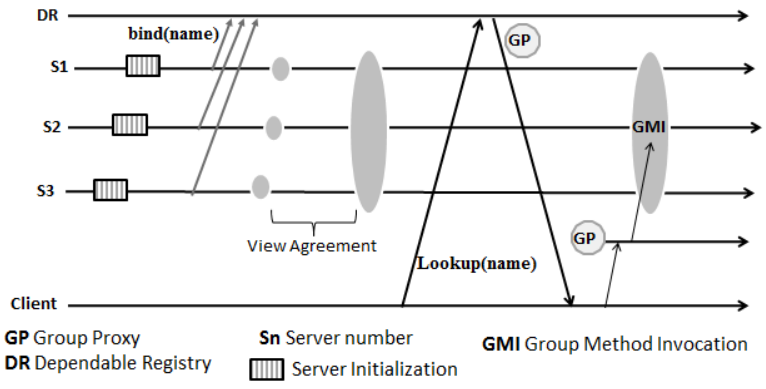
| | |
|---|---|
| Jgroup 3.0.1 distribution | [1] |
| Java J2SE version 1.8.0-20 | [2] |
| Apache Ant version 1.8.4 | [3] |
| Apache Log4j version 1.3-alpha-8 | [4] |

بعد تحضير بيئة الهدف وتحقيق نظام الملفات المطلوب، يتم تقييم الأداء عن طريق قياس زمن التأخير اللازم لرفع ملفات بحجوم مختلفة (10, 20, 50, 100KB) من جهاز الزبون إلى النظام، وذلك من أجل نسخة مخدّم واحدة فقط، أما في السيناريو الثاني من أجل نسختي مخدم وفي الثالث من أجل ثلاث نسخ خدمة. وفي الرابع من أجل أربع نسخ خدمة. تتوضّع نسخ الخدمة على الأجهزة Server1, Server2, Server3, Server4 على الترتيب.

## 4- مراحل العمل في Jgroup:

تزوّد Jgroup ثلاث خدمات أساسية وهي خدمة عضوية المجموعة القابلة للتجزئة ( Partitionable Group Membership Service PGMS)، خدمة دمج الحالة (State Merging Service SMS) وخدمة استدعاء طريقة المجموعة التي تنقسم إلى نوعين: داخلي (Internal Group Method Invocation IGMI) تسمح بالاتصال بين أعضاء المجموعة من خلال البث المتعدد، وخارجي (External Group Method Invocation EGMI) يسمح للزبائن بالاتصال مع أعضاء المجموعة كما لو أنها كيان واحد.

يوضح الشكل (2) مراحل العمل في Jgroup، حيث تقوم نسخ المخدمات (s1,s2,s3) بتسجيل نفسها تحت اسم المجموعة ضمن المسجل DR (Dependable Registry) في Jgroup من خلال الطريقة () bind. تكتشف خدمة عضوية المجموعة على كلّ نسخة مخدم قدرة جميع النسخ الثلاث على الاتصال فيما بينها، فتقوم بتحميل منظار واحد مؤلف من النسخ الثلاث، وذلك على كلّ نسخة مخدم منها. حتى يتمكّن الزبون من الاتصال مع مجموعة المخدم؛ ينبغي عليه الاستعلام (Lookup) من DR عن اسم المجموعة المطلوبة، ليحصل على وكيل المجموعة GP (Group Proxy)، يحوي وكيل المجموعة معلومات عن النسخ المكوّنة للمجموعة، ممّا يمكّن الزبائن من الاتصال مع المجموعة ككيان مفرد بإنجاز الاستدعاء عبر هذا الوكيل، تنسّق المخدمات أحداثها ضمن المنظار نفسه للحفاظ على حالتها المشتركة وتزويد خدمة موثوقة من خلال خدمة استدعاء الطريقة الداخلي (Internal Group Method Invocation IGMI).




