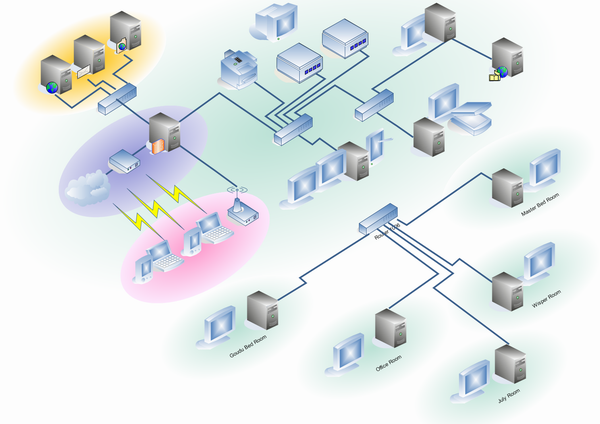
**بحث عن الأنظمة الموزعة هندسة برمجيات**



**الأنظمة الموزعة**

**اعداد**

**م/امين الملحاني**

**هندسة برمجيات- جامعة صنعاء**

*للتواصل :* [*ENG.AAHA13@GMAIL.COM*](mailto:ENG.AAHA13@GMAIL.COM) */00966554241989*

**الفصل الاول**

**المقدمة**

**لمحة تاريخية:**

تاريخيا بدأ الحديث عن الانظمة الموزعة في النصف الثاني من الثمانينات للحصول على حاسب بقدرة اكبر من المعطيات, وذلك للحصول على مركز حساب computing center لخدمة عدد كبير من المؤسسات.

اي كان لدينا حاسب واحد بقدرات كبيرة ويتم النفاذ الية عن طريق طرفيات في اماكن بعيدة.

مع الوقت زادت الرغبة لوجود نظام حاسوبي الى للمستخدم فظهر mini computer وهو كمبيوتر بحجم الطاولة تقريبا, وهو اول مرحلة دخل فيها مفهوم time sharing .في هذه المرحلة كان الحاسب نفسة يقوم بجميع الاعمال .

في مرحلة لاحقة ظهرت فكرة حواسيب متخصصة لنوع معين من التطبيقات بحيث يتم التركيز على جانب واحد من الاعمال حسب المجال الذي يعمل فيه (مثلا قدرة حسابيه عالية) وبالتالي اتجهنا الى مفهوم التخصص.

اعتبارا من اللحظة التي اتجهنا فيها الى التخصص ظهرت الحاجه لتقسيم العمل الى عدة اجزاء(العمل لم يعد موجود على حاسب وحيد وانما على عدة حواسيب).

* اذا المبدأ الاول لظهور الأنظمة الموزعة هو التخصص:

النظر الى المنظومة الحاسوبية ككل على انها تتكون من عدة

اجزاء كل منها متخصص بعمل ما وكل منها لديه عدد من الموارد مسؤول عنها ويتم التشارك فيها.

**تعريف**

معظم الأنظمة الحالية تعتبر انظمه موزعه والأنظمة الموزعة هي عباره عن انظمه تعمل على العديد من الأجهزة مرتبطة مع بعض بشبكه.

**دوافع توزيع الأنظمة ومزاياها**

* الموثوقيةReliability

كان مفهوم الموثوقية متخصصا بعالم التصنيع فعندما نقول تجهيزات ذات وثوقيه عالية يعني انها لا تتعطل, ومن ثم انتقل هذا المصطلح الى البرمجيات وظهر لنا سؤال طالما حيرنا وهو متى نعتبر النظام موثوقا؟

يختلف مفهوم الموثوقية حسب نسبة الخطأ المقبولة ,

لذا كان الوصول الى الموثوقية امرا صعبا بالطرق التقليدية ولرفع الموثوقية نسبة 20% يجب رفع الكلفة 50% من هنا ظهرت فكرة رفع الموثوقية من خلال الأنظمة الموزعة فبدلا من رفع وثوقيه كل جهاز يتم استخدام اجهزة عده وفي حالة تعطل واحد يعمل الاخر.

