الجمهورية العربية السورية
المعهد العالي للعلوم التطبيقية والتكنولوجيا
العام الدراسي 2017/2016

منهجية إدارة آمنة لتجزئة الملفات الموزعة في بنوك المعلومات الصغيرة

Methodology for Safe Management of Fragment Distribution files in Small Data Banks

دراسة أعدت لنيل درجة الماجستير في المعلوميات – نظم المعلومات ودعم القرار

إعداد
م. أحمد مهنا

إشراف
د. عصام غياض - د. محمد الجنيدي

دمشق في 2017/10/25

إهداء:

إلى من أستمد منهم القوة والتحمل، إلى من علماني الصبر والمثابرة، وغمراني دائماً بحبهما... أبي وأمي

إلى روح أخي الطاهرة

إلى أخوتي الأعزاء الذين دعموني وشاركوني ما أنا به...أنس ومحمد

كما أتقدم بالشكر العميق إلى أساتذتي

الدكتور عصام غياض

الدكتور محمد جنيدي

الدكتور كادان جمعة

لما قدموه لي من توجيهات ونصائح وأفكار وإشراف مميز لإتمام هذا العمل وإخراجه بالشكل اللائق.

Abstract

Most of the tools used in files fragmentation and distribution management domain, are based on a main idea which is dividing the folder into a specific number of segments and then re-assembling them to get the same original folder at the end .

Through my referential study it turned out that the mathematical samples used are doing the fragmentation and re-assembling procedures as mentioned before, but this process did not take into consideration the idea of safety and the fact that the files are originally important and critical data which affiliate an institutional information bank with a special status, that's why I designed a mathematical model more modified than the existing models, as I added the institution's requirements and its governing rules, without turning to segmentation, but instead I framed it within the frame of an institutional information bank. Furthermore, I designed a structure for this system where we can have a structural system for segmenting and distributing files on data center for storing, with another structural system for re-gathering the files to get as a result the entire original folder ready for the specific user to check it,

While designing the general structure for files fragmentation system, I took into consideration significantly the issue of storage volume, as it is an important thing for the system's investor, and although there is always an inverse relation between security and storage volume, I managed to overcome this with great accuracy, as the important files with small size need information about safety which might be bigger in size than the original folder. Another important issue is the speed of accessing the files by the specified user, as the user doesn't care about what is happening inside the system, in fact he cares only about getting the folder as quick as possible, and that is what I handled while designing this structure. The last issue in this matter is that this sample handles all kinds of files without exception and also processes big-size folders.

جدول المحتويات

جدول الأشكال

الملخـــــــــــــص

إن معظم الأدوات المستخدمة في مجال إدارة تجزئة الملفات وتوزيعها تستند على فكرة أساسية ألا وهي تقسيم الملف إلى عدد معين من القطع وإعادة تجميعها ليحصل في النهاية على الملف ذاته.

من خلال الدراسة المرجعية المتضمنة في هذا البحث، يتبين أن النماذج الرياضية المستخدمة تقوم بالتجزئة والتجميع كما هو مذكور سابقاً. ولكن هذه العملية لم تراعي فكرة الأمان وان الملفات هي عبارة عن بيانات مهمة وحساسة تابعة لبنك معلومات مؤسساتي له وضعه الخاص لذلك قمت بتصميم نموذج رياضي مُعدل عن النماذج الموجودة حيث أضفت متطلبات المؤسسة والقواعد الناظمة لها فلم الجأ إلى التقسيم كيفما اتفق وإنما قمت بتأطيره في إطار بنك معلومات مؤسساتي كما قمت بتصميم هيكلية لهذا النظام حيث يوجد هيكلية لتجزئة الملفات وتوزيعها على مركز تخزين البيانات كذلك هيكلية أخرى لإعادة تجميعها لنحصل في النهاية على الملف كاملا للمستخدم المحدد المسموح له الاطلاع عليه

عند تصميم الهيكل العام لنظام تجزئة الملفات قمت بمراعاة مسألة حجم التخزين بشكل كبير كونها مسألة مهمة بالنسبة لمستثمر النظام وعلى الرغم من أن هنالك علاقة عكسية بين الأمان وحجم التخزين فقد قمت بتلافي هذا الموضوع بقدر كبير من الدقة فالملفات الهامة ذات الحجم الصغير تحتاج إلى معلومات عن الأمان ربما تكون اكبر من حجم الملف الأصلي والمسألة الأخرى التي لا تقل أهمية عن التخزين وهي سرعة الوصول إلى الملفات للمستخدم المحدد حيث أن المستخدم لا يهمه ما يجري داخل النظام ما يهمه فعليا أن يحصل على ملفه بالسرعة المطلوبة وهذا الأمر قمت بمعالجته في تصميم هذه الهيكلية

المسألة الأخيرة في هذا الموضوع هي أن هذا النموذج يعالج مختلف أنواع الملفات دون استثناء كما أنه يعالج الملفات الكبيرة ذات الحجم العالي .

الفصل الأول

مقدمة

الفصـل الأول
مقدمـــــــــة

تتجه معظم المؤسسات لاسيما الحكومية الكبيرة للاعتماد على بنوك المعلومات في أداء مهامها وتحضير البنية التحتية المعلوماتية اللازمة وفقاً لخصوصية المعلومات لدى تلك المؤسسات؛ الأمر الذي قد يتطلب في بعض أنواع المعلومات المخزنة في الملفات ذات الحجم الكبير أو الحساسية العالية إلى تجزئة وتوزيع ملفات تلك البيانات ولكن معظم الأنظمة التي تعمل على تجزئة الملفات تقوم بتقسيم الملف إلى قطع وكأنها تقوم بتقطيع قطعة فاكهة دون أن تراعي طبيعة الملف وخصوصيته وهل هو متاح للجميع ام انه ملف سري ينبغي الحفاظ عليه من الاختراق؟ كما أن مسألة توزيع أو بعثرة الأجزاء هي أيضا مسألة جدا مهمة فعندما نقوم بتوزيع تلك القطع ومن ثم إعادة تجميعها فلن يستطيع أحد غير المخول برؤية الملف كاملا باستعراضه وهنالك نقطة إضافية وهي أن هنالك العديد من التقنيات التي تسمح بتجميع بعض أجزاء الملف وتحصل على الملف الكلي وهذه النقطة حساسة جدا لذلك لا يمكن السماح بمثل هذا الأمر في هذا البحث.

هنالك العديد من الأدوات التي تساعد مستخدم الحاسب على تجزئة الملف وجميعها يقوم على فكرة التقطيع ومن ثم التجميع ولكن لا يراعوا متطلبات المؤسسة وخصوصيتها وهي برامج عامة بالغالب تهدف إلى إعانة المستخدم ويلجا لها كون معظم المواقع على الشبكة العنكبوتية محدودة بحجم معين مثل البريد الالكتروني الذي لا يسمح مثلا برفع ملف اكبر من 25MB لذلك لجأت معظم تلك المواقع إلى إنشاء ما يسمى بالقرص المخصص لكل مستخدم وفيه يضع المستخدم ملفاته وبالغالب يكون الحجم المجاني 5GB فقط وإذا أراد مساحة تخزين اكبر يحتاج إلى شرائها من الموقع ولكن هل هنالك خصوصية في هذا الأمر؟ وإذا اخترق شخص ما هذا الموقع أصبحت بيانات المستخدمين معرضة للخطر والنشر

وإذا قمنا بتعميم هذه الفكرة على مؤسسة حكومية ذات خصوصية معينة وتم اختراق حساباتها فإن النتائج ستكون كارثية لذلك سنلجأ من خلال بحثنا هذا على تلافي الوقوع في محظورات لا تحمد عقباها.

إشكاليـــــة البحث

هنالك عدة مشاكل في تجزئة الملفات واسترجاعها وفقاً لحساسية المعلومات الموجودة فيها وحجمها ونوع الملف المراد تقسيمه.

من خلال المقدمة السابقة هنالك أسئلة مطروحة نجيب عليها من خلال بحثنا هذا

1- ما هي المتطلبات الواجب توافرها في النموذج للوصول للحل المراد؟

2- هل تكفي نماذج تجزئة الملفات الحالية لحل مشاكل الأمان وسرعة الوصول والاسترجاع وتقليل حجم التخزين؟

3- إذا لم يتوافر نموذج يقودنا إلى الحل فهل يمكن تعديل النماذج الموجودة؟

أهـــداف البحث

إن الهدف الأساسي من هذا البحث تكوين نموذج نظري لتقسيم واسترجاع الملفات بطريقة أمنة بناءاً على متطلبات بنك معلومات مؤسساتي من أمان في التخزين وسرعة الوصول للملفات بالإضافة إلى تقليل حجم التخزين قدر الإمكان.

منهجية البحث

قمنا بتعديل النماذج الموجودة سابقاً في تجزئة الملفات واسترجاعها للحصول على نموذج مقترح يراعي بنك معلومات مؤسساتي .

فرضيات البحث

معظم النماذج المستخدمة في تجزئة واسترجاع الملفات تعتمد على فكرة أساسية وهي تقطيع الملف إلى أجزاء دون أن تراعي طبيعة بنك المعلومات المؤسساتي فهنالك ملفات يمكن أن تكون سرية وهنالك أيضاً ملفات يمكن أن تكون متاحة لجميع المستخدمين

مخطط البحث

تم تنظيم البحث كما يلي:

- **الفصل الثاني:** يحتوي هذا الفصل دراسة مرجعية تتضمن

  أولاً: مفهوم بنك المعلومات الصغير (المؤسساتي) .

  ثانياً: طبيعة الملفات وتنسيقها وكيفية تموضعها في القرص الصلب .

  ثالثاً: الأنظمة الموزعة حيث تم توضيح بدايات ظهور أنظمة الملفات الموزعة، وعرض معظم هذه الأنظمة منذ بداية ظهورها حتى وقتنا الحالي.

- **الفصل الثالث:** يحتوي هذا الفصل على شرح مفصل عن المنهجية المقترحة للإجابة عن أسئلة البحث، والنموذج الجديد المقترح لتحقيق عملية تجزئة الملفات وتوزيعها بشكل آمن في بنك معلومات مؤسساتي، والميزات الجديدة التي تحققت بفضل الشكل الجديد للنموذج.

- **الفصل الرابع:** يستعرض هذا الفصل عدداً من الاختبارات لتطبيق الحل المقترح، ومناقشة هذه النتائج.

- **الفصل الخامس:** الخاتمة والآفاق المستقبلية

الفصل الثاني

الدراسة المرجعية

الفصل الثاني
الدراسة المرجعية

سيتم في هذا الفصل التكلم عن دراستنا المرجعية التي تتضمن عرضاً للإطار النظري عن طبيعة الملفات وكيفية توضعها فيزيائياً في القرص الصلب والخوارزميات المستخدمة في هذا الإطار كما سيتم استعراض الأنظمة الموزعة.

من أجل تكامل البحث، تم عرض ثلاثة مفاهيم وهي: كيفية إدارة الملفات وكيفية توضعها في القرص الصلب والخوارزميات المتبعة لتجزئة الملفات، كذلك يتضمن البحث الأنظمة الموزعة كونه يتضمن توزيع الملفات.

أولاً: مفهوم بنك المعلومات الصغير (المؤسساتي)

بنك المعلومات هو قاعدة معلوماتية ومخزن معلوماتي ضخم من المعلومات والبيانات بكافة أشكالها وصورها، مخزنة ومحفوظة بشكل يسهل التعامل معها والبحث عنها بواسطة الحاسوب، ويتألف من مباني ضخمة مكونة من العديد من الخوادم التي تعمل على مدار اليوم 24/24 مع توفر دائم لخدمة الإنترنت والكهرباء. يجهَّز هذا المبنى بأحدث التقنيات مع توفر بنية تحتية مدروسة ومدعومة بالعديد من تقنيات المعلومات ووسائط الاتصالات، وآليات الحماية والأمن التي تتيح للمستخدم استخدام نظم المعلومات والخدمات التي يقدمها.

أنواع بنوك البيانات

تقسم بنوك البيانات عادةً وفقاً لحجوم البيانات فيها وكذلك الوظائف والخدمات التي تقدمها إلى عدة أنواع ومستويات.

- **حسب حجم التخزين:**

  - بنوك بيانات صغيرة: تتكون على الأكثر من 10 مخدمات ووحدة تخزين 10TB، مثل بنوك البيانات المستخدمة لمراقبة تغيرات الطقس.
  - بنوك بيانات متوسطة: تتكون من 10–100 مخدمات ووحدة تخزين 100TB، مثل بنوك البيانات المستخدمة لحجز نطاقات الويب Web Hosting.
  - بنوك بيانات كبيرة: تتكون من أكثر من 100 مخدم ووحدة تخزين أكثر من 100TB، تستخدم مثل هذه البنوك في الهيئات والمؤسسات الحكومية والشركات التي تتعامل مع حجم كبير من المعلومات وكم كبير من العمليات عليها.

- **حسب الوظائف والخدمات:**

  - بنوك بيانات للمعالجة Processing.
  - بنوك بيانات للتخزين Storage.
  - بنوك بيانات كمزود لخدمات التطبيقات Application Service providers (ASP).

أهمية بنوك البيانات:

يمكن الاستفادة من بنوك المعلومات في المجالات التالية:

- إدارة ومعالجة البيانات بسرعة ودقة، خاصة في حالة استخدام بيانات خاصة مثل الصور.
- تكون إمكانية التحكم بالبيانات قوية وآمنة ضد الهجمات والمخاطر.
- مجهزة بنظام إدارة الأزمات، للعمل ضمن ظروف خاصة.
- إمكانية الاستفادة منها في آلية اتخاذ القرار.
- إمكانية تشفير البيانات وتبادلها بشكل آمن.
- إمكانية استضافة مجالات الويب والخدمات والشبكات الافتراضية.

ثانياً : كيفية إدارة الملفات وتوضعها على القرص الصلب

تنسيق الملف[1]:

هو طريقة لتشفير المعلومات داخل ملف الكمبيوتر بحيث يجب أن يكون البرنامج أو التطبيق قادراً على التعرّف على تنسيق الملف من أجل الوصول إلى البيانات داخله.

على سبيل المثال، متصفح ويب قادر على معالجة وعرض ملف في تنسيق ملف HTML بحيث يظهر كصفحة ويب. إذا واجه المتصفح ملف من صيغة أخرى، فقد يحتاج إلى الاتصال بمكون إضافي خاص لمشاهدته، أو قد يسمح لك ببساطة بتنزيل الملف لعرضه في برنامج آخر إذا لم يتمكن من التعرّف عليه.

غالباً ما يُشار إلى تنسيق ملف معين كجزء من اسم الملف بواسطة ملحق اسم الملف، يتبع التمديد بنقطة في اسم الملف ويحتوي على ثلاثة او أربعة أحرف تحدد التنسيق (jpg أو jpeg).

في أنظمة التشغيل Windows يمكنك اختيار عرض لاحقات الملف أو عدم عرضها، ("إخفاء ملحقات أنواع الملفات المعروفة" هو الخيار الافتراضي في مستكشف ويندوز، والذي يمكن عكسه لرؤية ملحقات الملفات).

أما في أنظمة MAC فإن الملفات غالباً لا تحتوي على لاحقة، وهذا بحد ذاته مشكلة عند نسخها لاستخدامها في نظام آخر.

وهنا تجدر الإشارة إلى ملكية الملفات، عادة يجب فتح الملفات من قبل البرنامج الذي تم إنشاؤها بواسطته. وقد لا يتمكن أي شخص بدون ترخيص للبرنامج من فتح الملف على الإطلاق.. و(Adobe (PDF هو خير مثال على ذلك.

يمكن قراءة الملفات النصية من خلال معيار التشفير مثل ASCII أو Unicode. لا يمكن قراءة الملفات الثنائية إلا بواسطة البرامج المعمولة بها، وقد تكون ملكية خاصة، وقد تحتوي بعض الملفات على نص وترميز ثنائي (ملفات RTF..).

في مرحلة ما قد يلجأ المستخدم لضغط الملفات من أجل التخزين أو النقل المحلي أو النقل عن طريق الشبكة وهذا ما يسمى بتخفيض معدل البتات، الذي ينطوي على ترميز المعلومات باستخدام بتات أقل من التمثيل الأصلي.

تستخدم ملفات tar (أرشيف الأشرطة) عادة في Linux / Unix لتجميع مجموعة من الملفات في ملف واحد لتقليل حجمها.

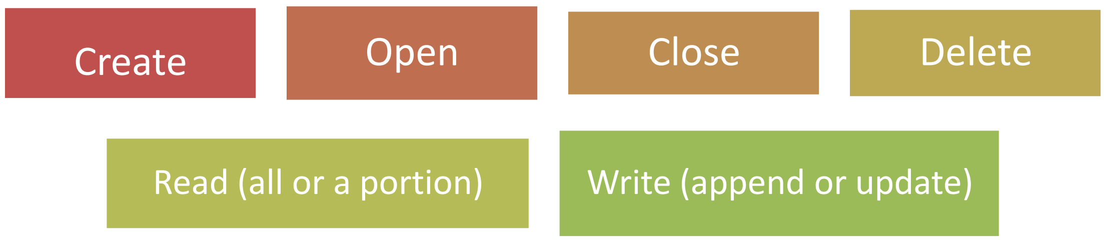
في بيئات مثل Windows يمكن استخدام ملف مضغوط لربط عدة ملفات مع بعضها البعض، مع أو بدون ضغط، أو مع ضغط أكثر أو أقل. توجد أيضاً الأدوات المفيدة لذلك.

في هذه المرحلة يلزمنا أن نتعرّف على بنية الملف وآلية تموضعه في الذاكرة وفي القرص الصلب.