**الشكل رقم (2): مراحل العمل في Jgroup**

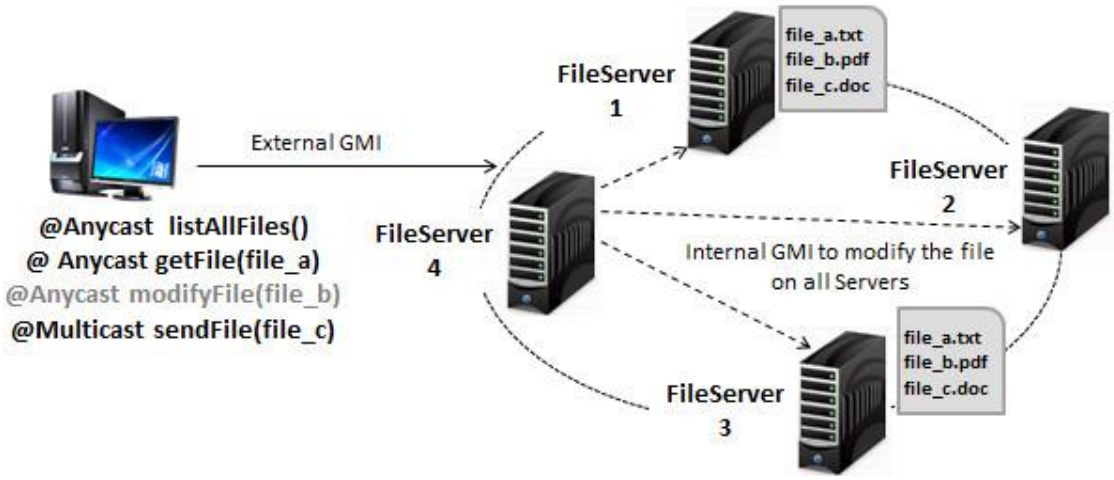
تعيد خدمة استدعاء طريقة المجموعة التي ينفّذها الزبون على مجموعة الغرض إجابة واحدة فقط بدلاً من مصفوفة نتائج، فيحقّق بذلك الشفافية في عملية الاستدعاء (كما لو أنه استدعاء طريقة على مخدم مفرد)، يمكن التمييز بين نوعين من استدعاء طريقة المجموعة الخارجي:

- Anycast EGMI: يتم إنجاز الاستدعاء من خلال تنفيذ الطريقة على مخدم واحد فقط على الأقلّ. ويعتبر بذلك ملائماً للطرائق التي لا يشمل تنفيذها كامل أعضاء المجموعة، كطرائق القراءة من قواعد البيانات (وهو ملائم في نظامنا لحالة استعراض كامل الملفات وحالة جلب ملف).
- Multicast EGMI: يتم إنجاز الاستدعاء من خلال تنفيذ الطريقة نفسها على كل مخدم موجود ضمن جزء الشبكة الذي ينتمي إليه الزبون المستدعي، وهي ملائمة للطرائق التي يؤثر تنفيذها على كلّ مخدم في المجموعة (وهو ملائم لحالة رفع ملف في نظامنا).

يضمن تحقيق EGMI في Jgroup خاصية تزامن المنظار (view synchrony)، فعندما يقوم مخدمان (s1, s2) بتحميل زوج من المناظير المتتالية (v1,v2) من خلال خدمة عضوية المجموعة، فإن كل من المخدمين يكون قد نفّذ المجموعة نفسها من استدعاءات EGMI وذلك ضمن المنظار الأول v1. بمعنى آخر، قبل أن يتم تحميل المنظار الجديد v2، يتوجّب على جميع المخدمات المنتمية إلى v1 أن تتوافق على جميع استدعاءات EGMI المنجزة خلال تواجدها معاً في المنظار v1. تضمن هذه الخاصية تكامل البيانات، ففي حالة نظام ملفات موزع نضمن أن التعديل المنجز على أحد النسخ من خلال استدعاء طريقة مجموعة خارجي قد أنجز على جميع المخدمات فتحافظ بذلك على حالة مشتركة فيما بينها.