* التشارك بالمصادر Sharing of resources

كان هناك ضرورة كبيرة للتشارك بالمصادر لأنها كانت مكلفة فظهرت فكرة توزيعها .

* تجميع القدرة الحاسوبية Aggregate computing power

الحصول على قدره هائلة والاستفادة منها واستغلالها وذلك عن طريق توزيع العمل على اجهزه .

* الانفتاح وقابلية التوسع Openness / Scalability

تعتبر الأنظمة الحاسوبية انظمه قابلة للتوسع ومفتوحه حيث يمكننا في لحظه تطويرها والتكامل معها

**تحديات ومساوئ الأنظمة الموزعة**

* الامن Security

تعتبر مشكلة الأمنية من المشاكل الهامه جدا في الأنظمة الموزعة وهنا تجدر الملاحظة الى ان الامن هو الفرع الاهم حاليا في الحواسيب فكلما ازداد انتشار الحوسبة في مجالات الحياة العامة زاد الخطر والخوف من اختراق المنظومات الحاسوبية.

ومن المشاكل الأمنية هجمات حجب الخدمة DOS (Denial Of Service) وهي عباره عن ارسال عدد كبير من الطلبات الى المخدم بحيث يتوقف عن العمل ويصبح عاجز عن استقبال طلبات جديده.

* التوزيع الفيزيائي للموارد مقابل الطلب Physical Distribution of resources vs. demand
* محدودية القدرة الحسابية لكل عقدة Computing power per node is limited

وهذه المشكلة تظهر عندما نريد استخدام الأنظمة الموزعة كبديل عن الأنظمة المتوازية

كمثال على ذلك بعض انظمة المحاكاة تحتاج الى نصف مليون جهاز لحل عمليه وحيده

**. انظمة التشغيل الموزعةDistributed Operating Systems**

هناك العديد من المتطلبات التي يجب توافرها كي نحكم على النظام بانه نظام تشغيل موزع

وهي كما يلي

* Transparency وتعني اعطاء المستخدم انطباع بانه لا يعمل على نظام تشغيل موزع
* اخفاء اماكن وجود الموارد
* توفير تقنيات لحماية الموارد وتعني قدرة النظام على حماية الموارد من المخاطر المختلفة على سبيل

الدخول غير المخول السيطرة عليها من قبل مستخدم ما

* القدرة على تامين الاتصالات ومنع سرقة المعلومات

**تعريف انظمة التشغيل الموزعة Distributed OS**

هي عبارة عن انظمة تشغيل تظهر للمستخدم وكانه يعمل على نظام مركزي الا انه في الحقيقة يعمل على عدة وحدات معالجه مركزيه تعمل على مستوى التوزيع

الفرق بين الأنظمة الموزعة والمتعددة المعالجات هو ان الارتباط بين الانظمة متعددة المعالجات شديد بينما يعتبر الارتباط بين الأنظمة الموزعة بسيطا .

**الفصل الثاني**

**بنيان الأنظمة الموزعة**

**هناك العديد من المعماريات المعروفة عن الأنظمة الموزعة وسوف نستعرضها كما يلي**

* **Clint/server model تتم في هذة الانظمة نمذجة التطبيق كمجموع من الخدمات التي يقدمها ال server, وكمجموعة من العملاء الذين يستخدمون هذة الخدمات المتاحة التي يعرفها العميل مسبقا, وليس من الظروري ان يهتم بالعملاء الاخرين ولن يحتاج لمعرفتهم , لان كل من العملاء والخدمات هي اجرائيات مستقل.**
* **DISTRIBUTED OBJECT MODEL**

**في هذا النموذج لا فرق بين كل الاجهزه المرتبطه فكلها يمكنها تنفيذ مهام مطلوبة وكلها يمكنها طلب خدمات من الاجهزه الاخرى .**

**معمارية الزبون / المزود**

اصبح التعبير Client[Server](http://www.damasgate.com/vb/f32/) في السنوات الأخيرة من أكثر التعبيرات رواجا في عالم [الحاسوب](http://www.damasgate.com/vb/f35/). وكثيرا ما نجد الباعة وهم يصفون النظم التي يسوقونها بأنها تعمل وفقا لنموذج Client[Server](http://www.damasgate.com/vb/f32/) أو نجد أنصاف المتعلمين وهم يرطنون بهذا المصطلح في سياقات تبتعد بذا القدر أو ذلك عن معناه الحقيقي .