العمليات المستخدمة على الملف:

عندما نتكلم على الملف علينا أن نعرف ما هي العمليات المستخدمة على الملف؟

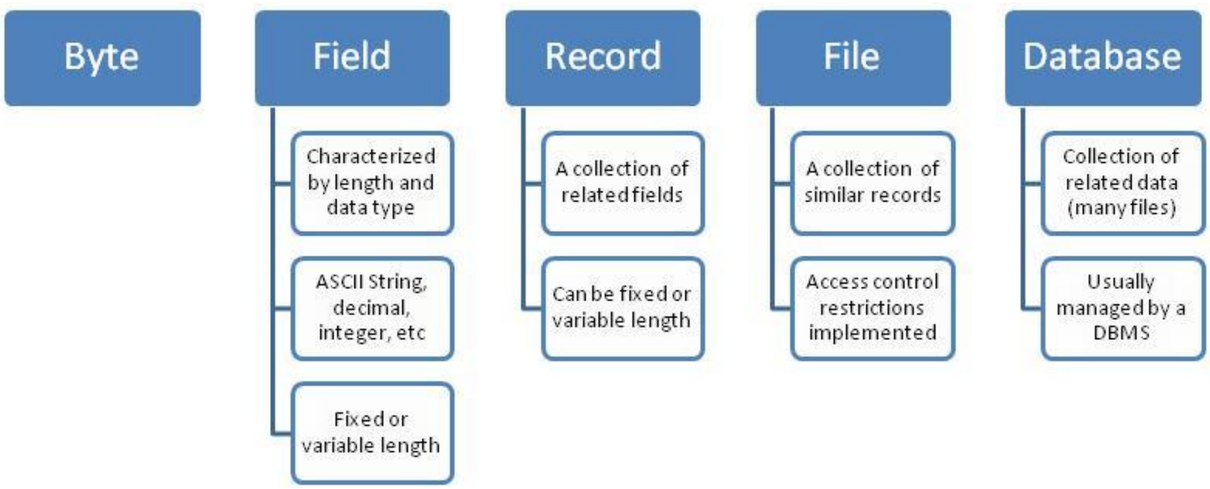
العمليات هي إنشاء ملف وفتحه وإغلاقه وحذفه بالإضافة لقراءة الملف والكتابة عليه الشكل التالي يوضح هذه العمليات.



الشكل 1 العمليات المستخدمة على الملف

هيكل الملف الداخلي:

مما يتكون الملف كبنية داخلية؟



الشكل 2 بنية الملف الداخلي

لغة الآلة تتكون من أصفار ووحدات التي تشكل البايت وبدورها مجموعة من البايتات تشكل الحقل

ومن ثم فان الملفات هي عبارة عن مجموعة من الـ record، وهنالك قيود على الولوج إليها.

عمليات الوصول للملفات:

عملية الوصول للملفات تعمل أساساً على السجلات، ولكن بشكل تجريدي يمكن تطبيقها على مجرد بايت، لذلك يمكن أن نقوم ببعض العمليات على الملفات وهي:

- Retrieve_All: قراءة جميع السجلات في الذاكرة بشكل متسلسل.
- Retrieve_One: وعادة ما ترتبط بشكل متفاعل مع التطبيقات الموجهة نحو المعاملات.
- Retrieve_Next/Previous: يكون الـ record معرّف مسبقاً منطقياً (أي لديه العناوين اللازمة لتلك العملية في الذاكرة) وبشكل متسلسل، وبالتالي هو متعلق بعملية البحث عن عناوين الـ records في الذاكرة.
- Insert_One: قد ينطوي على الوصول العشوائي او الإلحاق بملف ما.
- Delete_One: بعض الروابط أو غيرها من هياكل البيانات قد تحتاج إلى تحديث للحفاظ على تسلسل الأجزاء.
- Update_One: استرداد سجل وتحديث حقل أو أكثر من أجل إعادة كتابة السجل المحدَّث إلى الملف الأمر الذي يؤدي إلى تغير في طول الحقول والسجلات مما يستلزم التلاعب في هياكل البيانات للوصول للغرض المطلوب.
- Retrieve_Few: الحصول على عدد محدد من السجلات وهي مستخدمة في قواعد البيانات وفق معايير اختيار معينة.

نظام الملفات:

في الحوسبة، نظام الملفات File System هو طريقة في نظام التشغيل تستخدم للتحكم في كيفية تخزين واسترجاع وتنظيم وإدارة ملفات الحاسوب والبيانات التي تحتوى عليها تلك الملفات لتسهيل إيجادها واستخدامها، أنظمة الملفات قد تستعمل جهاز تخزين البيانات مثل القرص الصلب، بدون نظام الملفات فإن المعلومات الموجودة في القرص الصلب ستكون كأنها قطعة واحدة من البيانات التي لا يمكن التفريق بينها كذلك لا يمكن معرفة أين بداية أي جزء منها ونهايته وما الجزء التالي له من البيانات، أما باستخدام نظام الملفات يمكن الفصل بين أجزاء المعلومات وإعطاء كل جزء منها اسم وبالتالي يسهل الفصل بين الملفات وإعطاء كل منها معرف ومؤشر لها ، اسم نظام الملفات جاء من أن كل مجموعة من البيانات هي ملف file ولها بناء وقواعد يتم إدارتها بواسطة نظام system فيكون الاسم النهائي file system .

هناك العديد من الأنواع المختلفة لنظم الملفات ، كل واحد منها له بناء وخواص منطقية وسرعة مختلفة وأيضاً مرونة وأمان وحجم مختلف ... الخ . بعض نظم الملفات تم تصميمها للاستخدام مع تطبيقات معينة ، على سبيل المثال نظام ملفات أيزو 9660 تم تصميمه للأقراص البصرية.

نظم الملفات يمكن استخدامها في العديد من أنواع أجهزة ووسائط التخزين ، أشهر أنواع أجهزة التخزين المستخدمة حالياً هي القرص الصلب حيث يقوم نظام التشغيل بالتحكم في كيفية تخزين واسترجاع وتنظيم وإدارة ملفات الحاسوب والبيانات التي تحتوى عليها تلك الملفات لتسهيل إيجادها واستخدامها.

وهنا تجدر الإشارة إلى أن نظام إدارة الملفات (file management system) هو مجموعة برامج النظام التي تقدم خدمات للمستخدمين والتطبيقات المتعلقة باستخدام الملفات عادة الطريقة الوحيدة التي يمكن للمستخدم أو التطبيق الوصول إلى الملفات من خلال نظام إدارة الملفات.

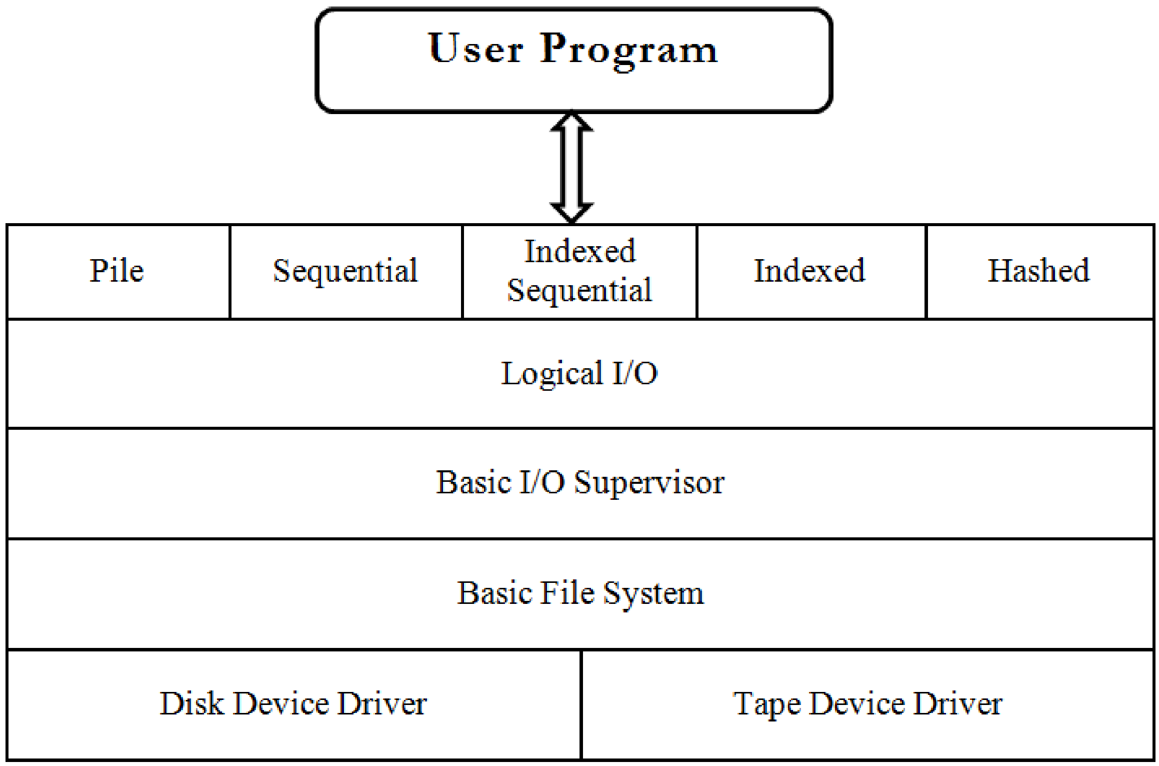
لكي تتم عملية إدارة الملفات بصورة صحيحة يجب مراعاة ما يلي:

- أن تكون بيانات الملف صالحة.
- تحسين الأداء (وقت الاستجابة قصير).
- توفير دعم I/O لمختلف أجهزة التخزين.
- تقليل او إزالة إمكانية فقدان البيانات او تدميرها.
- توفير مجموعة موحدة من إجراءات I/O الروتينية لاستخدام العمليات.
- تقديم الدعم I/O لعدة مستخدمين.

بنية نظام الملفات File System architecture

الأنظمة المختلفة تُنظم إدارة الملفات بشكل مختلف. الفقرة التالية تتضمن شرحاً عاماً عن بنية إدارة الملفات التي تلجأ إليها معظم أنظمة التشغيل.

على أدنى مستوى، فإن برامج تشغيل الأجهزة تقوم بالتواصل مباشرة مع الأجهزة الطرفية أو وحدات التحكم الخاصة بها أو القنوات. برنامج تشغيل الجهاز مسؤول عن بدء عمليات الإدخال / الإخراج على جهاز معين ومعالجة إتمام طلب إدخال / إخراج. بالنسبة لعمليات الملفات فإن الأجهزة النموذجية التي تسيطر عليها هي محركات الأقراص وعادة ما تعتبر برامج تشغيل الأجهزة جزءاً من نظام التشغيل.



File System software architecture

الشكل 3 هيكلية نظام الملفات

من خلال الشكل السابق يتم توضيح المفاهيم التالية

- برامج تشغيل الأجهزة: مسؤولة عن بدء واستكمال طلبات I/O لمختلف الأجهزة الطرفية.
- نظام الملفات الأساسية (المادية) I/O على مستوى نظام التشغيل) تبادل كتل البيانات دون أن يفهم المحتوى.
- I/O أساسي مراقب (جزء من نظام التشغيل): يحافظ على هياكل التحكم في أجهزة I/O والجدولة وحالة الملف.
- I/O منطقي هدفه الأساسي الوصول إلى السجلات ويحتفظ بمعلومات أساسية عن الملفات (مؤشرات وما إلى ذلك).

هنالك بعض المعايير لتنظيم الملفات منها:

- وقت الولوج قصير Short access time
- سهولة التحديث
- الاقتصاد في التخزين

- سهولة الصيانة
- الموثوقية

هذه المعايير يمكن أن تكون متضاربة فمثلاً الاقتصاد في التخزين مع تكرار البيانات والتكرار يزيد من سرعة الولوج إلى البيانات مع الموثوقية الأمر الذي يؤدي إلى زيادة في التخزين.

التوابع المستخدمة لإدارة الملفات

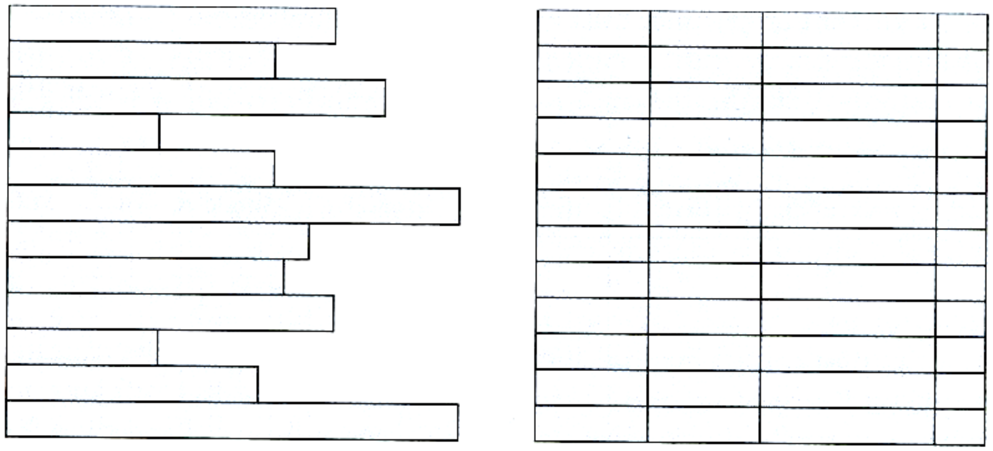
تتفاعل برامج المستخدمين والتطبيقات مع نظام الملفات عن طريق الأوامر الخاصة بإنشاء الملفات وحذفها وتنفيذ العمليات على الملفات. قبل تنفيذ أي عملية، يجب أن يقوم نظام الملفات بتحديد الملف المحدد وتحديد موقعه. وهذا يتطلب استخدام نوع من الدليل لوصف موقع الملفات بالإضافة إلى سماتها. وبالإضافة إلى ذلك، فإن معظم النظم المشتركة (shared systems) تفرض التحكم في النفاذ إلى المستعملين أي انه لا يسمح إلا للمستخدمين المرخص لهم بالوصول إلى ملفات معينة بطرق معينة. يتم تنفيذ العمليات الأساسية التي يمكن أن يقوم بها المستخدم او التطبيق على ملف على مستوى السجل.

استخدام تطبيقات معنية مع سجلات I/O يتم على أساس كتلة. وبالتالي يجب أن يتم حجز مساحة للسجلات الخاصة بالملف وحظر الكتابة عليها وفتحها بعد الإدخال. لدعم كتلة I/O من الملفات، هناك حاجة إلى عدة وظائف منها إدارة التخزين الثانوي وإدارة مساحة التخزين الحرة ، إدارة التخزين الثانوي تنطوي على تخصيص الملفات وتوزيعها إلى كتل مجانية على مساحة التخزين الثانوي، في حين أن إدارة مساحة التخزين الحرة تساعد في معرفة ما الكتل المتاحة للملفات الجديدة او النمو في حجم الملفات الموجودة مسبقاً. يعتمد التحسين على بنية الملفات وأنماط الوصول.

ومن خلال الشكل (3) نجد أن التقنيات الشائعة لهيكلية إدارة الملفات هي:

- الملف المعتمد على تابع بيل **PILE:**

يتم جمع البيانات في الترتيب الذي يصل ويتكون كل سجل من رشقه (burst) واحدة من البيانات، وقد تحتوي السجلات على تشكيلة متنوعة بشكل كبير من الحقول وأطوال الحقل ويجب أن يكون لكل حقل وصفاً ذاتياً.



الشكل 4 توضع السجلات باستخدام pile, sequential

➢ الملف المتسلسل Sequential File :

- شكل ثابت يستخدم للسجلات.
- طول وموقع كل حقل معروف، يتطلب أن يتم تخزين قيم الحقول فقط.
- الحقل الأول من كل سجل هو الحقل الرئيسي، والسجلات يتم تخزينها في تسلسل keys (يمكن أن يكون الاختلافات).
- ليست جيدة للتطبيقات التفاعلية مع الاستعلامات للسجلات الفردية أو التحديثات.
- إدراج السجلات هو أيضاً غير فعّال، تتطلب دورات من batch merges.
- يمكن تنفيذها من خلال تنظيم ملف كقائمة مرتبطة.

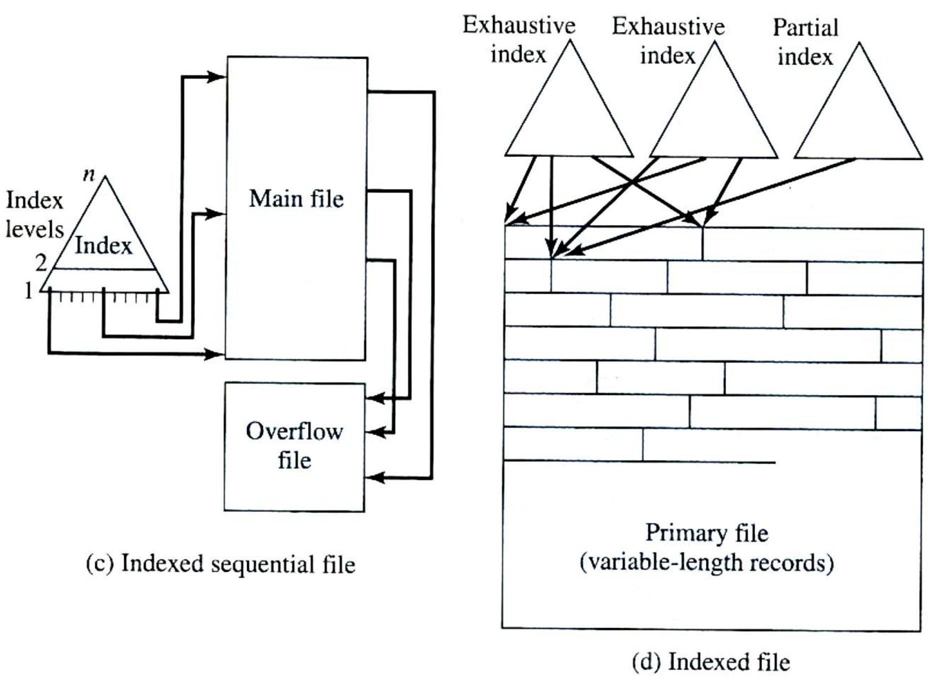
➢ الملف المتسلسل المفهرس Indexed Sequential File:

- يستخدم فهرساً لدعم الوصول العشوائي.

- يتطلب ملف تجاوز (overflow) للتعامل مع الإضافات.
- يستخدم الفهرس نفس مفتاح الملف الرئيسي، ويحتوي على مؤشر للملف، يحسن إلى حد كبير وقت البحث.
- يمكن أن يكون مؤشرات متعددة المستويات للحصول على سرعة أكبر للوصول.

➢ الملف المفهرسIndexed File:

- يستخدم فهرساً لدعم الوصول العشوائي.
- يحافظ على مؤشرات متعددة لكل نوع من الحقول التي قد تكون موضوع البحث.
- يتم الوصول إلى السجلات فقط من خلال مؤشراتها.
- يمكن استخدام الحقول ذات الطول المتغير.
- يمكن استخدام الفهرس الشامل والمؤشر الجزئي.