### 5- تحقيق نظام ملفات موزّع على Jgroup:

يوضّح الشكل (3) توصيفاً لنظام الملفات المنشأ على Jgroup. يسمح النظام للزبائن بتنفيذ أربع عمليات: استعراض كامل الملفات، الحصول على ملف من النظام (تحميل)، تعديل ملف، وإرسال ملف لتخزينه (رفع ملف).




**الشكل رقم (3): نظام ملفات موزّع على Jgroup.**

تم تحقيق النظام على النحو التالي: ينضم المخدم الأوّل File Server1 إلى مجموعة الغرض (File Server)، حيث يحوي هذا المخدم على الملفات المراد التعامل معها ضمن النظام، بعدها يتم انضمام المخدم الثاني FileServer2 إلى مجموعة الغرض نفسها File Server، عندها تكتشف خدمة عضوية المجموعة حالة الانضمام هذه لتقوم بتشكيل منظار جديد يحوي كلا المخدمين (FileServer1, FileServer2)، يقوم المخدم الجديد المنضم وهو FileServer2 بالحصول

على جميع الملفات المخزنة على FileServer1 باستدعائه طريقة من النمط الداخلي () get Files؛ حيث تمت هنا الاستفادة من خدمة استدعاء طريقة المجموعة الداخلية التي تزوّدها Jgroup لتحقيق التنسيق بين المخدمات. يحدث الأمر نفسه مع انضمام المخدمين (FileServer3, FileServer4) من حيث تشكيل مناظير جديدة والحصول على الملفات المطلوبة. يمتلك بعدها كل مخدم من المخدمات الأربع الموضحة في الشكل (3) مجموعة الملفات نفسها. وجميع المخدمات قادرة على الاتصال والتنسيق فيما بينها، يمكن للزبائن هنا طلب تنفيذ إحدى العمليات الأربع.

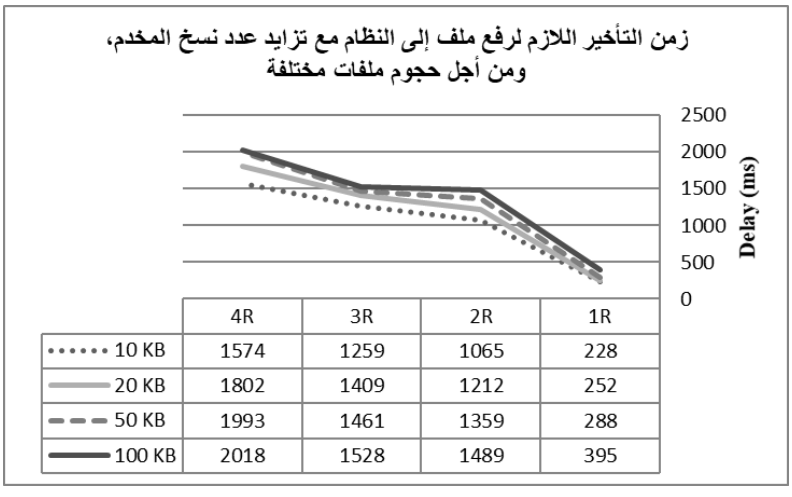
إن طلب استعراض الملفات يمكن إنجازه من خلال الاتصال مع نسخة مخدم واحدة فقط، كما أن تنفيذه لا يؤثر على حالة النظام (يكافئ نمط الاستدعاء Anycast)، وكذلك الأمر بالنسبة لطلب الحصول على ملف، في حين إرسال ملف لتخزينه على النظام، يتطلب اتصالاً مع جميع المخدمات ضمن مجموعة الغرض لإضافة هذا الملف لديها(وهو مكافئ لنمط الاستدعاء Multicast)، تسبب عملية تعديل أحد الملفات تغييراً في حالة النظام، وبالتالي يمكن إنجازها من خلال الاتصال مع مخدم واحد لتعديل الملف المطلوب (Anycast) ليتم ضمن هذا الاستدعاء تنفيذ المخدم المتصل معه لطريقة من النمط الداخلي يتم من خلالها نقل التعديل نفسه على الملف إلى باقي المخدمات ضمن مجموعة الغرض.