ويستخدم تعبير Client[Server](http://www.damasgate.com/vb/f32/) لوصف معماريات العديد من الأنواع البرمجية، مثل نظم تشغيل وبرمجيات ال[شبكات](http://www.damasgate.com/vb/f62/) وقواعد البيانات و[برامج](http://www.damasgate.com/vb/f9/) تطبيقية أخري. ونظرا لوجود معان عديدة لكل من كلمتي Client و [Server](http://www.damasgate.com/vb/f32/)، كما هو حال معظم الكلمات التي ن[جده](http://www.damasgate.com/vb/)ا في ال[قواميس](http://www.damasgate.com/vb/f9/)، فإن اختيار ال[ترجمة](http://www.damasgate.com/vb/f100/) المناسبة لهاتين الكلمتين يجب أن تأخذ بالاعتبار خصوصية ال[صور](http://www.damasgate.com/vb/f22/)ة المجازية التي ينبع منها مصطلح Client[Server](http://www.damasgate.com/vb/f32/) . وطبقا لهذه ال[صور](http://www.damasgate.com/vb/f22/)ة يمكن [ترجمة](http://www.damasgate.com/vb/f100/) كلمة Client بمعني [الزبون](http://www.damasgate.com/vb/t6510/) الذي يطلب الحصول علي [بضاعة](http://www.damasgate.com/vb/f8/) أو خدمة معينة . و[ترجمة](http://www.damasgate.com/vb/f100/) كلمة [Server](http://www.damasgate.com/vb/f32/) بمعني النادل أو [المزود](http://www.damasgate.com/vb/t6510/) الذي يستطيع تزويد الزبون بال[بضاعة](http://www.damasgate.com/vb/f8/) أو الخدمة التي يطلبها منه .

ماهي [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود؟

تقوم [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود ب[صور](http://www.damasgate.com/vb/f22/)تها المجردة علي بناء ال[برامج](http://www.damasgate.com/vb/f9/) ضمن وحدتين، تدعي الاولي وحدة الزبون Client، وتتلخص مهمتها الرئيسة في إدارة التفاعل مع المستخدم وتلقي طلباته وصياغتها علي شكل رسائل Messages قياسية وتدعي الثانية وحدة المزود [Server](http://www.damasgate.com/vb/f32/) ، وتتلخص مهمتها الرئيسة في تلقي رسائل وحدة الزبون وتفسيرها والقيام بأعمال المعالجة وإرسال نتائج المعالجة إلي الزبون الذي طلبها . ويمكن ل[برنامج](http://www.damasgate.com/vb/f9/)ي الزبون والمزود أن ينفذا بواسطة [الحاسوب](http://www.damasgate.com/vb/f35/) ذاته ، ولكن غالبا ما ينفذان بواسطة حاسوبين مختلفين يلائمان الاحتياجات الخاصة بكل منهما .