(c) Indexed sequential file

(d) Indexed file

الشكل 5 توضع السجلات باستخدام indexed,indexed sequential

➢ ملف هاش Hashed File:

تعتمد تقنية Hashed على قيمة رئيسية تقوم بالذهاب مباشرةِ إلى السجل على القرص. تعطي كفاءة في المقام الأول للسجلات ذات الطول الثابت وعمليات Retreive_One التي ذكرت سابقاً.

بالمجمل، الجدول التالي يبين مقارنة عن أداء هذه التقنيات من حيث المساحة والتحديث والاستعادة:

File method	space		Update		retrieval		
	Attributes		Record size		Single record	subset	exhaustive
	variable	fixed	Equal	greater			
pile	A	B	A	E	E	D	B
sequential	F	A	D	F	F	D	A
Indexed sequential	F	B	B	D	B	D	B
indexed	B	C	C	C	A	B	D
hashed	F	B	B	F	B	F	E
A: جيد جدا وهو مناسب لهذا الغرض =O(r) B: جيد =O(o*r) C: مقبول =O(r*log(n)) D: يحتاج بعض الإضافات =O(n) E: يحتاج للكثير من الإضافات =O(r*n) F: غير ملائم لهذا الغرض =O(n^>1)				بحيث r: حجم النتيجة n: عدد السجلات في الملف o: عدد السجلات الفائضة			

الجدول 1 مقارنة بين الاجرائيات pile,sequential,indexed,hashed

دليل الملف File Directory :

تعني حاوية افتراضية ضمن نظام الملفات، والذي يمكن أن يضم الملفات والمجلدات ويقوم بتنظيمها. هو تقريباً الملف نفسه يحتوي على المعلومات التالية:

- اسم الملف ونوعه والمنظمة التي ينتمي إليها.
- حجم، عنوان البداية، حجم المستخدمة/المخصصة.
- المالك، والوصول إلى المعلومات، والإجراءات المسموح بها.

- تاريخ الإنشاء، الخالق، مشاركة الوصول، آخر وصول، آخر تعديل، آخر النسخ الاحتياطي،

حق الوصول للملف (سماحيات الوصول للملف):

يمنح الأفراد أو مجموعات المستخدمين حقوقاً معينة للملفات أو الدلائل حسب التسلسل الهرمي التالي:

- ✓ **None**: لا نعرف شيء عن وجود الملف أو دليله.
- ✓ **Knowledge** : يمكن للمستخدم تحديد وجود الملف ومالكه.
- ✓ **Execution** : يمكن للمستخدم تحميل وتنفيذ البرنامج ولكن لا يمكن النسخ.
- ✓ **Read** : يمكن للمستخدم قراءة الملف لأي غرض.
- ✓ **Append**: يمكن للمستخدم إضافة بيانات إلى الملف ولكن لا يمكنه التعديل أو الحذف.
- ✓ **Update** : يمكن للمستخدم التعديل، الحذف والإضافة إلى بيانات الملف (ربما بشكل متدرج).
- ✓ **Change protection** : يمكن للمستخدم تغيير حقوق الوصول الممنوحة للمستخدمين الآخرين.
- ✓ **Deletion** : يمكن للمستخدم حذف الملف من نظام الملفات والقيام بأي شيء آخر.

الوصول المتزامن للملف:

عندما يتم منح الوصول (إلحاق أو تحديث ملف) لأكثر من مستخدم واحد يجب على نظام إدارة الملفات أو نظام الملفات فرض الانضباط من خلال نهج القوة الغاشمة (brute–force) وهو السماح للمستخدم لقفل الملف بأكمله عندما يتم تحديثه بكلمات أخرى هو قفل السجلات الفردية أثناء التحديث.

توزيع الملف أو التخصيص File Allocation

الفكرة الرئيسية وراء التخصيص أو التوزيع هي الاستخدام الفعال لمساحة التخزين والوصول السريع للملفات.

هنالك نوعين للتخصيص وهما توزيع مسبق Preallocation أو تخصيص ديناميكي Dynamic فما الفرق؟

- Preallocation

يتم الإعلان عن حجم الملف الأقصى في وقت إنشاء الملف يكاد يكون من المستحيل تقدير موثوق لمعظم التطبيقات ومن المحتمل جدا الإسراف.

- Dynamic

تخصيص مساحة لملف في أجزاء حسب الضرورة. بالمقابل لتوزيع الملف تظهر قيود لتوزيع الملف منها حجم الجزء (Portion Size).

عند التكلم عن حجم الجزء المراد تخصيصه علينا أن نأخذ بعين الاعتبار المقايضة من خلال المساحات المستمرة في فضاء التخزين الأمر الذي يزيد من الأداء وسرعة الوصول إلى البيانات، والعكس عند وجود عدد كبير من الأجزاء الصغيرة فإن ذلك يؤدي إلى زيادة في حجم الجداول اللازمة لإدارة معلومات التخصيص وبالتالي بطيء الأداء والتقليل من سرعة الوصول إلى البيانات .

وتتمثل إحدى الوظائف الهامة لنظام الملفات في إدارة المساحة على وحدة التخزين الثانوية، التي تتضمن تتبع كل من كتل الأقراص المخصصة للملفات والكتل الحرة المتاحة للتخصيص. المشاكل الأساسية وراء التخصيص هي الاستخدام الفعال لمساحة الملف والوصول السريع للملفات.

إدارة كتل القرص هي مشكلة مألوفة في إدارة الذاكرة الرئيسية. ولكن التخزين الثانوي يقدم مشكلتين إضافيتين وهما البطيء في زمن الوصول للقرص وعدد الكتل الكثيرة من أجل التعامل معها.

وعلى الرغم من ذلك، فهناك ثلاثة تقنيات توزيع مستخدمة على نطاق واسع وهي: المستمرة Contiguous والمتصلة أو المرتبطة Linked Allocation or Chained والمفهرسة Indexed Allocation [2].

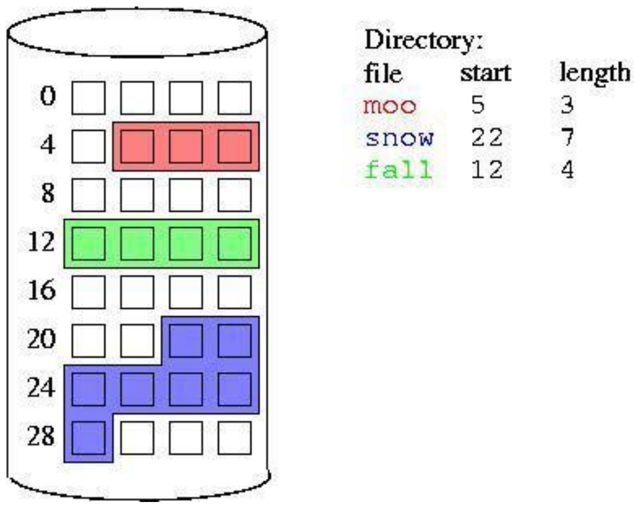
تقنية التوزيع المستمرة (Contiguous) تنتمي إلى النوع الأول Preallocation في حين تنتمي كلاً من تقنيات التوزيع المتصلة والمفهرسة إلى النوع الثاني Dynamic allocation. لكل طريقة محاسنها ومساوئها.

أثناء مناقشة استراتيجيات التخصيص هذه، سنعتبر أن الملف عبارة عن سلسلة من القطع جميع عمليات (الإدخال / الإخراج) على القرص تحدث في أجزاء الكتل والتحويل من السجلات المنطقية إلى كتل فعلية يتم بواسطة البرنامج.

- **تقنية التوزيع المستمرة:**

في تقنية التوزيع المستمرة يتم تعيين الملفات إلى المناطق المستمرة للتخزين الثانوي. يحدد المستخدم مسبقاً حجم المساحة المطلوبة للملف ليتم إنشاؤه. إذا لم تكن تلك المساحة المتجاورة متاحة، فلا يمكن إنشاء الملف.

وتتمثل إحدى مزايا هذه التقنية في أن جميع السجلات المتتابعة للملف تكون عادة متجاورة مادياً مع بعضها البعض. وهذا يزيد من سرعة الوصول إلى السجلات. مما يعني أنه إذا كانت السجلات متناثرة على القرص فإن وصولها سيكون أبطأ وتكون عملية الوصول إلى الملف الذي تم تخصيصه بشكل متواصل أمر سهل إلى حد ما، من الضروري الاحتفاظ بعنوان البداية لكل ملف وحجمه ، إدخال الدليل لكل ملف يشير إلى عنوان بداية الكتلة وحجم وطول هذا الملف. والشكل التالي يوضح تقنية التوزيع المستمرة.



الشكل 6 تقنية التوزيع المستمرة

➢ **التخصيص غير المستمر الديناميكي:**

لأن الملفات تميل إما إلى النمو او التقليص ولأن المستخدمين نادراً ما يعرفون مسبقاً توزيع ملفاتهم وفيما إذا كانت الملفات متجاورة أم لا. يتم استبدال أنظمة تخصيص التخزين بنظم تخصيص تخزين أكثر ديناميكية وغير متجاورة.

1. التخصيص المرتبط:

التخصيص المرتبط يحتوي عدد قليل من وحدات البايت بحيث يكون لكل كتلة من القرص عنوان للكتلة التالية. يحتوي الدليل على مؤشر إلى الكتل الأولى (والأخيرة) من الملف. كما هو مبين بالشكل 8.

مميزاتها:

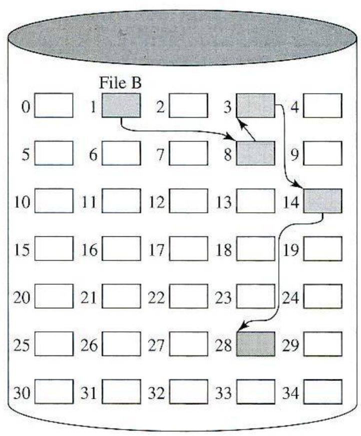
(1) بساطته (2) لا يلزم ضغط القرص. وبسبب الطبيعة غير المتجانسة للتخصيص، فإن الربط لا ينتج عنه أي تجزؤ خارجي للقرص. أي كتلة القرص على المساحة حرة يمكن استخدامها لتلبية الطلب، لأن كل الكتل

ترتبط معاً. ولا توجد حاجة أيضا للإعلان حجم الملف في التخصيص المرتبط أثناء إنشائه. لذلك من الممكن أن يستمر نمو الملف وهناك أيضا كتل مجاني. وبالتالي، فإنه ليس من الضروري أبداً ضغط القرص.

عيوبها:

الوصول البطيء المباشر لأي كتلة قرص – لمعرفة كتلة N^{th} من ملف، يجب أن نبدأ في بداية هذا الملف واتباع تسلسل المؤشرات حتى نصل إلى كتل N^{th} ، هنالك تسلسل للمؤشرات بين كتلة البداية و النهاية، وبالتالي يتطلب كل ملف مساحة أكبر.

الموثوقية – بما أن كتل الأقراص مرتبطة بواسطة مؤشرات، يمكن أن يؤدي مؤشر تالف واحد إلى جعل الآلاف من كتل القرص غير قابلة للوصول، تتعامل بعض أنظمة التشغيل مع هذه المشكلة من خلال تخزين مؤشرات في ملف مخصص وإصدار نسخ مكررة منه، والفكرة هي لنسخ قائمة المؤشرات في الذاكرة الرئيسية، وبالتالي تسهيل الوصول بشكل أسرع إلى كتل القرص. يتم استخدام التكرار من ملفات المؤشرات لجعل ذلك أكثر أماناً.



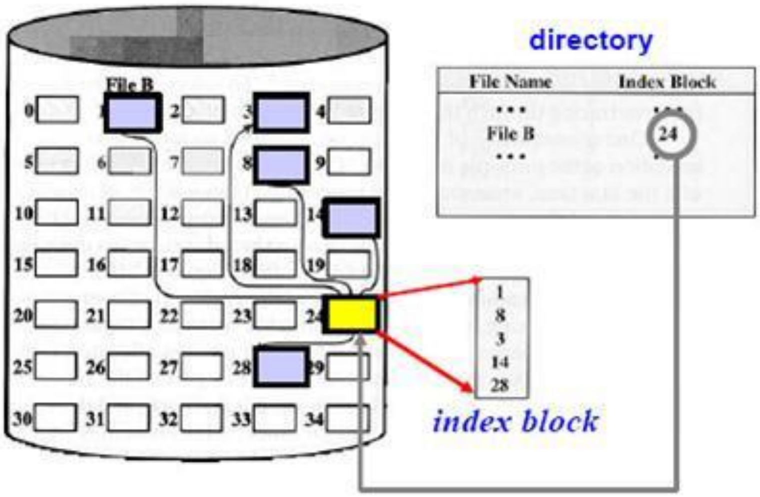
File Allocation Table

File Name	Start Block	Length
•••	•••	•••
File B	1	5
•••	•••	•••

الشكل 7 التخصيص غير المستمر الديناميكي

2. التخصيص المفهرس:

أحد العيوب مع تقنية التخصيص المرتبطة هو أنه لا يدعم الوصول المباشر للكتل المتناثرة في جميع أنحاء القرص. يتم حل هذه المشكلة من خلال تخصيص مفهرس عن طريق وضع كافة المؤشرات معا في كتلة الفهرس كما في الشكل التالي



الشكل 8 التخصيص المفهرس

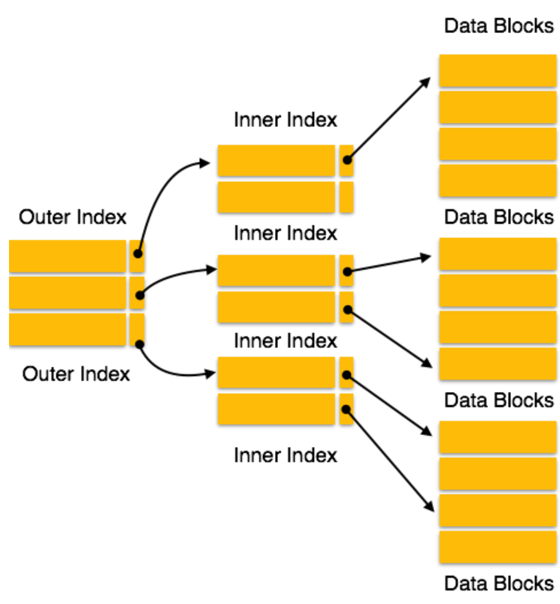
مميزاتها:

تشمل ميزة تخصيص ملف مفهرس عدم وجود تجزئ خارجي وكفاءة الوصول العشوائي. وعلاوة على ذلك، الفهرسة تُكوّن بسهولة خريطة حول كتل القرص السيئة. فهرسة المساحة الحرة يمكن إنجازها عن طريق خريطة البتات (bit map)

عيوبها:

العيب الأساسي في الفهرسة هو عدد مرات الوصول إلى القرص لاسترداد عنوان كتلة الهدف على القرص. يمكن تخفيض هذه النفقات العامة من خلال الحفاظ على بعض كتل مؤشر في الذاكرة. العيب الثاني هو أن التخصيص المفهرس يتطلب مساحة كبيرة لحفظ المؤشرات.

معظم الملفات صغيرة عموماً مما يثير مسألة حجم كتلة الفهرس. يجب أن يحتوي كل ملف على كتلة فهرس، وبالتالي، يجب أن يكون أصغر في الحجم. ولكن إذا كانت كتلة الفهرس صغيرة جداً، فإنه لن يكون قادراً على الاحتفاظ بما يكفي من المؤشرات لملف كبير، وبالتالي يجب أن يكون هناك آلية مناسبة للتعامل مع هذه المسألة فكان هنالك خيار بإضافة مستويين من الفهارس، الشكل التالي يوضح ماسبق.



الشكل 9 التخصيص المفهرس متعدد المستويات

هناك بديل يستخدم في نظام Unix وهو الجمع بين العناوين المباشرة والمفهرسة لتوفير مساحة وتوفير إمكانية الوصول السريع إلى الملفات الصغيرة. في Unix يتم الاحتفاظ بالمؤشرات الـ 15 الأولى من كتلة المؤشر في دليل الجهاز. أول 10 مؤشرات تحتوي على عنوان مباشر من 10 كتل للبيانات على القرص. وتشير 3 مؤشرات التالية إلى كتل غير مباشرة. في الجدول التالي ملخص لما سبق ذكره.

FILE ALLOCATION METHOD	**CONTINUOUS**	**CHAINED**	**INDEXED**	
Preallocation?	necessary	possible	**POSSIBLE**	
Fixed or variable size portions?	variable	Fixed blocks	Fixed block	**VARIABLE**
Portion size	large	small	small	**MEDIUM**
Allocation frequency	once	Low to high	high	**LOW**
Time to allocate	medium	long	short	**MEDIUM**
File allocation table size	One entry	One entry	large	**MEDIUM**

الجدول 2 مقارنة بين تقنيات التخصيص

توزيع الفهرسة هو أمر أكثر تعقيداً. إذا كانت كتلة الفهرس موجودة بالفعل في الذاكرة، فيمكن إجراء الوصول مباشرة. ومع ذلك، يتطلب الحفاظ على كتلة الفهرس في الذاكرة مساحة أكبر فإذا كانت مساحة الذاكرة غير متوفرة فسوف نضطر إلى قراءة كتلة الفهرس أولاً ثم قراءة كتلة البيانات المطلوبة. وبالتالي يعتمد أداء التخصيص المفهرس على بنية الفهرس وحجم الملف وموقع الكتلة المطلوبة.

بعض الأنظمة تجمع بين تقنية التخصيص المتجاورة وتقنية التخصيص المفهرس باستخدام تخصيص متجاور للملفات الصغيرة (من 3 إلى 4 كتل) وتتحول تلقائياً إلى تخصيص مفهرس إذا كان الملف ينمو بشكل كبير. وبما أن معظم الملفات صغيرة، والتخصيص المتجاور فعال للملفات الصغيرة، يمكن أن يكون متوسط الأداء جيد جداً.