تكتشف خدمة عضوية المجموعة القابلة للتجزئة التي تزوّدها Jgroup التغيّرات الحاصلة في شبكة الاتصال كحدوث انقطاع يؤدي إلى انفصال المخدمين FileServer1, FileServer2 عن المخدمين FileServer3, FileServer4. لتقوم بتشكيل منظارين (view1, view2) يكافئ كل منهما جزءاً من الشبكة، يحوي كل منظار على مخدمين، وتبقى خدمة التعامل مع الملفات متوفرة ضمن كل جزء، في حين لا تسمح أنظمة اتصالات مجموعة أخرى بذلك لتبقى الخدمة متوفرة ضمن جزء واحد فقط يسمى بالجزء الأساسي (primary–partition).

تعالج Jgroup أيضاً عملية العودة من التجزئة، كعودة الاتصال بين الجزأين، حيث تقوم خدمة عضوية المجموعة بتشكيل منظار جديد يحوي المخدمات الأربع، وتتم معالجة جميع التحديثات المتناقضة التي يمكن حصولها خلال فترة التجزئة باستخدام خدمة دمج الحالة التي تزودها Jgroup.

**6- نتائج ومناقشة:**

يظهر الشكل (4) زمن التأخير اللازم لرفع ملفات بحجوم مختلفة إلى النظام (مقدراً بالميلي ثانية) مع تزايد عدد نسخ المخدم.




| | 4R | 3R | 2R | 1R |
|---|---|---|---|---|
| 10 KB | 1574 | 1259 | 1065 | 228 |
| 20 KB | 1802 | 1409 | 1212 | 252 |
| 50 KB | 1993 | 1461 | 1359 | 288 |
| 100 KB | 2018 | 1528 | 1489 | 395 |

**الشكل رقم (4): زمن التأخير (ميلي ثانية) اللازم لتحميل ملفات بحجوم مختلفة مع تزايد عدد نسخ الخدمة.**

✓ يتزايد زمن التأخير اللازم لإرسال ملف إلى النظام مع تزايد عدد نسخ الخدمة (Number of Replicas). فمن أجل إرسال ملف بحجم 10KB يتزايد زمن التأخير من 228 ميلي ثانية في حالة مخدم واحد إلى 1574 ميلي ثانية مع وجود 4

مخدمات، وكذلك الأمر مع الملفات بحجوم (20, 50, 100KB). يعود سبب ذلك إلى تنفيذ بروتوكولات الموثوقية في النمط multicast (نمط استدعاء الطريقة () send File) والتي تضمن الحفاظ على الحالة المشتركة بين جميع نسخ المخدم المتواجدة معاً ضمن المنظار نفسه.

- ✓ من أجل عدد نسخ محدد ضمن مجموعة الغرض؛ لا يؤثر تزايد حجم الملف بشكل كبير على زمن التأخير،فمن أجل نسخة مخدم واحدة فقط، تزايد زمن التأخير من 228 ميلي ثانية لاستعادة ملف بحجم 10KB إلى 395 ميلي ثانية لاستعادة ملف بحجم 100KB.

يظهر البحث **فائدة إنشاء نظام ملفات على Jgroup**، من خلال تقييم أداء النظام في استجابته لطلبات الحصول على ملفات (تحميل ملف) مع وجود 5 مخدّمات ضمن مجموعة غرض Jgroup. حيث يتم قياس زمن التأخير اللازم لجلب ملف على كل زبون، ودراسة تأثير تزايد عدد الزبائن على هذا الزمن. يمثّل زمن التأخير الفاصل الزمني ما بين لحظة استدعاء الزبون لطريقة الحصول على الملف (() read File) وحتى حصوله على هذا الملف، ويتم حسابه من خلال تعليمات جافا يتم إدراجها ضمن كود الزبون.