وتحتوي معظم نظم التشغيل ال[شبكات](http://www.damasgate.com/vb/f62/) المحلية ذات مزودات الملفات مثل (Novell NetWare )علي بعض الأشكال من [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود في توفير كل من [خدمات](http://www.damasgate.com/vb/f64/) نقل الملفات والطباعة لمستخدمي ال[شبكة](http://www.damasgate.com/vb/f62/)، ففي مثل هذه ال[شبكات](http://www.damasgate.com/vb/f62/) يقوم كل من الحواسيب المتصلة بال[شبكة](http://www.damasgate.com/vb/f62/) بتشغيل [برنامج](http://www.damasgate.com/vb/f9/) موجه Redirector ويقوم هذا ال[برنامج](http://www.damasgate.com/vb/f9/) بتلقي طلبات خدمة الملفات الصادرة عن مستخدم ذلك [الحاسوب](http://www.damasgate.com/vb/f35/) أو عن أحد ال[برامج](http://www.damasgate.com/vb/f9/) التطبيقية التي يشغلها ( جهة الزبون ) ، وإحالة الطلب إلي نظام ملفات ال[شبكة](http://www.damasgate.com/vb/f62/) NFS ( جهة المزود ) . وعندها يقوم مزود الملفات بتنفيذ الخدمة المطلوبة وإفادة الزبون طالب الخدمة بالنتيجة ( يستخدم تعبير مزود الملفات File [Server](http://www.damasgate.com/vb/f32/) لإشارة إلي [برنامج](http://www.damasgate.com/vb/f9/) بهذا الاسم وكذلك إلي [الحاسوب](http://www.damasgate.com/vb/f35/) الذي يقوم بتشغيل هذا ال[برنامج](http://www.damasgate.com/vb/f9/) ).  
  
  
  
ال[برنامج](http://www.damasgate.com/vb/f9/) الزبون :

تمثل [برامج](http://www.damasgate.com/vb/f9/) الزبون الواجهة الأمامية Fron-end التي يتعامل المستخدم مع التطبيق من خلالها . وتراعي في [تصميم](http://www.damasgate.com/vb/f12/)ها وبالتالي ط[بيع](http://www.damasgate.com/vb/f8/)ة التطبيق والمشكلات التي يرمي إلي حلها، من وجهة نظر مستخدم التطبيق . فمثلا ، إذا كان التطبيق من فئة نظم إدارة قواعد البيانات فإن عمل [برنامج](http://www.damasgate.com/vb/f9/) الزبون تمكين المستخدم من إدراء العمليات المألوفة لقواعد البيانات . مثل إدخال البيانات وتعديلها وإعداد تقارير وصياغة الاستعلامات بلغة SQL وما إلي ذلك . ومن ثم تقوم ب[ترجمة](http://www.damasgate.com/vb/f100/) إجراءات المستخدم إلي رسائل قياسية وإرسالها عبر ال[شبكة](http://www.damasgate.com/vb/f62/) إلي ال[برنامج](http://www.damasgate.com/vb/f9/) المزود . ومن مهام [برنامج](http://www.damasgate.com/vb/f9/) الزبون أيضا إدارة الموارد المحلية لمحطة العمل مثل لوحة المفاتيح والشاشة والذاكرة والمعالج والأجهزة المحيطية المرتبطة بها مباشرة . ولا يدخل تنفيذ [تعليم](http://www.damasgate.com/vb/f35/)ات الاستعلام المكتوبة بلغة SQL وغيرها من فعاليات معالجة قواعد البيانات في نطاق اختصاص [برنامج](http://www.damasgate.com/vb/f9/) الزبون .

ال[برنامج](http://www.damasgate.com/vb/f9/) المزود :

يتخصص ال[برنامج](http://www.damasgate.com/vb/f9/) المزود بتقديم صنف معين من ال[خدمات](http://www.damasgate.com/vb/f64/) مثل [خدمات](http://www.damasgate.com/vb/f64/) إدارة الملفات أو [خدمات](http://www.damasgate.com/vb/f64/) الطباعة أو الاتصالات أو إدارة البريد الالكتروني e-mail أو [خدمات](http://www.damasgate.com/vb/f64/) استعلام قواعد البيانات query بلغة SQL وإجراء المعاملات transaction معها، أو غيرها من ال[خدمات](http://www.damasgate.com/vb/f64/) . ويمثل ال[برنامج](http://www.damasgate.com/vb/f9/) المزود نهايةة back-end التطبيق. وهو يقف علي أهبة الاستعداد لتقديم ال[خدمات](http://www.damasgate.com/vb/f64/) التي يختص بها عند تلقيه "رسالة" بهذا الصدد من أحد الزبائن . وعندها يقوم بتفسير تلك الرسالة وتنفيذ العمليات التي تتطلبها وإرسال النتائج إلي زبون طالب الخدمة، إن وجدت .