خلاصة: **معظم الفقرات السابقة تتكلم عن كيفية تموضع الملفات وتقسيمها على القرص الصلب بصورة فيزيائية وهي معلومات جداً مهمة لفهم آلية إدارة الملفات وتقسمها وتوزعها بالإضافة إلى صلاحيات وقيود الوصول إليها في معظم أنظمة التشغيل، الفقرات اللاحقة تتناول إدارة الملفات وتقسيمها وتوزيعها منطقياً عن طريق الشبكة.**

مقدمة في إدارة الملفات وتقسيمها وتوزيعها منطقياً عن طريق الشبكة:

معظم المستخدمين يلجئون إلى الانترنت لتحميل ملفات مختلفة الأنواع لهدف محدد أو أنه بحاجة إلى رفع ملف إلى بريد إلكتروني ما أو موقع معين لتحقيق فائدة معينة، مع العلم أن هنالك قيود على عملية الرفع والتحميل وهي أن الملفات ذات الحجم الكبير لا يمكن رفعها مباشرة أو تنزيلها لذلك فهو بحاجة إلى برامج معينة تقوم بتجزئتها مثل (bit torrent, download manager...) دون أن يعلم المستخدم ما يحصل فهو يريد الملف كاملاً بغض النظر عن الطريقة والآلية التي يتم بها ذلك.

التحميل والرفع (Download–Upload)

نقل الملفات المجزأة[3] (المعروف أيضاً باسم نقل الملفات متعدد المصادر أو نقل الملفات المحتجزة) هو النقل المنسق لملف كمبيوتر مصدره مخدمات متعددة إلى وجهة واحدة. ويمكن تطبيقها كذلك عند تحميل نفس الملف من نفس الخادم في أجزاء مختلفة. يقوم برنامج الكمبيوتر بتحميل (استرجاع) أجزاء مختلفة من الملف من مصادر مختلفة في وقت واحد، ويتم جمع الملف على جهاز تخزين البيانات في كمبيوتر الوجهة.

ربما تكون بداية استخدام التحميلات المجزأة تعود إلى وكالة ناسا، كذلك أنظمة الملفات القائمة على الشريط المغناطيسي. ومع ذلك، خلال الستينيات إلى الثمانينيات من القرن المنصرم كان هناك الكثير من التجارب مع عملية تحميل الملفات، وتحميل (وتزامن) البيانات عبر عرض النطاق الترددي ووصلات الاتصالات المقيدة من قبل معظم مستخدمي الكمبيوتر المركزية الكبيرة. وبالتالي فإن أصول التحميل المجزأ ليست واضحة تاريخياً. ومن المعلوم أن بعض بعثات ناسا تستخدم نوعا من تقنية التحميل المجزأة.

إن أول نظام p2p (peer to peer) أي تسليم المحتوى الكبير الذي نفذ نوعاً من تكنولوجيا التحميل المجزأ تم اختراعه بالإضافة للبروتوكول الخاص به في عام 1999 من قبل جوستين شابويسك وتم بيعها آنذاك إلى اوبينكولا، التي أصدرت البرنامج بموجب ترخيص.

وهناك الكثير من المصطلحات المستخدمة في تكنولوجيا التحميل المجزأ أصلها من سوارمكاست، مع تورنت كونها المساهم المهم الآخر الوحيد في الشروط المستخدمة.

تأثير الشبكة

تم تصميم معظم شبكات IP للمستخدمين للتنزيل(download) أكثر من التحميل(upload) وعادة ما تكون النسبة المتوقعة (download: upload) 3: 1 أو أكثر.

عانت معظم شركات تزويد خدمة الانترنت من ضيق الحزمة وذلك عند استخدامها من قبل 20٪ فقط من قاعدة المستخدمين لتقنية التحميل المجزأة لدرجة انه أصبح من الضروري إعادة برمجة أجهزة التوجيه وإعادة النظر في تصميم الشبكات.

فوائد الشبكة

الملفات الكبيرة يمكن أن تكون متاحة بكفاءة لكثير من المستخدمين الآخرين من قبل شخص ليس لديه عرض النطاق الترددي الكبير لـ upload . يعلم معظم مزودي خدمة الإنترنت التعامل مع تكنولوجيا التحميل المجزأ ولكن التكيف يعني النشر الإلزامي لحركة المرور TCP/IP .

محدوديات الشبكة

تكنولوجيا تقسيم الـ download لا يمكن أن تحل جميع مشاكل الـ download بطريقة سحرية فهناك قيود رياضية على فعالية التكنولوجيا.

سلامة البيانات

يمكن للتطبيقات البسيطة جداً التي تعتمد على تكنولوجيا التحميل المجزأة أن تؤدي في كثير من الأحيان إلى تلف الملفات، حيث لا توجد في كثير من الأحيان طريقة لمعرفة ما إذا كانت جميع المصادر تقوم بالفعل بتحميل شرائح من نفس الملف . هنا عنصر الأمان مغيب تماماً بالنسبة للمصادر التي يؤُخذ منها الملف.

وعادة ما يفضل استخدام خوارزميات مثل MD5 (Message–Digest 5)[4] خوارزمية خلاصة الرسالة (تقوم على اختزال رسالة ذات طول متغيّر إلى طول ثابت هو 128 بت بغضّ النظر عن طولها الأصليّ) في تكنولوجيا التحميل المجزأة لضمان سلامة البيانات، وهنالك خوارزميات أخرى مستخدمة للتأكد من صحة البيانات.

الـ upload المجزئ

على الرغم من أنه فيما يتعلق بتورنت وغيرها من بروتوكولات نقل الملفات الموزعة لا يوجد فرق بين التحميل والتنزيل (كما يمكن للعملاء القيام به على حد سواء) أي أنه لا يمكن التمييز بين العميل والخادم.

تعتبر نظم الاتصالات القائمة على قطاع الفضاء هي الحالات الوحيدة المعروفة على نطاق واسع اعتمدت على تكنولوجيا التحميل المجزأة. ويرجع ذلك أساساً إلى عرض النطاق الترددي المحدد والقيود الأخرى على قطاع الفضاء.

واستخدمت نظم الإرسال والاستقبال الفضائي المباشرة إلى أنظمة الاشتراك في المنازل التي تم نشرها في أوروبا وأمريكا الشمالية نهجاً لتطوير البرمجيات على أجهزة العملاء عن طريق إرسال بضع بايتات فقط في وقت واحد (2k أو أقل) على مدى فترة طويلة من الزمن. عموماً هذه الطرق للتحميل المجزأة هي ملكية خاصة وذات صلة بأمن بطاقة SIM وآلية الاشتراك.

من أشهر الأدوات التي تقوم بعملية التحميل وتجزئة الملفات ورفعها هو البت تورنت .

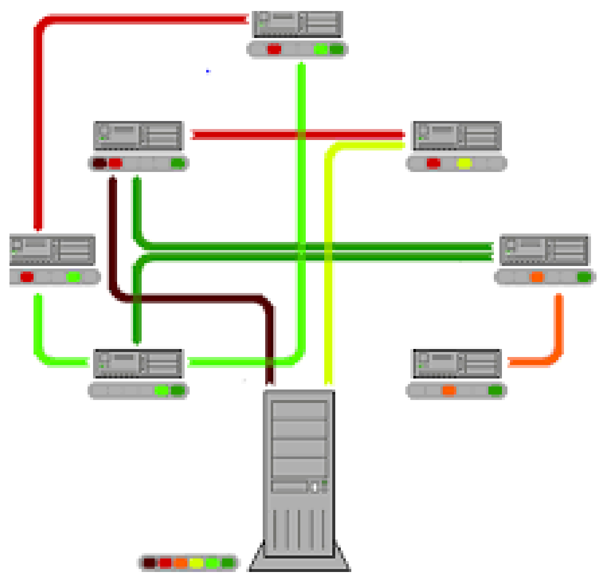
بت تورنت (Bit Torrent)

مصطلح يطلق على أحد بروتوكولات مشاركة الملفات عبر الانترنت والمسماة الند للند أو النظير للنظير أو ما يعرف باللغة الإنجليزية (Peer–to–Peer) ، ويسمح بروتوكول بت تورنت بتبادل الملفات بين المستخدمين دون وجود وسيط عدا المتتبع (Tracker) وهو برنامج عادة ما يكون مستضاف على مخدم (Server) والذي يتكفل بتنسيق عملية الاتصال ما بين النظراء (المستخدمين). كل نظير يقوم بتحميل (Download) البيانات وفي نفس اللحظة يقوم برفع (Upload) البيانات إلى النظراء الآخرين.

في ابريل (نيسان) من العام 2001 قام المبرمج برام كوهين (Bram Cohen) بتصميم البروتوكول؛ وتم تطبيقه لأول مره في 2 يوليو (تموز) 2001 وهو مدعوم من شركة كوهين (Bit Torrent, Inc) واللغة المستخدمة في برمجة البروتوكول هي لغة بايثون (Python) .

كيفية عمل البروتوكول:

يسمح برتوكول البت تورنت بتبادل الملفات كبيرة الحجم عبر الإنترنت دون التعرض لمشاكل كثرة الطلبات والضغط التي تتعرض لها المخدمات العادية، كما أنه يسمح للأجهزة ذات الموارد الضعيفة (كالهواتف الذكية مثلاً) بالمشاركة في عملية تبادل البيانات ذات الحجوم الكبيرة.



الشكل 10 بت تورينت

في الشكل السابق الأشرطة الملونة تحت كل من العملاء السبعة تمثل الملف، فكل لون يمثل قطعة واحدة، يتم توزيع هذه القطع بشكل عشوائي بين العملاء، كما أن المزود الأولي للملف (العميل الأكبر أسفل الشكل) يحتاج فقط لإرسال الملف مرة واحدة فقط، مبعثراً قطعه بين العملاء، ثم يكمل بقية العملاء عملية التبادل فيما بينهم.

تقوم فكرة هذا البروتوكول على تقسيم الملف الكبير إلى عدد محدد من الأجزاء الصغيرة المتساوية في الحجم تسمى قطع(Pieces) ، ولها حجوم قياسية (32KB, 64KB, 128KB, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB)، كما أن هذه القطع بدورها محددة بأقسام أصغر يمثل كل قسم 16KB .

ثم يتم توزيع هذه القطع بين المستخدمين بشكل عشوائي، وكلما زاد عدد المستخدمين الذين يتبادلون الملف نفسه كلما زادت الفرص بالحصول على سرعة أكبر وزمن أقل في الحصول على الملف كاملاً.

لا بد من وجود وسيط بين المستخدمين، يقوم على تنسيق عملية التبادل بين المستخدمين، ويسمى المتتبع (Tracker)، ويعمل المتتبع على تسجيل معلومات عن عدد المستخدمين الكلي النظراء(Peers)، عدد

المستخدمين الذين يملكون الملف كاملاً (مزودو الملف(Seeders))، عدد المستخدمين الذين مازالوا يحملون الملف ولم يكتمل (المُحمّلونLeechers) ، عدد مرات اكتمال تحميل الملف، كما يقوم المتتبع بتسجيل عناوين النظراء ليسهل عليهم الاتصال ببعضهم.

ليكن ملف بحجم 3520MB، في البداية يجب إنشاء ملف بلاحقة torrent. باستخدام أحد البرامج المختصة بالتعامل مع هذه التقنية مثل µTorrent وأثناء إنشائه يتم تحديد حجم القطعة وليكن 4 MB، عندئذٍ سيجزأ الملف إلى 880=3520/4قطعة وكل قطعة تتألف من 256=16/(4*1024) قسماً.

لو كان حجم الملف 3521MB عندئذٍ: 880.25=4/3521. إذاً سيجزّأ الملف إلى 881 قطعة وكل قطعة بحجم 4 MB مما يؤدي إلى زيادة حجم التحميل بمقدار 4 MB تقريباً أي أنك ستقوم بتحميل 3524 MB ولكن الملف الأصلي يبقى كما هو دون أي تعديل. وأيضاً أثناء إنشاء الملف يجب تحديد عنوان المتتبع.

للبدء بمشاركة هذا الملف يجب توزيع الملف ذو اللاحقة torrent. على المستخدمين الراغبين بالحصول على الملف الأصلي، يمثل المستخدم الذي يملك الملف الأصلي في البداية المزود الأولي للملف(Initial Seeder) ، ثم يبدأ بقية المستخدمون بالاتصال به وبدء عملية التحميل، وكل منهم في هذه الحالة يسمى 'مُحمِّل (leecher) ' ويتم توزيع القطع بشكل عشوائي بين المستخدمين مما يسمح لكل مستخدم بتحميل القطع غير المتوفرة لديه من مستخدم آخر يملكها حتى لو لم يكن كلا المستخدمين قد أكمل تحميل الملف كاملاً. عند اكتمال تحميل الملف لدى أحد المستخدمين يتحول دوره من مُحمِّل Leeacher إلى مزود للملفSeeder ، وتقاس "صحة" ملف التورنت بعدد المزودين المشتركين في الحشد الكلي(Swarm) أي بعدد نسخ الملف الكاملة المتوفرة.

تسمح هذه التقنية بنشر نسخ كثيرة من الملفات الكبيرة خلال وقت قصير وبدون ضغط على المزود الأول للملف (الذي ربما يقوم بتوزيع الملف لمرة واحدة فقط)، كما يمكن أن يعيش ملف التورنت لسنوات طالما هناك من يزوده.

ملف التورنت

لمشاركة ملف أو عدة ملفات باستخدام بروتوكول بت تورنت، لا بد من إنشاء ملف بلاحقة torrent. مثلاً (somefile.torrent) يسمى هذا الملف: ملف التورنت أو اختصاراً التورنت.

يحتوي هذا الملف على قسمين أساسيين، الأول هو قسم الإعلان Announce وفيه يحفظ عنوان المتتبع، والثاني هو قسم المعلومات Info، ويحتوي على أسماء الملفات وحجمها، وعدد القطع الكلي، بالإضافة إلى هاش

(Hash) خاص بكل قطعة باستخدام التابع SHA-1 (خوارزمية SHA-1 أفضل الخوارزميات التي تستخدم نظام البعثرة، وهي تستخدم في الكثير من التطبيقات الأمنية ومنها OpenPGP وهي تعتمد على فكرة التشفير باستخدامMD5 التي سبق ذكرها)، البرنامج العميل Bit Torrent يبحث في هذين القسمين لدى إضافة ملف التورنت إليه ليبدأ عملية المشاركة.

يقوم العميل بعد الانتهاء من تحميل كل قطعة بالقيام باختبار الاكتمال مستفيداً من الهاش المسجلة في ملف التورنت، وذلك لضمان التحميل الخالي من الأخطاء، وفي حال فشل الاختبار يتم إعادة تحميل هذه القطعة من جديد، وهذا أحد أسباب زيادة حجم التحميل عن الحجم المحدد للملف. كلما صغر حجم القطعة الواحدة كلما كانت عملية المشاركة أكثر فاعلية، وبالمقابل يزداد حجم ملف التورنت.

لماذا بت تورنت؟

يختلف بروتوكول البت تورنت عن الطرق الأخرىHTTP(و FTP) في عدة نقاط أساسية :

- ينشئ البت تورنت عدة طلبات صغيرة الحجم عبر بروتوكول TCP متصلاً بعدة نظراء، بينما في الطرق العادية الأخرى يتم إنشاء طلب وحيد عبر بروتوكول TCP موجه إلى طرف وحيد.
- يحمل بروتوكول البت تورنت بطريقة عشوائية أو بطريقة "النادر – أولاً"، مما يرفع مستوى التوافرية، بينما الطرق الأخرى تحمل بطريقة تسلسلية.

هذه النقاط تجعل برتوكول البت تورنت يستهلك القليل من موارد المزودين، ويعطي توافرية عالية للملفات المشاركة، كما يتخلص من مشاكل الضغوطات العالية التي تتعرض لها المخدمات العادية. إلا أن لهذا البروتوكول بعض نقاط الضعف نظرياً: الوصول للسرعة القصوى للتحميل قد يأخذ بعض الوقت، نظراً للوقت الذي تستغرقه عملية الاتصال بالنظراء، كما يمكن أن تتغير السرعة أثناء عملية التحميل صعوداً وهبوطاً تبعاً لحالة الاتصال بالنظراء بينما في الطرق الأخرى يتم الوصول إلى السرعة القصوى في وقت قصير، كما تبقى السرعة ثابتة طوال فترة التحميل.

تحميل ومشاركة ملفات التورنت

يقوم المستخدم باستعراض الإنترنت لإيجاد ملفات التورنت التي توافق رغباته، ثم يقوم بتحميلها وفتحها بأحد البرامج الداعمة لصيغة التورنت ومن هذه البرامج (bittorrent) والتي تعرف بالعميل (The Client) حيث

يوصله هذا البرنامج بالمتتبع المسبق تحديده في ملف التورنت فيتسلم قائمة بالنظراء الذين يتبادلون أجزاء ملفات التورنت المحدد. يصبح المستخدم في هذه الحالة نظير هو الآخر يتشارك أجزاء الملفات مع قرنائه. كل مجموعة من النظراء تتشارك في أجزاء ملف التورنت تسمى الحشد او (Swarm) من أمثلة البرامج الداعمة للتورنت

(The Client): BitTorrent, µTorrent, BitComet, KTorrent, Azureus ,Deluge, Free Download Manager and BitLord

ثالثاً:الأنظمة الموزعة[5]

أنظمة الكمبيوتر تطورت بشكل كبير جدا وصولاً إلى الوقت الراهن، التطور الأول: هو تطوير المعالجات في 1980من 8 بت إلى 64 بت. وكانت نقاط القوة لهذه الحواسيب أنها أصبحت حواسيب مركزية المعالجة حيث أصبحت تكاليف معالجة الأوامر منخفضة. التطور الثاني: هو استخدام الشبكات المحلية ذات السرعة العالية مع عدد كبير من العقد (nodes) الأمر الذي أدى إلى وصول سرعة نقل البيانات إلى 1Gbps بل وزاد عن ذلك. وفي نهاية هذه التطورات أصبحت الأنظمة الموزعة تستخدم أجهزة حاسوبية متعددة المعالجات مع شبكات ذات سرعة عالية بدلا من استخدام جهاز حاسوبي كبير وقوي يملك معالج واحد.

وقد تم اول تطوير لأنظمة الموزعة Distribution File Systems (DFSs) في عام 1970 وكان هذا النظام مرتبط بنظام التخزين FTP من حيث الهيكلية ولكن بقي هذا النظام محدود الاستخدام والسبب في ذلك الوقت هو مساحات التخزين الضعيفة.