يوضّح الجدول 2 أزمنة التأخير اللازمة لاستعادة ملف بحجم 10KB على كل زبون، حيث يتزايد عدد الزبائن من 1 إلى 5، ليتم دراسة مدى تأثير هذه الزيادة على زمن التأخير اللازم لاستعادة الملف على الزبون الأول (Client1). يوضّح الجدول 3 زمن التأخير اللازم لاستعادة الملف على الزبون الأول مع تزايد عدد الزبائن من 6 إلى 10.

**الجدول رقم (2): أزمنة التأخير (ميلي ثانية) اللازمة لاستعادة ملف بحجم 10KB على كل زبون (يتزايد عدد الزبائن من 1 إلى 5).**

| Client 5 | Client 4 | Client 3 | Client 2 | Client 1 |
|---|---|---|---|---|
| - | - | - | - | 11243 |
| - | - | - | 12066 | 11488 |
| - | - | 11925 | 12546 | 11361 |
| - | 12055 | 10504 | 11909 | 11290 |
| 11745 | 11307 | 10492 | 12600 | 12140 |

**الجدول رقم (3): أزمنة التأخير اللازمة لاستعادة ملف على الزبون الأول (مع تزايد عدد الزبائن من 6 إلى 10).**

| Delay on Client1 (MS) | Number of Clients |
|---|---|
| **11938** | **6 Clients** |
| **11944** | **7 Clients** |
| **11933** | **8 Clients** |
| **12028** | **9 Clients** |
| **11821** | **10 Clients** |

- ✓ يلاحظ من الجدول 2 والجدول 3 عدم تأثر زمن التأخير اللازم لتحميل ملف من قبل Client1 مع تزايد عدد طلبات الزبائن، فقد بقي بحدود 11 ثانية ليصل أحياناً إلى 12 ثانية في حالة 5 زبائن وحالة 9 زبائن. وهذا لا يتعلق بتزايد عدد الطلبات وإنما ناتج عن تأخيرات شبكية بسبب اتصال الزبون مع مجموعة الغرض. وكذلك الأمر بالنسبة لباقي الزبائن؛ ففي الجدول 2 تطلبت معظم الزبائن تأخيرات مشابهة للزبون الأول للحصول على الملف.
- ✓ نظراً للآلية التي تتبعها Jgroup في استدعاء طريقة المجموعة الخارجي من النوع Anycast، حيث لا تقوم خدمة استدعاء طريقة المجموعة الخارجية في Jgroup باختيار المخدم نفسه للإجابة على كلّ طلب؛ فهي تقوم بتوجيه كلّ زبون إلى مخدّم مختلف ضمن مجموعة الغرض ممّا يساعد على موازنة الحمل بين مخدمات مجموعة الغرض ويزيد من سرعة تنفيذ الطلبات (يبرز هنا فائدة إنشاء نظام ملفات موزّع باستخدام Jgroup).

## 7- الاستنتاجات والتوصيات:

حققنا في هذه المقالة نظام ملفات موزّع على Jgroup يستفيد من الميزات التي تقدّمها هذه المنصة لتحقيق التوافرية والموثوقية المطلوبة، من خلال معالجتها لحالات التجزئة الممكن حصولها في شبكة الاتصال.