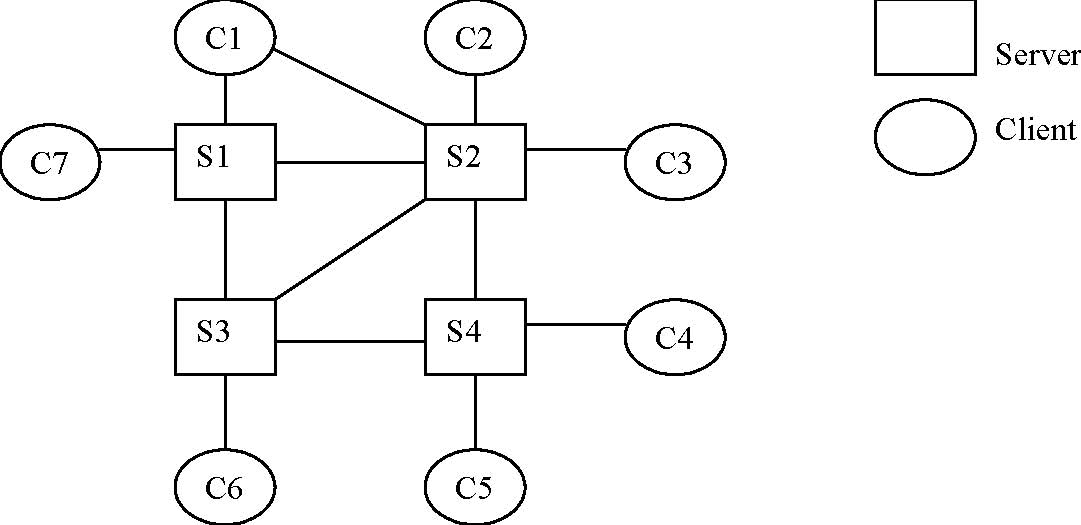
ويتألف [برنامج](http://www.damasgate.com/vb/f9/) المزود من محرك Engine لتنفيذ المهام التي يختص بها، وهو يوفر واجهة ل[برمجة](http://www.damasgate.com/vb/f81/) التطبيقات Application Programming Interface API واضحة ومحددة . وتعين واجهة [برمجة](http://www.damasgate.com/vb/f81/) التطبيقات مجموعة ال[خدمات](http://www.damasgate.com/vb/f64/) التي يمكن للمزود أن يوفرها، وصيغ الرسائل التي تلتمس هذه ال[خدمات](http://www.damasgate.com/vb/f64/). ولا يحتاج ال[مبرمج](http://www.damasgate.com/vb/f81/)ون الذي يطورون [برامج](http://www.damasgate.com/vb/f9/) من فئة الزبون، عمليا، لمعرفة تفاصيل آلية عمل [برنامج](http://www.damasgate.com/vb/f9/) المزود الذي سيخدم [برنامج](http://www.damasgate.com/vb/f9/)هم الزبون . كل ما يحتاجونه بهذا الصدد هو معرفة واجهة ال[برمجة](http://www.damasgate.com/vb/f81/) API الخاصة بال[برنامج](http://www.damasgate.com/vb/f9/) المزود المعني .وتتمثل ال[وظيفة](http://www.damasgate.com/vb/f8/) الأساسية لـ API في إخفاء التعقيدات الداخلية لل[برنامج](http://www.damasgate.com/vb/f9/) المزود وراء ستار أنيق وواضح يسهل التعامل معه.  
ويتيح الفصل بين وحدتي الزبون والمزود إمكانية تشغيل كل منهما علي منصة مفصلة. حيث أن لكل منهما متطلبات تشغيل مختلفة . بل أن بالإمكان تشغيل برمجيات الزبون والمزود تحت نظم تشغيل مختلفة وباستخدام فئات مختلفة من العتاد مثل استخدام نظم PC عاملة تحت DOS و Windows لتشغيل [برامج](http://www.damasgate.com/vb/f9/) الزبون، واستخدام نظام ميني فريم يعمل تحت النظام [Unux](http://www.damasgate.com/vb/f32/) كمنصة للمزود، او استخدام أية تشكيلات أخري غير متجانسة من العتاد أو نظم التشغيل.