تقوم DFSs بمعالجة البيانات المولدة على منصات البيانات الرقمية. ويتم التنفيذ بأمان وكفاءة وسرعة. عندما تنمو البيانات بشكل سريع والوصول إليها يصبح سريع أيضاً، الأمر الذي يؤدي إلى نمو موارد التخزين وتطويرها لتصبح أكثر كفاءة وقدرة على تحمل هذا الكم الهائل من البيانات وهنا ظهر مصطلح Big Data.

وأشهر مثال على الأنظمة الموزعة هي النظم السحابية Cloud Systems. حيث يتم تخزين ملفات DFSs على واحد او أكثر من الحواسيب كل منها يدعى خادم (server)، وأجهزة الكمبيوتر تدعى عملاء (client)، ويتم الوصول إلى تلك الملفات كما لو كانت ملف واحد.[6]

الهياكل المستخدمة في الأنظمة الملفات الموزعة:

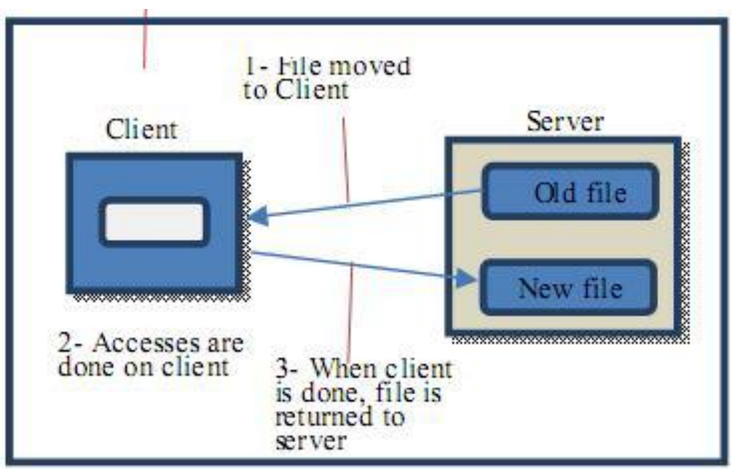
الهدف العام من تصميم DFS هو استخدام أقل لموارد الأجهزة المحلية من خلال تقاسم موارد تلك الأجهزة والاستفادة من منها في إدارة الملفات وهذا مفيد جدا في التصميم العام.

في حين تم تصميم DFS لتوفير خدمات الملف للعميل في هذا الهيكل، يستخدم العملاء الواجهات لإنشاء الملفات وحذف الملفات وقراءة الملفات وكتابة الملفات وإجراء عمليات الدليل. قد يكون نظام التشغيل المستخدم لتنفيذ هذه العمليات نظام تشغيل موزع او طبقة وسيطة بين نظام التشغيل ونظام الملفات الموزع.[7]

تستند بنية DFS إلى ثلاث هياكل :

1-هيكلية مخدم-عميل The Server-Client based architecture

بنية client –server استخدمت على نطاق واسع في DFS والتي تتكون من نموذجين



الشكل 11 هيكلية client-server

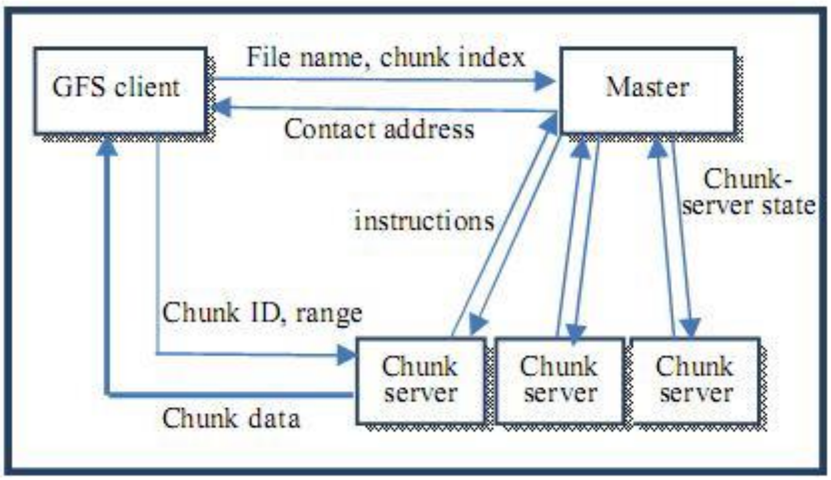
النموذج الأول: هو نموذج الوصول عن بعد. في هذا النموذج، يوفر العميل واجهة مع عمليات الملفات المختلفة. يتم تنفيذ العمليات من خلال هذه الواجهة. ويجب أن يستجيب الخادم لهذا الطلب.

النموذج الثاني: هو نموذج التحميل / التنزيل على عكس نموذج العميل / الخادم[8] في هذا النموذج يقوم العميل بتنزيل الملف الذي ستتم معالجته من قبل العميل محلياً. يتم استخدام نموذج الخادم/العميل من قبل NFS وأصبحتNFS أكثر استخداماً في الوقت الراهن.

2– الهيكلية المعتمدة على التجميع Clustered based architecture

في هذه الهيكلة لا يوجد خادم واحد وإنما يوجد أكثر من خادم ولكن أحد هذه الخوادم هو الخادم أساسي يدعى master server

الخادم الأساسي يحتفظ metadata للبيانات أما الخوادم الأخرى (chunk) تتعامل مع العميل ومن الممكن أن يتعامل مع خادم ثانوي(chunk) مع أكثر من عميل الأمر الذي يودي إلى معالجة كم هائل من البيانات هذه البنية مستخدمة في GFS (Google file system)[9]



الشكل 12 هيكلية google file system

3–الهيكلية المتناسقة architecture Symmetric

في الهيكلية السابقة لم تكن هنالك طبقة تخزين موزعة لذلك يتضح لنا أن الفرق الأكثر أهمية بين خوادم DFS مع symmetric architecture هو إذا كانت تقوم بإنشاء نظام الملفات على طبقة تخزين موزعة، أو أن يتم تخزين كافة الملفات في nodes التي تم إنشاؤها.

تتكون هذه الهيكلية من ثلاث طبقات منفصلة:

- ✓ الطبقة الاولى هي مميزة البحث اللامركزية الأساسية.
- ✓ الطبقة الوسطى عبارة عن طبقة تخزين موزعة بشكل كامل.
- ✓ الطبقة العليا هي طبقة تنفذ نظام الملفات.

معايير التصنيف المتبعة لتصميم النظام الموزع

لدى DFSs عدة تصنيفات تؤثر على صفات الخادم وفيما يلي أهم هذه التصنيفات:

- التسامح في الخطأ Fault Tolerance :

عندما يصبح أي جزء من الموقع الموزع تالفاً، فإنه يتم التسامح دون أن يشعر العميل بذلك.

- الشفافية Transparency :

بالنسبة للعميل فانه يرى النظام الموزع كأنه مخدم واحد وهذا المعيار الأكثر أهمية عند عملية التصميم.

- التكرار Replication :

يتم إنشاء أكثر من نسخة للملفات المخزنة وتخزينها في النظام الموزع وهنا يأتي مصطلح الموثوقية وهو عندما يتم تلف ملف ما ضمن النظام فان ذلك لن يدمر شيء ولن يؤدي إلى أي عطب في المنظومة لأنه يحوي أكثر من نسخة وسهولة استرجاعها من أبسط ما يكون .

- التزامن Synchronization :

بما أننا أنشانا أكثر من نسخة للملف على أكثر من server فأي تعديل على إحدى النسخ يجب أن يتم على باقي النسخ .

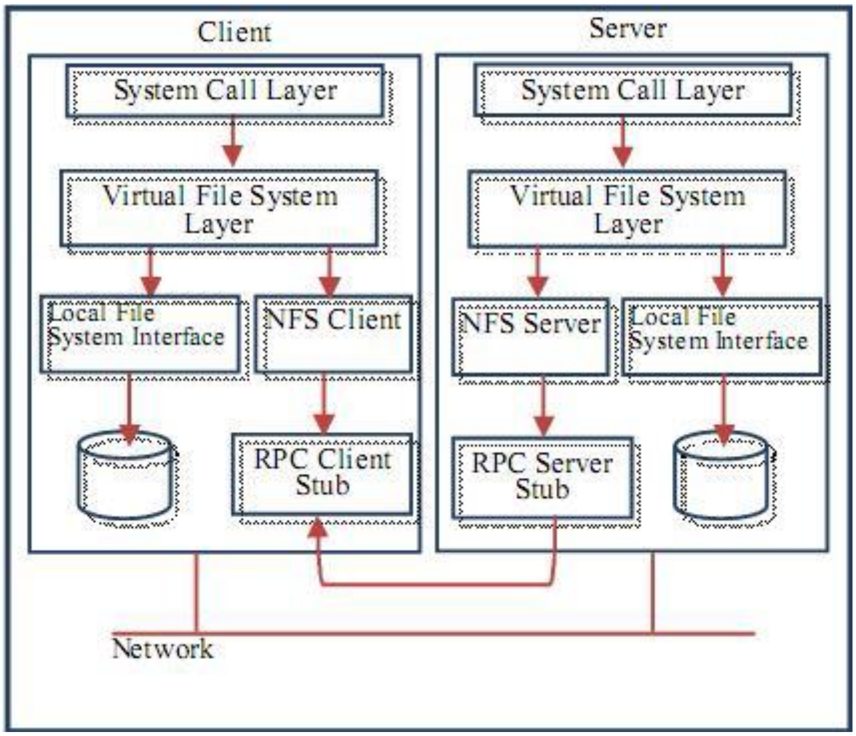
- التسمية Naming :

الأسماء كلها مصادر في النظام الموزع فهنالك يوجد الحواسيب والمخدمات والـ objects البعيدة يقوم النظام الموزع بجعل أسماء الكيانات متناسقة.

الأنظمة الموزعة الشهيرة Distributed File Systems (DFS)

أنظمة الملفات الشبكية Network File System (NFS) [10]

بدأ تطوير NFS في عام 1984 من قبل شركة Sun Microsystems باستخدام أنظمة UNIX



الشكل 13 نظام الملفات الشبكية

وهي تستخدم إجرائية RPC remote procedure call للتواصل بين العميل –الخادم آخر إصدار NFS هو الإصدار الرابع .

هيكل التصميم الأساسي هو التنفيذ الموزع لنظام الملفات UNIX الكلاسيكي. ويتم استخدام نظام الملفات الظاهري (Virtual file system) الذي يعمل كطبقة وسيطة هذا يسمح للعملاء للعمل بسهولة مع أنظمة الملفات المختلفة.

نظام التشغيل عبارة عن استدعاءات متداخلة بين نظام التشغيل ونظام الملفات الموزع كما ويمكن إرسال أكثر من أمر واحد من RPC في الإصدار الأخير.

التسامح بالأخطاء عالي علىNFS [11]، حيث يتم الاحتفاظ بمعلومات حول حالة الملفات. وفي حال حدوث خطأ من العميل، يتم إعلام الخادم. لا يتم النسخ المتماثل على NFS وإنما يتم نسخ النظام بأكمله ويتم التخزين في الذاكرة المؤقتة ومقارنة بالنسخة الموجودة على الخادم فإذا كانت الأوقات مختلفة يتم تعديل الملف وتجاهل ذاكرة التخزين المؤقت كما أن NFS لا يستخدم أسلوب المزامنة لأنه لا يتم نسخ الملفات المتماثلة ويتم تنفيذ العمليات على خادم واحد وهذا ما يسمى نموذج upload–download وتتم الآلية وفق الآتي:

ترسل البيانات التي أجري عليها آخر تعديل إلى الخادم وعندما يرغب العميل أن يفتح ملف ما مخزن مؤقتاً في الذاكرة فيتم التحقق من الملف المخدم كون الملف المخزن في الذاكرة مقفل وبالتالي يتم تحقيق الاتساق بهذه الطريقة وال objects. تتم مشاركتها في حالة واحدة وهي أن تكون ضمن ملف أو جزء من دليل (directory)

في NFS التطبيقات تتم مشاركتها عبر الخادم ولا تحتاج إلى تثبيت على القرص المحلي وبالتالي يمكن أن يكون الجهاز إما خادم أو عميل على حد سواء ونتيجة لذلك فان كلفة التخزين تنخفض بشكل كبير[12].

النظام الموزع CEPH[13-14]

هو كود مفتوح المصدر قائم على النظام الموزع المعتمد على object-based حيث ازدادت شعبيته بشكل كبير.

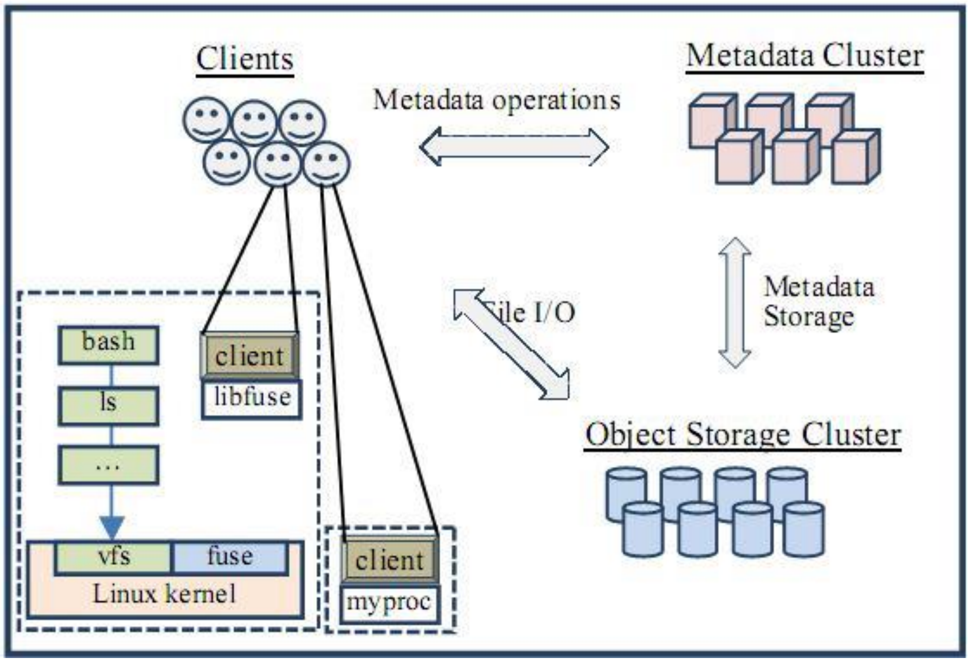
التقرير الأول تم من قبل Weil et al في عام 2006 وتم شراء هذا النظام من قبل Red hat في عام 2014 وتم تطوير 7 إصدارات لهذا النظام وآخر نظام ثابت هو Giant حيث يقدم ثلاث هياكل مختلفة للتخزين وهي object-based storing، block-base storing، file system ، أهم سمة في نظام CEPH هي الموثوقية reliability والقابلية للتوسع scalability.

Metadata هي عبارة عن معلومات عن الداتا المخزنة وبشكل عام الأنظمة الموزعة تقوم بتخزين metadata في خوادم منفصلة ولا يمكن الوصول إلى البيانات إذا كانت Metadata غير متاحة ولكن نظام CEPH لا يستخدم خادم منفصل للاحتفاظ بالـ Metadata وإنما يستخدم خوارزمية خاصة به تسمى (Controlled Replication Under Scalable Hashing)CRUSH .

حيث أن العميل يستخدم تلك الخوارزمية لتحديد وقراءة موقع مجموعة البيانات (dataset) لذلك لم تعد هنالك مشكلة في metadata وكيفية الوصول لها .

Ceph DFS يتم توزيع أكثر من نسخة على الخوادم وبالتالي يتم تحقيق التكرار بهذه الطريقة ووفقاً لقياسات حجم العمل فان CEPH تقدم أداءً جيد جداً لعمليات الدخل -الخرج.

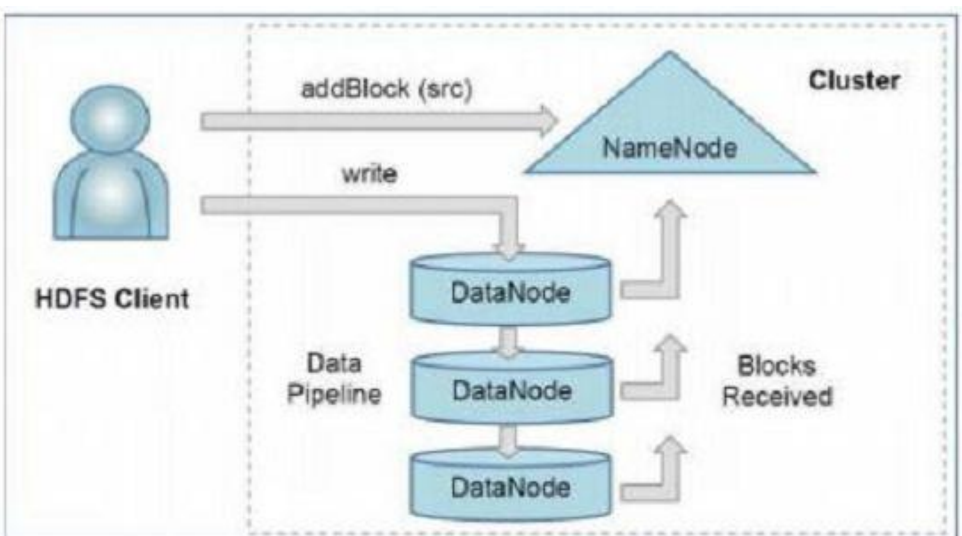
إن هذا النظام الموزع لديه القدرة على إدارة بيانات metadata بشكل عالي جداً قد يصل عدد العمليات في الثانية الواحدة إلى 250000، عملية فمن الممكن دمجه في بيئات أخرى كالسحابة أو الأنظمة الافتراضية (OpenStack, Cloud Stack, VMware).



الشكل 14 نظام CEPH

أنظمة الملفات الموزعة (HDFS) Hadoop Distributed File System.

يستخدم لغة البرمجة JAVA وهيكلته تعتمد على MapReduce حيث يسمح هذا النظام إلى تحليل وتحويل كم هائل من البيانات [15]



الشكل 15 نظام Hadoop

يعتبر HDFS نظام موثوق جداً لتخزين بيانات كبيرة جدا وتأمين تدفق تلك البيانات إلى العميل مع عرض نطاق ترددي عالي جداً[16].