a. يساهم النظام المحقّق في زيادة توافريه الخدمة، حيث لا يؤدي تعطّل أحد المخدمات إلى ضياع الملفات وتوقّف الخدمة؛ بسبب وجود نسخ مشابهة منها على باقي المخدمات. كما يحسّن من أداء النظام مع وجود عدد كبير من الزبائن.
b. يتزايد الزمن اللازم لإجراء عملية تعديل على الملفات (رفع ملف) مع تزايد عدد نسخ المخدم، يعود ذلك إلى تنفيذ بروتوكولات الموثوقية المطلوبة لتحقيق تزامن المنظار وتحقيق تكامل البيانات بين النسخ المكررة.
c. من الممكن العمل على تحقيق نظام ملفات على Jgroup يستخدم التقنية zero-copy [13] والتي تسرّع من أداء النظام. كما يمكن إجراء مقارنة بين نظامنا مع نظام ملفات محقق على نظام اتصالات مجموعة آخر.
d. تعتبر إضافة مزايا جديدة إلى Jgroup من الأعمال المستقبلية المقترحة؛ من المزايا المقترح إضافتها:
   - تحسين الخوارزمية التي تعتمدها الخدمة الأساسية في Jgroup وهي خدمة عضوية المجموعة.
   - دمج قاعدة بيانات مع Jgroup بالاعتماد على أدوات مقابلة الغرض العلائقية ( Hibernate, [15] Eclipse [14] Link).
   - تحقيق خدمة JavaSpaces [16] مكررة على Jgroup.

## 8- المراجع العلمية المستخدمة في البحث:


1- UNIVERSITY OF BOLOGNA, 2008, Jan.2017<http://jgroup.sourceforge.net/index.html>

2- SOFTONIC, 2009, June.2016. <http://java-development-kit-jdk.en.softonic.com/>.

3- APATCHE ANT ,2014, 3February.2016. <http://ant.apache.org/>.

4- APACHE ,2012, 10July.2016. <http://logging.apache.org/log4j/1.2/>.

5- Meling, H., (2008), An **Architecture for Self-healing Autonomous Object Groups**, University of Stavenger, Department of Electrical Engineering and Computer Science, N-4036 Stavenger, Norway.

6- Meling, H.; Montresor, A.; Helvik, B. E. and Babaoglu, O., (2008), **Jgroup/ARM: a distributed object group platform with autonomous replication management**, Softw. Pract. Exper., 38: 885–923. DOI: 10.1002/spe.853.

7- Vitenberg, R.; Keidar, I.; Chockler, G. and Dolev, D., (1999), **Group Communication Specifications: A Comprehensive Study**, Technical Report CS99-31, Institute of Computer Science, The Hebrew Univ. of Jerusalem.

8- Ban, B., (1998), **JavaGroups:Group communication patterns in Java ,** Technical Report, Department of Computer Science, Cornell University.

9- Montresor, A., (2000), **System Support for programming object-oriented dependable applications in partitionable systems**, (PhD Thesis), University of Bologna, Department of Computer Science. PP: 87-100.

10- Deitel, H.M., Deitel, P.J. and Santy, S.E., (2001), **Advanced Java 2 Platform: How to PROGRAM,** New Tersey: Prentice-Hall, Inc.

11- Deitel, H.M., Deitel, P.J. and Santy, S.E., (2002), **Advanced Java 2 Platform: HowTo PROGRAM**, New Tersey: Prentice-Hall, Inc, pp: 1260-1317.

12- Bernstein, P., (1996), **Middleware: A Model for Distributed System Services,** Communications of the ACM, 39:2, 86-98.

13- Palaniappan, S.K. Nagaraja, P.B., (2008), **Efficient data transfer through zero copy**, www.ibm.com/developerworks/ibm/trademarks/.

[14] MINTER, D.; LINWOOD, J. 'Beginning Hibernate: From Novice to Professional'. Apress,Inc, United States of America, 2006, pp. 11-25.

[15] ECLIPSE, 2016, 10January.2016.
<http://www.eclipse.org/eclipselink/documentation/2.6/concepts/toc.htm>.

[16] Deitel, H.M., Deitel, P.J. and Santy, S.E. **Advanced Java 2 Platform: How to PROGRAM**, New Tersey: Prentice-Hall, Inc, 2002