حوار الزبون والمزود :

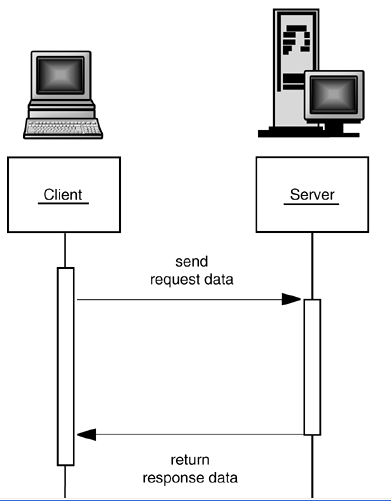
يمتاز الحوار الذي يدور بين الزبون والمزود بالتنظيم والفاعلية، مما يخفف من اكتظاظ كابلات ال[شبكة](http://www.damasgate.com/vb/f62/) بتدفق البيانات بدون طائل كما في حالة التطبيقات العاملة في بيئة شبكية تقليدية . فلو تأملنا مثلا طريقة تنفيذ استعلام في نظام إدارة قواعد البيانات وفق [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود ، نجد أن العملية تبدأ بتلقي الزبون جملة الاستعلام من المستخدم مكتوبة بلغة SQL ( تقوم واجهات استخدام بعض [برامج](http://www.damasgate.com/vb/f9/) الزبون ب[كتاب](http://www.damasgate.com/vb/f57/)ة جمل الاستعلام بهذه اللغة بناء علي طلب المستخدم بالنيابة عنه لتوصيلها إلي المزود)، وعندها يقوم [برنامج](http://www.damasgate.com/vb/f9/) الزبون ب[تحويل](http://www.damasgate.com/vb/f49/) الاستعلام إلي رسالة قياسية وإرسالها إلي ال[برنامج](http://www.damasgate.com/vb/f9/) المزود عبر ال[شبكة](http://www.damasgate.com/vb/f62/). يحلل ال[برنامج](http://www.damasgate.com/vb/f9/) المزود الرسالة القادمة وينفذ [تعليم](http://www.damasgate.com/vb/f35/)ات SQL التي تحتويها، في بيئته الخاصة . وأخيرا يرسل البيانات التي يتمخض عنها البحث فقط إلي الزبون الطالب، بدون الحاجة إلي إرسال كامل الملف إلي محطة العمل لتنفيذ الاستعلام المطلوب في فضائها الخاص .  
) من الامثلة الشائعة التي نراها دائما في مجال برمجيات الزبون/المزود : ال[متصفح](http://www.damasgate.com/vb/f9/)ات والمزودات في المواقع التي نزورها بال[متصفح](http://www.damasgate.com/vb/f9/) (.  
  