ويقوم بعمل نسخة احتياطية للبيانات فإذا حدث أي فقدان للبيانات[17] فان ذلك لن يؤثر عليها بشيء وإنما يتم نسخ تلك البيانات من عقدة أخرى وبالتالي تبقى البيانات آمنة كلياً كذلك هذا النظام قابل للتوسع لأبعاد تصل إلى petabyte[18].

HADOOP مستخدم في العديد من الشركات الصناعية الكبيرة وكذلك الأكاديمية منها Facebook, yahoo, Twitter, LinkedIn, eBay, AOL, Alibaba, Adobe, IBM[19]

العديد من الشركات تستخدم اليوم الأنظمة الموزعة حيث يتم استخدامها في الشركات الكبيرة والصغيرة والمشاريع أيضاً ومع نمو البيانات بشكل كبير جداً فإن الحاجة ملحة لتلك الأنظمة حيث تقدم إدارة وأداء وتخزين تلك البيانات وخصوصاً BIGDATA.

كما أن تخزين البيانات في الأنظمة السحابية هو خدمة جزئية subservice من نظام الملفات الموزع يستخدم بيئة UNIX حيث يعتمد على تخزين وإدارة البيانات الكبيرة ومشاركة الموارد hardware and software.

الميزة الأساسية في CEPH أنه لا يحتاج إلى خادم يوضع عليه الـ metadata هذه الميزة تجعل من النظام أكثر وضوحا وثقة ويعالج BigData.

فقد أصبح أكثر من 200 مطور وأكثر من 50 شركة مؤيدة لهذا النظام حيث انه مثالي للشركات ذات الميزانيات المنخفضة

أما**HADOOP** يستخدم للبيانات الكبيرة وهو من أكثر الأنظمة المستخدمة في الوقت الحالي حيث يحوي على العديد من الخوادم الموزعة والتسامح في الأخطاء يجعل منه نظاما أمناً وقابلاً للتوسع بشكل كبير فهو مستخدم في تكنولوجيا السحابة التي تعتمد عليه اعتماداً كبيراً في الوقت الراهن وهو الذي سيطغى على الأنظمة الأخرى في المستقبل

ملخص الفصل :

في الدراسة المرجعية هذه تم استعراض تقنيات التقسيم المتبعة فيزيائياً على القرص الصلب ومزايا وعيوب كل طريقة كونها أساس التقسيم في معظم أنظمة التشغيل الموجودة حالياً وفهمها فيزيائياً، ثم تم استعراض تقنيات التحميل والرفع المتبعة حالياً، وفوائد الشبكة وتأثيراتها على تلك التقنيات، لان أي عمل يتعلق بالملفات بحاجة إلى هذه المفاهيم، كذلك تم استعراض الأنظمة الموزعة المستخدمة حالياً بمحاسنها ومساوئها وتعقيداتها.

من خلال الدراسة، تم التطرق إلى نموذج التورنت وهو النموذج الذي يعتمد على التقسيم والتوزيع على أكثر من مستخدم، ولكن هذا التقسيم لم يراعي حساسية الملفات وطبيعتها وطبيعة بنك المعلومات المؤسساتي كونه موجه بشكل عام لجميع المستخدمين لذلك فإن الهدف هو تعديل النماذج المتبعة في معظم أنظمة التقسيم وتكوين نموذج معدل يراعي طبيعة الملفات وحساسيتها.

الفصل الثالث

النموذج المقترح

الفصل الثالث
النموذج المقترح

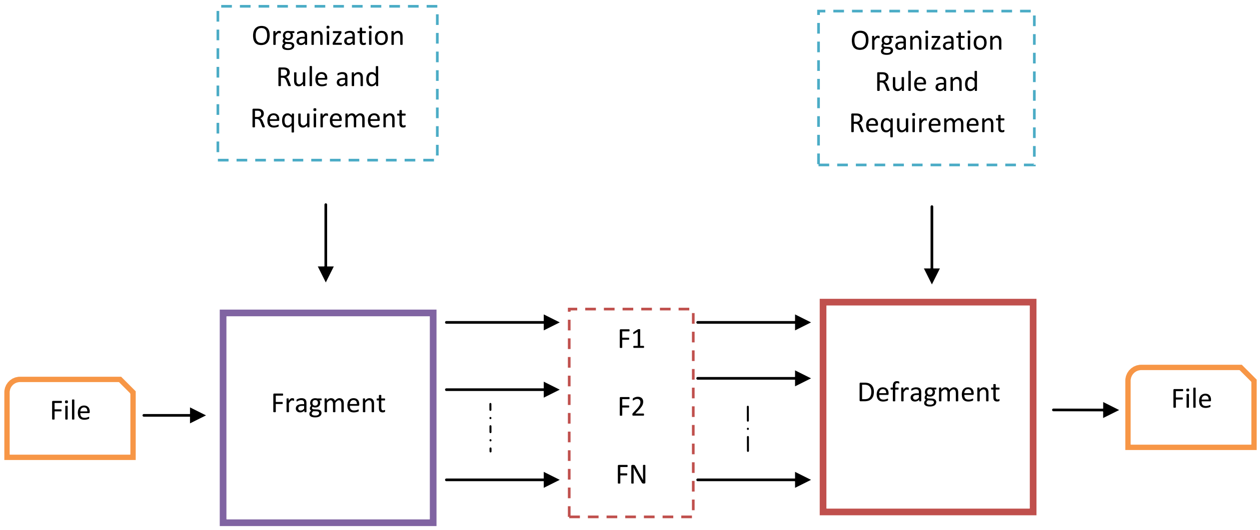
معظم التطبيقات التي تعتمد على تقسيم الملف إلى أجزاء وتقوم باسترجاع تلك الأجزاء تعتمد بشكل أساسي على حجم الملف وتقوم بالتقسيم على أساسه ولكن لاتراعي خصوصية تلك الملفات وطبيعة المؤسسة وكون عملنا يندرج في بنك معلومات مؤسساتي فهذا يحتم علينا ان ننطلق من خصوصة المؤسسة وفهم طبيعة الملفات الموجودة فيها لذلك قمنا بتصميم نموذج يراعي توصيف تلك الحالة حيث انطلقنا من الملف وقمنا بالتقسيم على أساس متطلبات المؤسسة وهي تسمى Organization rule and requirement

ماهي Organization rule and requirement؟

عند إجراء التقسيم والاسترجاع ففي أي نموذج نظري فإننا نحتاج إلى محددات لعمل التقسيم ففي البت تورنت عملية التقسيم تستند على حجم الملف وفي العديد من البرامج المستخدمة والشائعة جداً فان السبب الأساسي للتقسيم والاسترجاع هي الحجم ولا يراعوا ماهية الملفات وطبيعتها في النموذج المقدم الحجم ليس الوحيد ذو قيمة وإنما هنالك محددات أخرى تطلبها المؤسسة ونحن نعمل ضمن بنك معلومات مؤسساتي وهذا بالضرورة يحتم علينا أن نحرص كل الحرص على طبيعة البيانات ولا ننشرها كيفما اتفق وإنما هنالك محددات علينا النظر إليها جيدا قبل الشروع في التقسيم وهذه المحددات **هي سرية المعلومات حماية الملف نوع الملف المدروس حجم الملف بالإضافة لعدد الأقسام المجزئة وهل الملف متاح للجميع أم انه ملف عام بإمكان أي شخص أن يحصل عليه**

عندما يتم تقسيم ملف فإنه يخضع للقواعد الناظمة له وهي حجم الملف ونوعه وسريته وعدد الأجزاء المراد تقسيمه إليها وخرج هذه المرحلة هو عدة ملفات من الملف الأصلي طبعاً.

تلك الأجزاء تدخل إلى مرحلة الاسترجاع، في حال طلبه من أحد المستخدمين فعلينا أيضا أن نراعي تلك القواعد قبل الخوض في عملية الاسترجاع وتتم عملية الاسترجاع لكل الأجزاء وليس لبعض الأجزاء وهنا تأتي المساهمة الفعلية في هذا البحث.

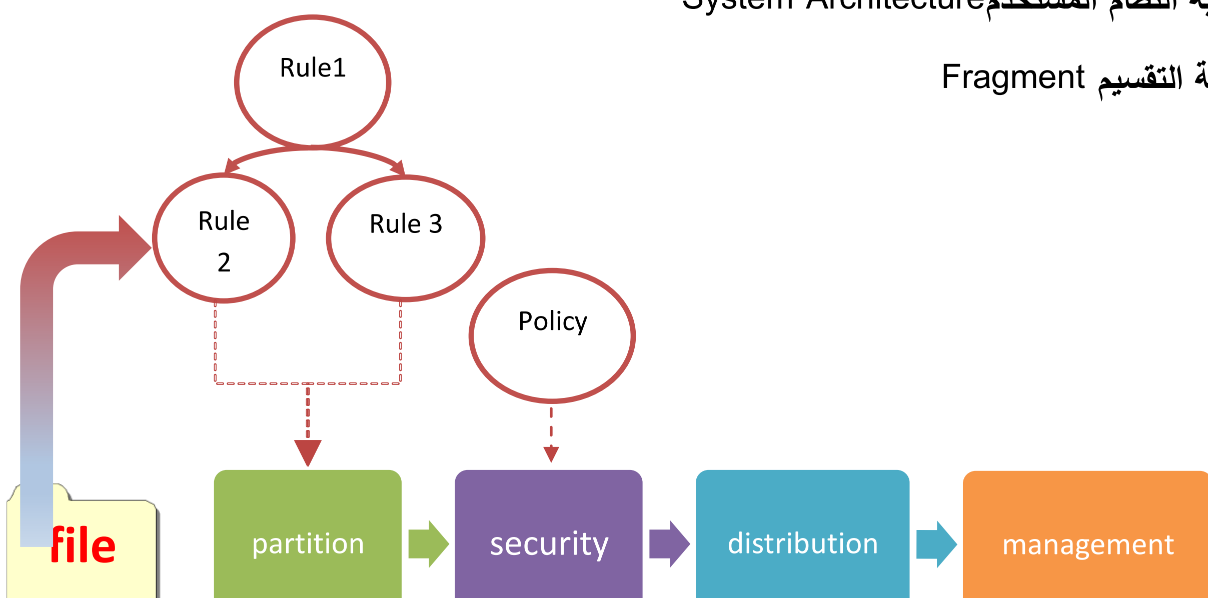


الشكل 16 النموذج المقترح

من خلال النموذج السابق فان الملف يدخل إلى مرحلة التقسيم بناءاً على المحددات أنفة الذكر ويتم تقسيمه إلى عدد من القطع ومن ثم يعاد تجميع تلك القطع بناءاً أيضا على تلك المحددات .

هيكلية النظام المستخدم System Architecture

مرحلة التقسيم Fragment



الشكل 17 هيكلية النظام المقترحة لمرحلة التقسيم

في البداية عندما يأتي ملف مهما كانت حجمه أو لاحقته فانه يدخل إلى النموذج الرياضي السابق وتتم المعالجات الآنية في كل مرحلة .

في كل مرحلة هنالك دخل وخرج خرج المرحلة السابقة هو دخل للمرحلة التالية والملف الأساسي يدخل إلى النموذج الرياضي ضمن مراحل سنقوم بتوصيفها وفق ما يلي:

Partition وفيها يتم تحديد خصوصية الملف أي تحديد فيما إذا كان الملف سري أم لا فإذا كان سري فانه يدخل إلى مرحلة الـ security وينفذ السياسات المطبقة على security أما إذا كان غير سري فإن مرحلة الـ security تتحول إلى جسر عبور فقط.

بعد التعرض للقواعد السابقة فإننا نقوم بتقسيم الملف إلى عدد من الأجزاء أو لا نقوم بالتقسيم مطلقاً وذلك اعتماداً على قواعد التقسيم .

وخرج مرحلة Partition هي عدة ملفات مقسمة أو الملف بحد ذاته دون أن يخضع للتقسيم .

Security: هنالك سياسة أمن علينا مراعاتها عندما يشار إلى الملف انه سري وهي في تطبيقنا الحالي من المخول استعراض الملف؟ أي أن الملف متاح للجميع أم غير متاح.

عن التكلم عن خصوصية الملف هنا لدينا خيارين

الأول هل هو عام هذا يقتضي الذهاب مباشرة إلى مرحلة التوزيع

الثاني هنا نقوم بتطبيق سياسات أمن في مرحلة الـ **security** حيث أن مكون سياسة الأمن عام ولا يفرض سياسة معينة وبالتالي يمكن تطبيق أي سياسة لها أي تعقيد في هذه المرحلة

أمثلة على ذلك:

- ✓ سياسة الصلاحيات مع منظومة لتحقيقها .
- ✓ سياسة توقيع رقمي بمعنى أن الملف خاص يتم تحقيق من سيوقعه وكيفية التحقق من التوقيع عند الاسترجاع .
- ✓ سياسة تشفير بمعنى أن الملف الخاص يمكن تشفيره بخوارزمية محددة مع بروتوكولات لإدارة المفاتيح

Distribution: دخل هذه المرحلة عبارة عن ملف أو عدة ملفات بالإضافة إلى معلومات مؤقتة (مخزنة ضمن Buffers) من المراحل السابقة أما خرجها فهو عبارة عن مسارات الملفات المجزئة والمعلومات السابقة.

Management: هي عبارة عن إدارة المعلومات المخزنة مؤقتا في المراحل السابقة وتخزينها أي هي عبارة عن إدارة Metadata وجميع المعلومات السابقة المخزنة مؤقتا يتم الاستفادة منها وتأطيرها في إطارها الصحيح وإهمال ما يمكن إهماله لكي لا يصبح لدينا أحجام كبيرة للتخزين ويصبح حجم الملف اكبر من حجم الملف الأصلي لذلك لجئنا إلى التخزين المؤقت.

مرحلة الاسترجاع Defragment

وهي عملية استرجاع الملفات المجزئة والموزعة وتتم وفق ما يلي:

عندما يطلب مستخدم ملف معين فانه يدخل إلى مرحلة الـ Management وهي Metadata للملف المطلوب فإذا كان الملف غير متاح للمستخدم الحالي فإنه يعتذر عن إجراء العملية المطلوبة كونه غير متاح له

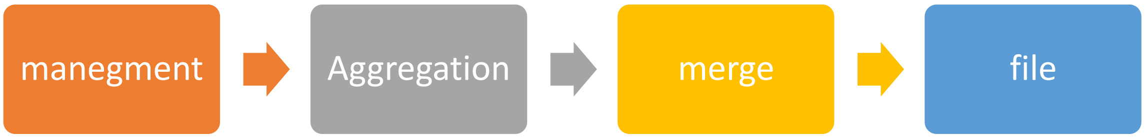
مرحلة الـAggregation : وهي عملية تجميع مسارات الملفات المجزأة للملف الأصلي وهي العملية المعاكسة لعملية Distribution

من مرحلة Management هنالك بعض الملفات متاحة جزئياً أي أن بعض الأجزاء متاحة للجميع وبعض الأجزاء غير متاحة فعند التجميع يقوم بالتأكد من تلك المعلومات وتجميعها للمستخدم وعرض الجزء المتاح له فقط

مرحلة الـ Merge: وهي دمج الملفات من المسارات المختلفة للحصول على الملف الأصلي

في مرحلة الـ Fragment هنالك مرحلة Security وفيها يتم معرفة فيما إذا كان الملف متاح أم لا، أما عند مرحلة Defragment فلا حاجة لذلك كون مرحلة Management تحوي معلومات عن كافة أجزاء الملف وعن سريتها.

هنالك بعض الخوارزميات تقوم باسترجاع الملف كاملا من بعض أجزاءه وهذه العملية خطيرة جداً بالنسبة لنا لأننا نحرص كل الحرص على بقاء سرية الأجزاء لدينا لذلك لا يمكن أن نعرض الملف بدون الحصول على كل جزء من أجزاءه.



الشكل 18هيكلية النظام المقترحة لمرحلة الدمج

ملاحظة:

- في مرحلة التقسيم هنالك مرحلة Security وفيها يتم معرفة فيما إذا كان الملف عام أم لا، أما عند مرحلة التجميع فلا حاجة لها كون تطبيقنا ناقش فقط خصوصية الملف ولكن عند تطبيق سياسة امن معينة كالتوقيع الرقمي أو إدارة الصلاحيات فهو بحاجة إلى دراسة مستقبلية .

التنجيـــز

مقدمة

تم تصميم النموذج لتطبيقه في بيئة العمل لحل مشكلة حقيقية نواجهها في مجال عملنا، لذا تم تنجيز النموذج كنموذج برمجي يدل على إمكانية تحقق هذا النموذج في بيئة عمل حقيقية، لذلك تم إثبات صحة النموذج المقترح من خلال تطبيق عملي، عبارة عن موقع الكتروني يتم من خلاله اجراء اختبار لعملية التقسيم والتجميع

تم اعتماد بيئة MVC–ASP.NET لما تقدمه من تسهيلات برمجية ومكتبات تساعد على سرعة وسهولة في تطوير المواقع، حيث سنقدم شرحاً مختصراً في البداية نعرف من خلالها أهمية هذه البيئة.