  
الزبون/المزود .. لماذا؟

لم تنشأ [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود محصلة للتطور التكنولوجي فقط ، بل جاءت كذلك تلبية لمتطلبات المناهج الادارية والتنظيمية التقدمية التي تتبناها أعداد متزايدة من المؤسسات ال[حديثة](http://www.damasgate.com/vb/f8/) . ففيما تقوم المؤسسات التقليدية علي نظام مركيز هرمي صارم للمراتب القيادية، تنحو المؤسسات ال[حديثة](http://www.damasgate.com/vb/f8/) إلي تبني نماذج تنظيمية تتسم بقدر أكبر من التسطح flat، وتؤكد علي العمل الجماعي وتشجيع التفاعل الافقي بين زملاء العمل علي حساب الهيمنة المركزية وال[روابط](http://www.damasgate.com/vb/f9/) العمودية بين رؤساء ومرؤوسين، وتؤكد المناهج التنظيمية ال[حديثة](http://www.damasgate.com/vb/f8/) علي ضرورة توفير [نقطة](http://www.damasgate.com/vb/f101/) وحيدة يتعامل معها العميل في كل مرة يتصل فيها بالمؤسسة. وضرورة اتصال العميل بالموظف ذاته بصرف النظر عن ط[بيع](http://www.damasgate.com/vb/f8/)ة حاجته، ويتطلب ذلك تحلي الموظفين بمواهب ومهارات متعددة . كما يتطلب تأمين وصولهم السريع والفعال لنطاق واسع من التطبيقات التي تستخدمها المؤسسة، خلال الفترة القصيرة التي يستغرقها اتصال العميل. من ناحية أخري هناك ميل في المؤسسات ال[حديثة](http://www.damasgate.com/vb/f8/) لتنويع مصادر العتاد والبرمجيات وعدم الارتهان لمصدر واحد لها، كما أن احتياجات المؤسسات لنظم المعلومات تستر في النمو والتنوع، فيما تزداد الحاجة لإدماجها في وحدة متكاملة.  
  
وتلبي نظم الزبون/المزود معظم هذه الاحتياجات والتوجهات للمؤسسات ال[حديثة](http://www.damasgate.com/vb/f8/) . وهو سر رواجها في السنوات الأخيرة علي الرغم من كل التعقيدات والتكاليف التي ترافق بناء مثل هذه النظم .  
  
فإذا كانت مؤسستك من هذا الطراز أو ثمة خطط ل[تحويل](http://www.damasgate.com/vb/f49/)ها بهذال الاتجاه، فستجد في [معمارية](http://www.damasgate.com/vb/f26/) الزبون/المزود النموذج المنشود والخيار الأفضل لبناء نظام المعلومات الملائم لهذا التوجه ولتطويره مستقبلا .

**مخطط يبين برمجة معمارية المزود والزبون**



**شكل يوضح حوار الزبون / المزود**



**طبقات التطبيق application layer**

عند تصميم نظام الزبون / المزود يتم تقسيم النظام تطبيقيا الى ثلاث طبقات توضح كما يلي

تقسم التطبيقات الى ثلاث طبقات :

1-طبقة العرض

وفيها يتم عرض البيانات وتوفر واجهة للمستخدم النهائي

2-طبقة المعالجة

وفيها يتم القيام بعمليات المعالجة واجراء العمليات الحسابية

3-طبقة ادارة قواعد البيانات

وفيها يتم ادارة قاعدة البيانات و تقديم البيانات المطلوبة من قبل الزبون

طبقات التطبيق ..لماذا؟

نحن نعرف ان الأنظمة الموزعة من احد مشكلاتها الأمنية ,ان من حلول تعزيز الأمني في الأنظمة هو تقسيم النظام الى عدة طبقات بحيث يتم وضع الاجزاء المهمة في الطبقات الدنيا للنظام وبهذا نستطيع وضع قوانين ومنهجيات للوصول المخول لمثل هذه الاجزاء ومن هنا جاءت معمارية الطبقات الثلاث للتطبيق.

وعلى اساس طبقات التطبيق تنوعت معماريات العميل الزبون الى نوعين هما

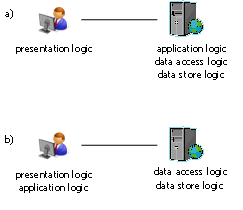
* Two-tier C/S architecture
* Three-tier C/S architecture

في Two-tier architecture يتم توزيع الطبقات بنسبة2:1 وهذا يعني انه على المزود او الزبون ان ياخذ طبقتين ويأخذ الاخر الطبقة الاخرى وقد انبثق عنها معماريتين حسب

عدد الطبقات التي على الزبون هما

* THIN CLINT architecture
* FAT CLINT architecture

**المخطط التالي يوضح كلا من المعماريتين**



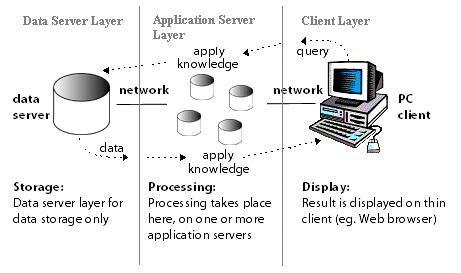
دلكل واحدة من المعماريات استخداماتها حسب مجالات التطبيقات فمثلا تكون المعمارية الاولى اجدى عند تطوير الأنظمة من انظمه قديمة او في حالة عدم الحاجة لخدمة احد الطبقات بينما النوع الثاني يستخدم عندما تكون انظمتنا تتطلب عمل اكبر على جميع الطبقات.