ماهي MVC-ASP.NET ؟

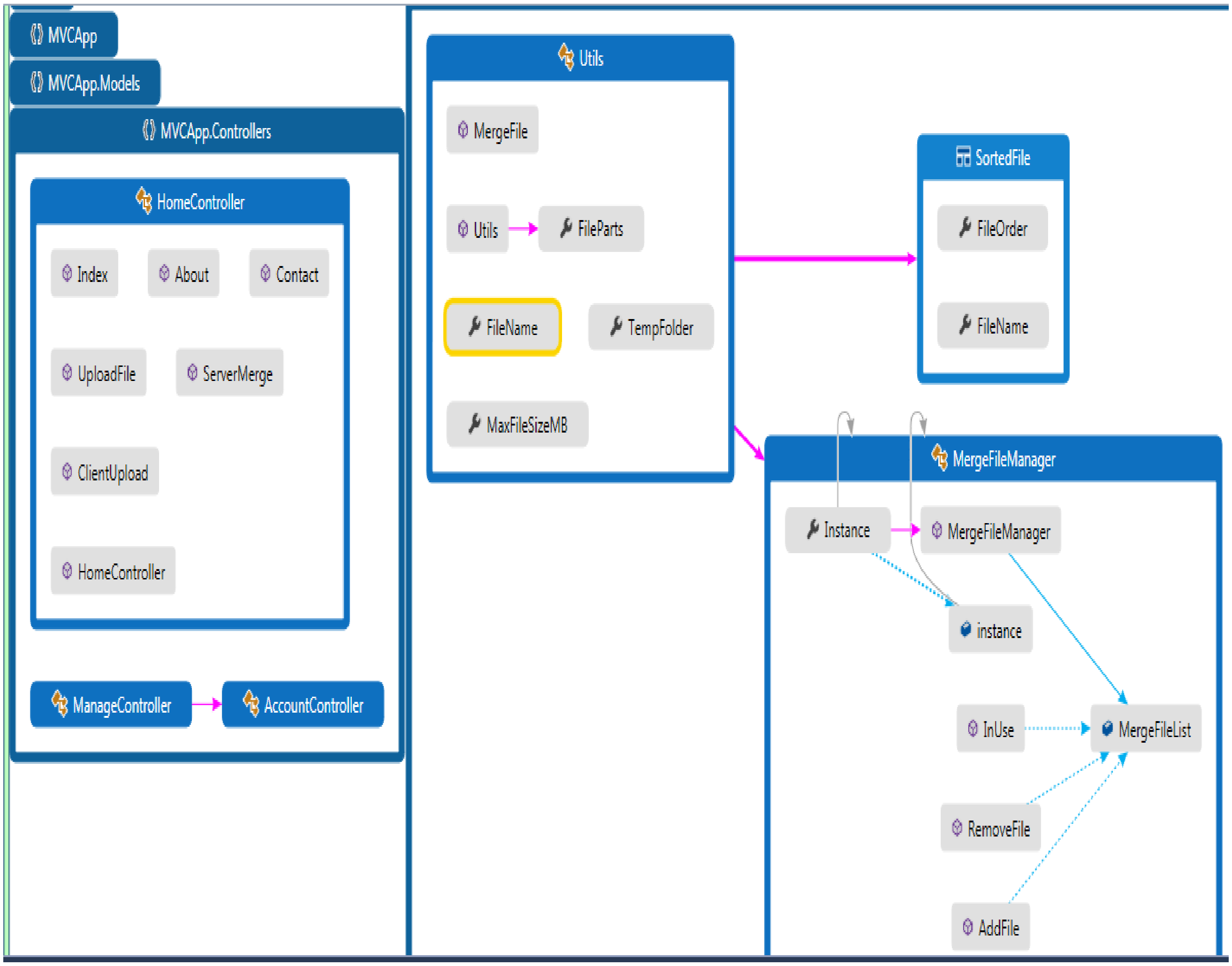
تُعتبّر الـ (ASP.Net) لغّة برمجة من تصميم شركة مايكروسوفت من أجل تطوير البرامج، أو هي آخر عمليّة تطوير لإصدارات التكنولوجيا من مايكروسوفت في عمليّة برمجة الصفحات للخادم النشط (Active Server Pages) تقوم على بناء التطبيقات في الإنترنت سواء مواقع ثابتة أو ديناميكية.

مفهوم **الـ** (Models–Views–Controllers) MVC ، لكل جزء وظيفة.

- Models: هي الطبقة المسؤولة عن التعامل مع قواعد البيانات.
- Views: هي الطبقة التي تمثل واجهة التطبيق وهو الجزء المرئي من التطبيق والذي يظهر للمستخدم ويتضمن في العادة أكواد التصميم.
- Controllers: تمثل هذه الطبقة حلقة الوصل التي تربط ما بين طبقة البيانات Models وطبقة الواجهة Views.

ومفهوم ال MVC ليس حكراً على ASP.NET وإنما هذا التقنية مستخدمة في لغات برمجة أخرى مثل PHP

خريطة الكود البرمجي



الشكل 19 خريطة الكود البرمجي لتقسيم ودمج الملفات

في المخطط السابق يوجد Home Controller وهي الصفحة الرئيسة التي تتحكم بما يعرض للمستخدم تحوي بداخلها مجموعة من الإجرائيات أهمها Server Merge و Client Upload.

Client Upload

يقوم المستخدم برفع ملف إلى Server عن طريق هذه الإجرائية ولكن عملية الرفع لن تتم مباشرة وإنما هنالك قيود على تلك العملية وهي "كما تم توضيحها سابقاً في هيكلية النظام في مرحلة التقسيم" سؤال المستخدم فيما إذا كان هذا الملف سري أم لا لنحصل على المعلومات التي سنستخدمها لاحقا في عملية الدمج. وعملية الرفع ليست

للملف كاملاً وإنما تتم العملية بتقسيم الملف إلى أجزاء بعد الأخذ بعين الاعتبار المعلومات المتعلقة بطبيعة الملف.

Server merge

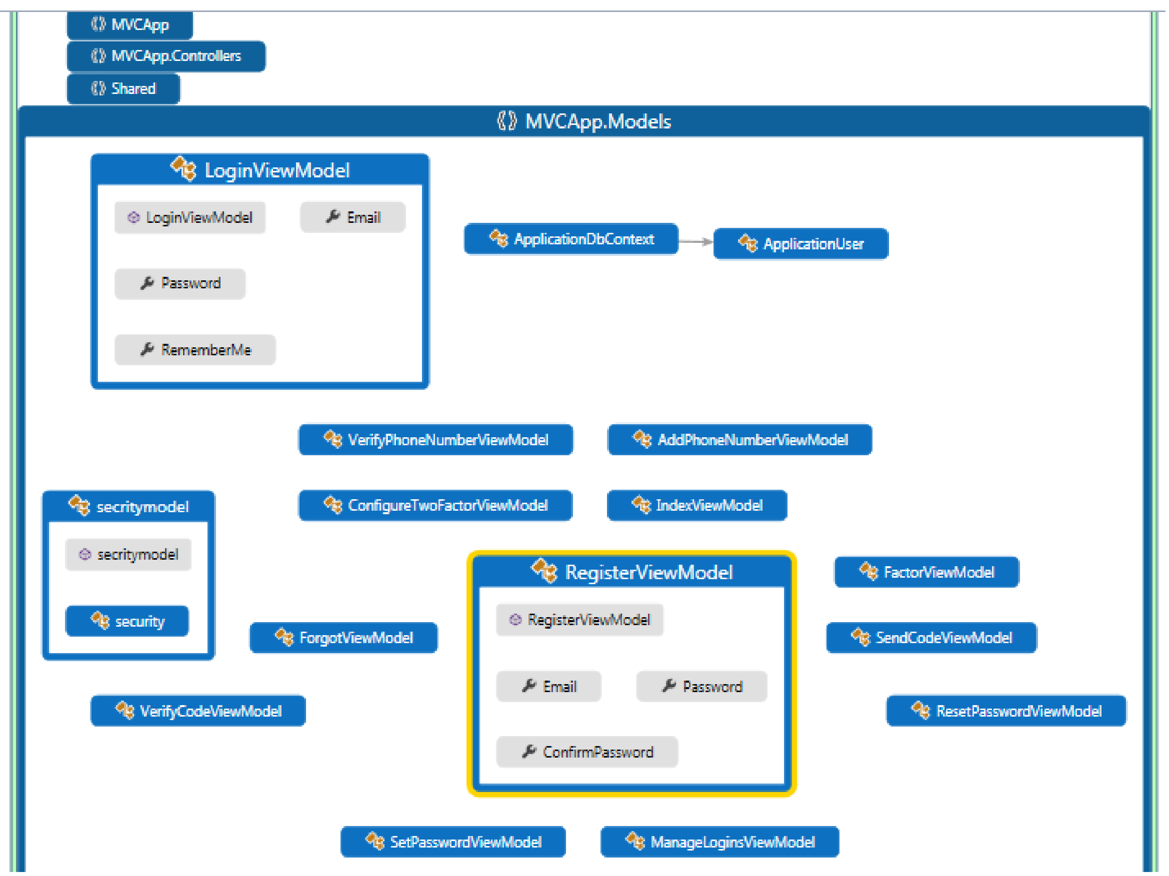
تقوم هذه الإجرائية بعملية الدمج للأجزاء المقطعة بعد جمع المعلومات حول مسارات الأجزاء وطبيعة تلك الملفات.

حيث أن class_utils يحوي على اسا الملف عدد الأجزاء حجم الملف الملفات ومسارات الملفات المؤقتة.

كما يوجد كلاسين تحكم وهو كلاس الإدارة (Manage Controller) الذي يقوم بعملية إدارة المعلومات الواردة من الملف و(Account Controller) الذي يقوم بإدارة الحسابات للمستخدمين.

هنالك مسألة مهمة للغاية وهي أن العمليات تتم على Server ولا تتم على جهاز المستخدم فعملية التقطيع والتوزيع لن تأخذ أي حجم للتخزين على جهاز المستخدم.

تسجيل الدخول للموقع بالإضافة الى تسجيل مستخدم جديد Users Register And Users Login



الشكل 20 خريطة الكود البرمجي للمستخدمين

في خريطة الكود البرمجي هنالك اثنين من النماذج وهي Register View Model And Login View Model يقومان بعملية التسجيل للمستخدمين بالإضافة للسماح لهم بالدخول إلى الموقع وهنالك النموذج الأهم وهو موديل Security Model الذي يقوم بإدارة الملفات من حيث السرية وهل الملفات متاحة للجميع أم أنها عامة لجميع المستخدمين .

ملخص الفصل :

في هذا الفصل تم التعرف على النموذج الرياضي المقترح بالإضافة لهيكلية النظام للتقسيم وهيكلية النظام للتجميع كما تم التعرف في مرحلة التنجيز على بيئة العمل التي تم عليها تنفيذ المشروع .

كذلك تم سرد أهم الإجرائيات التي استخدمت في المشروع للحصول على الغرض المطلوب وهو الإدارة الآمنة لتجزئة الملفات الموزعة .

الفصل الرابع

الاختبارات والنتائج العملية

الفصل الرابع
الاختبارات والنتائج العملية

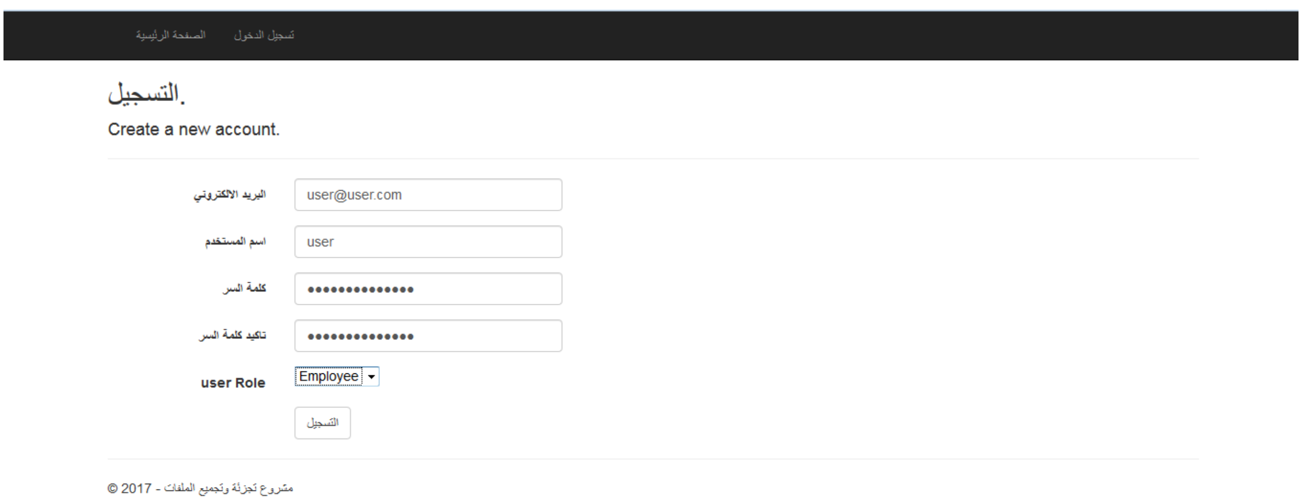
سيتم في هذا الفصل عرض مجموعة من الاختبارات لإثبات إمكانية تحقيق النموذج المقترح في بيئة بنك معلومات مؤسساتي فعلي، إضافة إلى مناقشة النتائج.

نظرة عامة

في هذا الفصل سيتم تقديم سيناريو عملي للنموذج المقترح وتطبيقه في بنك معلومات مؤسساتي والميزة التي قدمها هذا المشروع بالمقارنة مع النماذج المطروحة بالإضافة لاختبار النتائج.

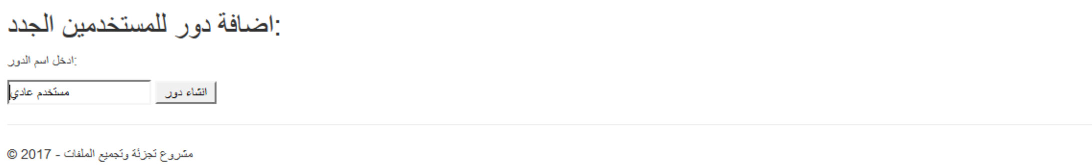
سيناريو الاختبار

يريد أحد المستخدمين الدخول إلى الموقع ورؤية الملفات الخاصة به (عملية تقسيم الملفات لن تظهر للمستخدم) بالإضافة للملفات العامة المشتركة مع جميع المستخدمين ورفع ملف جديد متاح لجميع المستخدمين كما يقوم المستخدم برفع ملف آخر خاص به فقط.



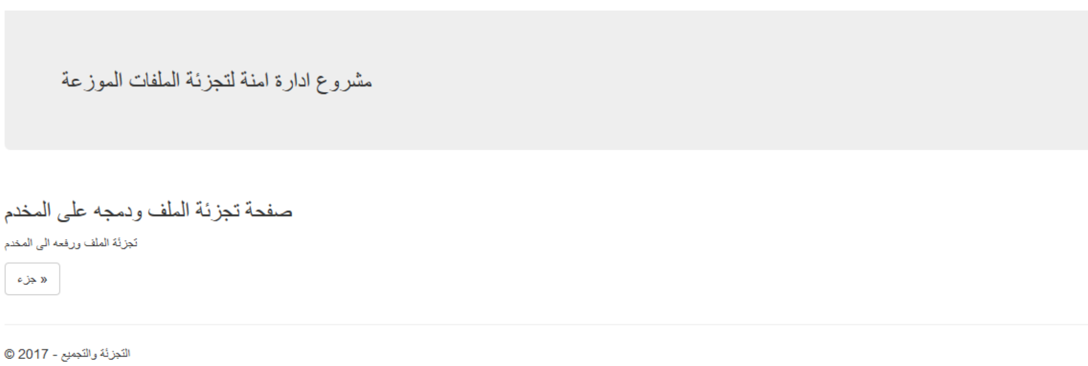
في هذه الواجهة يتم تسجيل كمستخدم جديد إلى الموقع كما تمت إضافة دور للمستخدم هل هو مستخدم عادي أم هو مدير نظام administrator وعند التطرق لمسالة الأدوار فهذا مشروع بحد ذاته لذلك تمت إضافة دور بسيط لتكوين صورة كاملة عن المشروع وهنا المستخدم المدير هو من ينشئ الأدوار.

الشكل 21 إضافة دور جديد للمستخدم



الشكل22 عملية انشاء الدور

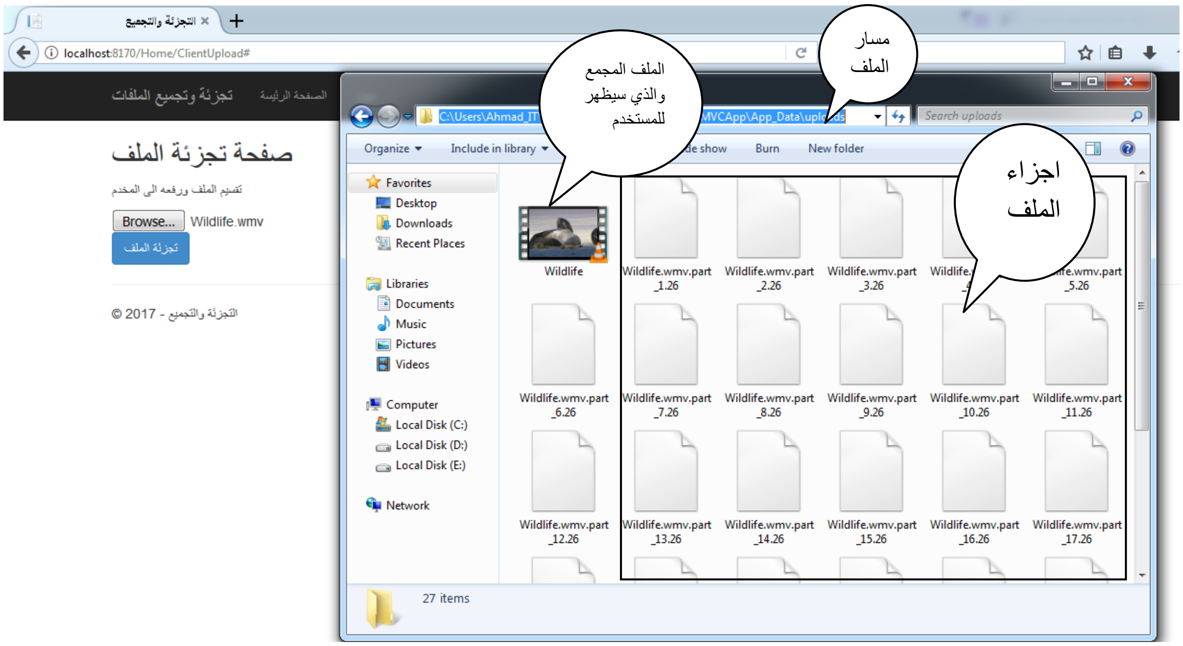
الآن يريد المستخدم رفع ملفه إلى الموقع في هذه المرحلة المستخدم لن يرى آلية التقسيم المتبعة على المخدم



الشكل23 تجزئة الملف ووضعه على المخدم

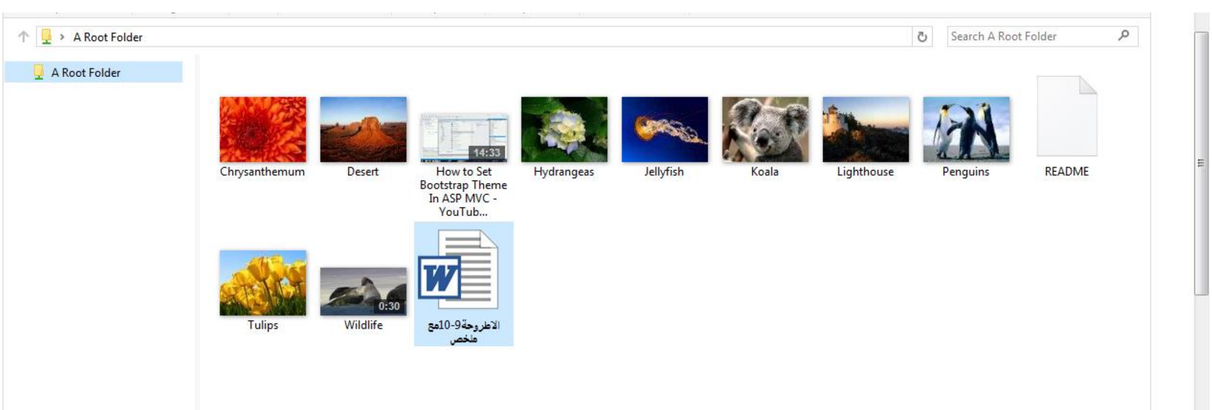
الشكل 24 معلومات الملف وطبيعته

في هذه الصفحة يقوم المستخدم بتجزئة الملف وتحديد إذا كان الملف سري أم لا ويقوم بالضغط على زر التجزئة ليحصل على الملف في ملفه الشخصي إذا كان سري وإذا لم يكن سري سوف يراه جميع المستخدمين لسهولة توصيف السيناريو سيتم عرض الملف مع أجزائه على المخدم في ملف الملفات السرية للمستخدم.



الشكل 25 مكان توضع الملفات السرية على المخدم

بالنسبة لمستعرض الملفات فانه يقوم باستعراض جميع الملفات الشخصية الخاصة بالمستخدم من مختلف الأنواع وهذه هي ميزة إضافية فليس هنالك حاجة لإضافة برامج تدعم تلك الصيغ على حاسب المستخدم من الأنواع التي يدعمها المستعرض ملفات pdf,mp4, word, txt بالإضافة للواحق الصوت المختلفة .

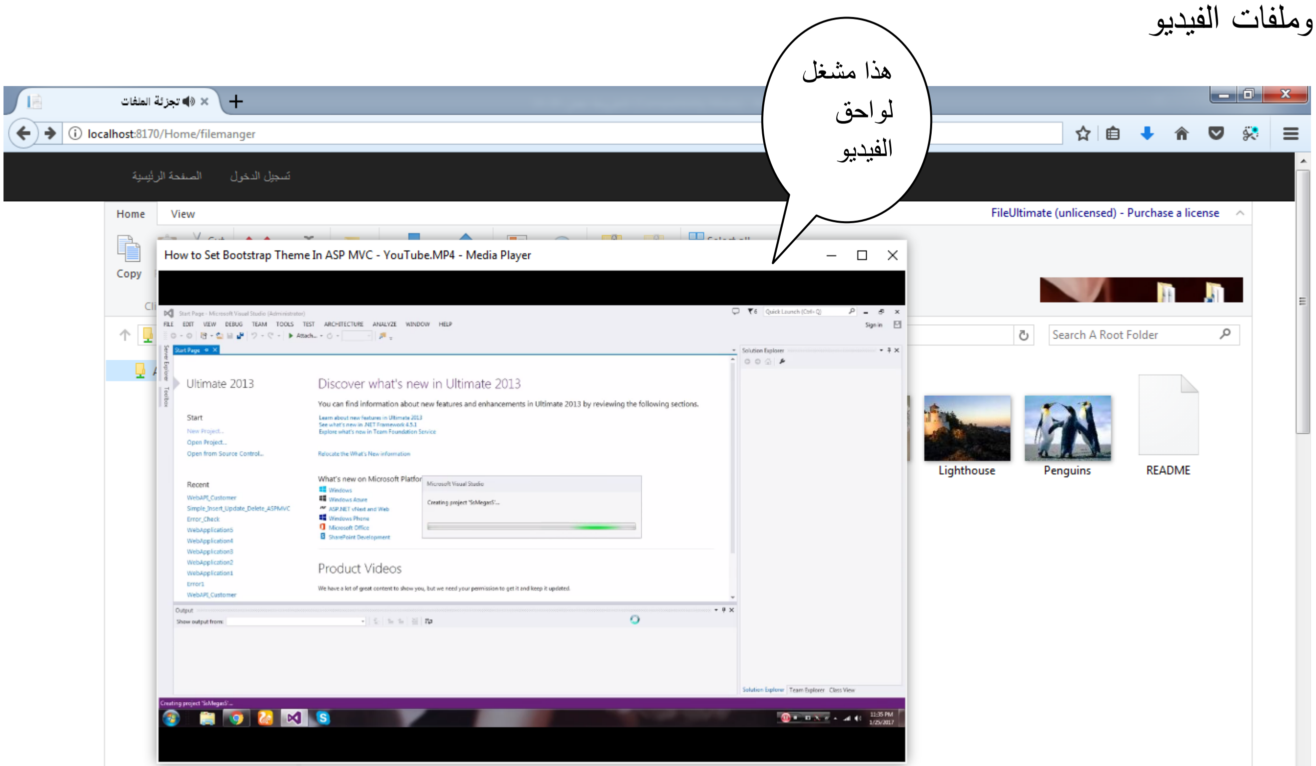


الشكل 26 مستعرض الملفات

وهذا مثال على ملفات word وكيفية استعراضها على المتصفح

الشكل 27 مستعرض ملفات word

وملفات الفيديو



الشكل 28 مستعرض ملفات الفيديو

النتائج

في هذه الدراسة قمنا بتجزئة الملفات وتجميعها بناءا على طبيعة المؤسسة حيث تم الأخذ بعين الاعتبار طبيعة الملف وما يحويه من معلومات قد تكون سرية وقد لا تكون، أي أنها تكون متاحة لجميع المستخدمين.

دراسة حالة:

حجم التخزين

ليكن لدينا مستخدم لديه 100 ملف يريد تقسيم كل ملف إلى 10 أجزاء حجم كل جزء مختلف عن الآخر وفق الجدول التالي وكذلك معامل الزيادة نظرا لكون المعلومات الخاصة بهذه الملفات تشغل حيز من التخزين:

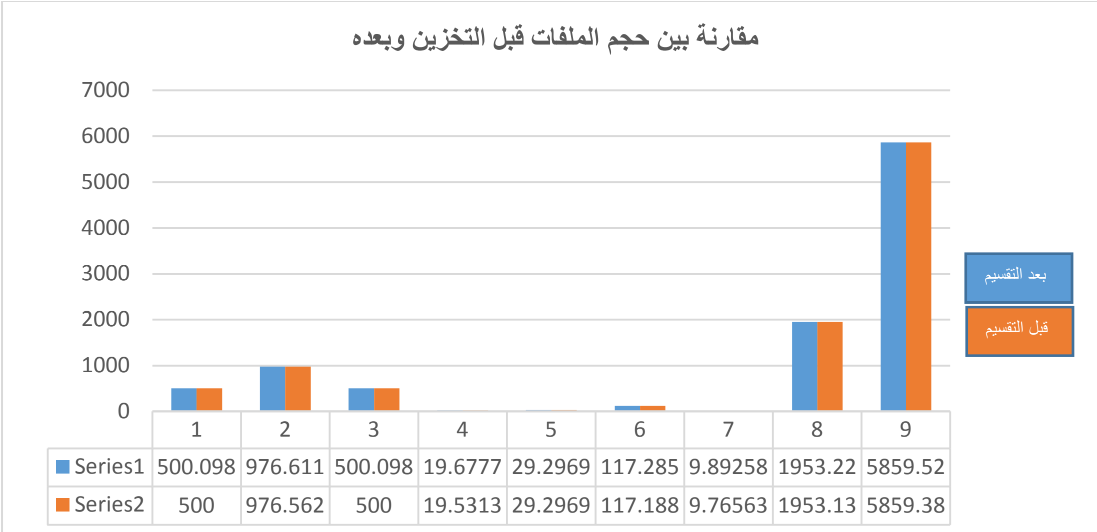
الحجم بالميغا قبل التقسيم 100 ملف	الحجم بالميغا بعد التقسيم 100ملف	معامل الزيادة	حجم كل جزءkb	عدد الملفات للمستخدم الحالي	حجم الملف الكلي قبل التقسيمkb
500	500.0976563	1	512	100	5120
976.5625	976.6113281	0.5	1000	100	10000
500	500.0976563	1	512	100	5120
19.53125	19.67773438	1.5	20	100	200
29.296875	29.296875	0	30	100	300
117.1875	117.2851563	1	120	100	1200
9.765625	9.892578125	1.3	10	100	100
1953.125	1953.222656	1	2000	100	20000
5859.375	5859.521484	1.5	6000	100	60000

الجدول 3 المقارنة الحجم قبل التقسيم و بعده

ويتم حساب الحجم الكلي وفق المعادلة التالية:

حجم الملفات الكلي=عدد الملفات*(عدد الأجزاء* حجم كل جزء+معامل الزيادة) ولدينا معامل الزيادةX حيث

$0kB<X<2kB$



الشكل 29 حجم الملفات قبل التخزين وبعده

السلسلة الأولى تدل على حجم التخزين بعد التقسيم
السلسلة الثانية تدل على حجم التخزين قبل التقسيم

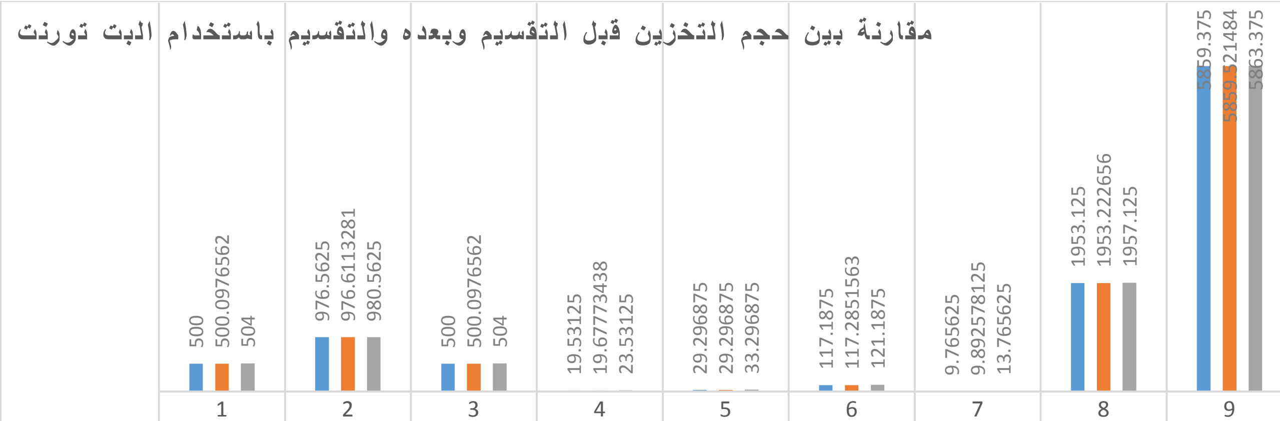
من خلال الشكل السابق نجد أنه لا زيادة تلحظ إذا أصبح هنالك حجوم أكبر بالمقارنة مع الاحتفاظ بمعلومات عن طبيعة الملفات وحساسيتها.

فيما يلي مقارنة مع البت تورنت من حيث عملية التقسيم وهي الموضحة في الفصل الثاني – الدراسة المرجعية .

إن البت تورنت يقوم بتقسيم الملف إلى قطع بحجوم قياسية (32KB, 64KB, 128KB, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB) كما أن هذه القطع بدورها محددة بأقسام أصغر يمثل كل قسم 16KB وعلى هذا الأساس تتم المقارنة باستخدام نفس قيم المثال السابق:

حجم الملف بعد التقسيم في البت تورنت MB	عدد الأقسام	حجم القطعة في البت تورنت MB	الحجم بالميغا بعد التقسيم 100ملف	الحجم بالميغا قبل التقسيم 100 ملف	حجم كل جزء بالنسبة لخوارزمية التقسيم المتبعة
504	256	4	500.0976563	500	512
980.5625	256	4	976.6113281	976.5625	1000
504	256	4	500.0976563	500	512
23.53125	256	4	19.67773438	19.53125	20
33.296875	256	4	29.296875	29.296875	30
121.1875	256	4	117.2851563	117.1875	120
13.765625	256	4	9.892578125	9.765625	9.892578125
1957.125	256	4	1953.222656	1953.125	1953.222656
5863.375	256	4	5859.521484	5859.375	5859.521484

الجدول 4 مقارنة مع آلية التقسيم في البت تورنت



	1	2	3	4	5	6	7	8	9
Series1	500	976.5625	500	19.53125	29.296875	117.1875	9.765625	1953.125	5859.375
Series2	500.0976562	976.6113281	500.0976562	19.67773438	29.296875	117.2851563	9.892578125	1953.222656	5859.521484
Series3	504	980.5625	504	23.53125	33.296875	121.1875	13.765625	1957.125	5863.375

الشكل 30 المقارنة مع البت تورنت

السلسلة الأولى تدل على حجم التخزين قبل التقسيم
السلسلة الثانية تدل على حجم التخزين بعد التقسيم باستخدام الإجرائية المتبعة
السلسلة الثالثة تدل على حجم التخزين بعد التقسيم باستخدام البت تورنت

من خلال الشكل السابق نلاحظ أن حجم التخزين في إجرائية التقسيم المتبعة اقل من التقسيم المتبع في البت تورنت ولو بجزء بسيط كون هنالك معلومات عن طبيعة الملفات وحساسيتها .

بالنسبة لزمن الوصول للملفات لا يجوز أن نقارن زمن الوصول للملفات في البت تورنت كون المستخدم ينتظر حتى يكتمل تنزيل الملف ليتمكن من استعراض الملف على جهازه الخاص ويحجز مساحة تخزينية أيضاً على جهازه أما في المشروع المقترح فالمستخدم يستعرض الملفات من على المخدم دون أن يحتاج إلى تنزيلها على جهازه الخاص.

الفصل
الخامس

الخاتمة والآفاق المستقبلية

الفصل الخامس
الخاتمة والآفاق المستقبلية

من خلال هذا البحث تم استعراض بعض المفاهيم المتعلقة بالملفات من حيث تجزئتها وتوزيعها وتوضعها أيضاً.

كما تم استعراض بعض النماذج عن تجزئة الملفات ولكي تكون المساهمة فاعلة في هذا المجال تم تصميم نموذج رياضي يقوم بتقسيم وإدارة الملفات الموزعة في بنك معلومات مؤسساتي يلبي احتياجات المؤسسة كما تمت مقارنة هذا النموذج مع نموذج البت تورنت من حيث مساحة التخزين وكانت النتائج جيدة جداً في هذا المجال.

فيما يتعلق بالأبحاث المستقبلية المتعلقة بهذا الموضوع فهنالك العديد من الأمور التي تحتاج إلى دراسة، على سبيل المثال لا الحصر:

هنالك العديد من الاتجاهات والمحاور التي يمكن العمل عليها مستقبلاً منها:

- ✓ مسألة إدارة الصلاحيات والأدوار للمستخدمين وهي مسالة غاية في الأهمية في مثل هذا النوع من بنوك المعلومات .
- ✓ العديد من الأبحاث المتعلقة بإدارة المحتوى فلو تم ربط نوع الملفات مع ما تحويه تلك الملفات من معلومات لأصبح هنالك تكامل في المشروع.

References المراجع

([1])Available online http://datalib.edina.ac.uk/mantra/fileformatandtransformation/

([2]) Available online https://webservices.ignou.ac.in/virtualcampus/adit/course/cst101/block4/unit4/cst101-bl4-u4-06.htm

([3])Available online https://en.wikipedia.org/wiki/Segmented_file_transfer

([4])Available online https://en.wikipedia.org/wiki/MD5

([5]) A.S.Tanenbaum and M.V.Steen, "Distributed Systems Principles and Paradigms", USA: Pearson Prentice Hall, 2.Edition, 2006

([6]) P.J.Braam, "The Coda Distributed File System," Linux Journal, vol. 50, no. 6, pp. 10-20, 1998.

([7])Available online: http://www.coda.cs.cmu.edu/ljpaper/lj.html .

([8]) A.S.Tanenbaum and M.V.Steen, "Distributed SystemsPrinciples and Paradigms", USA: Pearson Prentice Hall, 2.Edition, 2006.

([9]) A.S.Tanenbaum and M.V.Steen, "Distributed SystemsPrinciples and Paradigms", USA: Pearson Prentice Hall, 2.Edition, 2006.

([10]) R. Sandberg, D. Goldberg, S. Kleimen, D. Walsh and B. Lyon, "Design and Implemantation of The Sun Network File

([11]) G. Coulouris, J. Dollimore and T. Kindberg, DistributedSystems: Concepts and Design, USA: Addison-Wesley Publishing Company, 2011.

([12])B.KarasuluandS.Korukoğlu, "ModernDağıtıkDosyaSistemlerininYapısal Karşılaştırılması," in X. AkademikBilisim,Çanakkale,Turkey, 2008.

([13]) S. Weil, S. Brandt, E. Miller, D. Long and C.Maltzahn,"Ceph: A scalable, high-performance distributed file system (pp. 307-320). USENIX A," in 7th Symposium on Operating Systems Design and Implementation, USENIX, USA, 2006.

([14])S. Weil, S. Brandt, E. Miller, D. Long and C.Maltzahn,"Ceph: A scalable, high-performance distributed file system (pp. 307-320). USENIX A," in 7th Symposium on Operating Systems Design and Implementation, USENIX, USA, 2006.

([15]) M. Grossman, M. Breternitz and V. Sarkar, "Hadoopcl2: Motivating the design of a distributed, heterogeneous programming system with machine-learning applications," IEEE Transactions on Parallel and Distributed Systems, vol.27, no. 3, pp. 762-775, 2016.

([16]) N. Shankaran and R. Sharma, "Cloud Storage Systems-A Survey," Indiana University, USA, 2011

([17]) K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The hadoop distributed file system (pp. 1-10)," in IEEE 26th symposium on mass storage systems and technologies (MSST), NV, USA, 2010.

([18]) G. Yavuz, S. Aytekin and M. Akçay, "Apache Hadoop veDagıtıkSistemlerÜzerindekiRolü," Journal of the Institute of Science & Technology of Dumlupinar University, no. 27, pp. 43-54, 2012

([19])Available online: https://wiki.apache.org/hadoop