من عيوب المعمارية الاولى هو اجهاد المزود والشبكة بينما تخلصت المعمارية الثانية من هذه المشكلة لكن فقد ظهرت بها مشكلة جديد تتعلق بعمليات الادارة والصيانة فأي تغيير على المزود يتطلب تغييرا على كل زبون.

Three-tier C/S architecture

جاءت هذه المعمارية بحلول لكل مشاكل معمارية Two-tier C/S architecture

واحد امثلة هذه المعمارية هي شبكة الانترنت والشكل التالي يوضح هذه المعمارية



**طريقة الاتصال :**

Remote Procedure Call (RPC)

ظهرت طريقة الاتصال Remote Procedure Call (RPC) في الثمانينات لكنها مرت بمراحل تغيير عديده وهي الطريق المتبعة للتواصل بين المزود والزبون وسوف سنتعرف عليها في هذه السطور تقوم (Remote Procedure Call)على مبدأ توزيع العمل على عدة مزودات واستخدام التفرعات البرمجية لتنفيذ هذه العمليات .

نموذج التنفيذ يعتمد على واجهة للزبون عن بعد STUP (تسمى بالنسبة للزبونPROCXY)



**خصائص PRC**

* Uniform call structure

وهذه تعني ان طريقة الاستدعاء لا زالت بنفس الطريقة المستخدمة في العمل على حاسب وحيد

* :Type checking

توفر ميزة فحص اخطاء انواع المتغيرات كتلك الموجودة بلغات البرمجة

* Full parameter functionality

نستطيع عن طريقها تمرير المتغيرات سواء كان التمرير بالعنوان أو القيمة

* Distributed binding

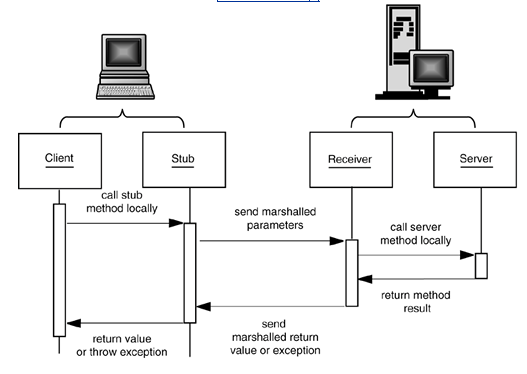
يمكننا التنفيذ على اكثر من مزود

* :Recovery of orphan computations

في حالة تجاهل اي مهمة فانه من الممكن استعادتها

**الية الاستدعاء في PRC**





**(نموذج الكائن الموزع) DISTRIBUTED OBJECT MODEL**

ليس هناك فرق بين معمارية الكائن الموزع وبين الخادم والعميل حيث ان:

* كل كينونة قابلة للتوزيع (distributable) هو كائن (object) يزود خدمات لكائنات اخرى ويستقبل خدمات من كائنات اخرى.
* التواصل بيت الكائنات تتم عن طريق وسيط (middleware system) يدعى بوسيط طلب الكائانات (object request broker)
* معمارية الكائن الموزع اكثر تعقيدا في التصميم من معمارية (c/s)

تعتبر أنظمة الكائن الموزع من أكثر الأنظمة الموزعة مرونة حيث تتركز فكرة المعمارية الأساسية إن مجموعة الكائنات التي تقدم عدة خدمات عن طريق واجهاتها تقوم كائنات أخرى باستدعاء هذه الخدمات وذلك دون اي فصل بين (c \ s)

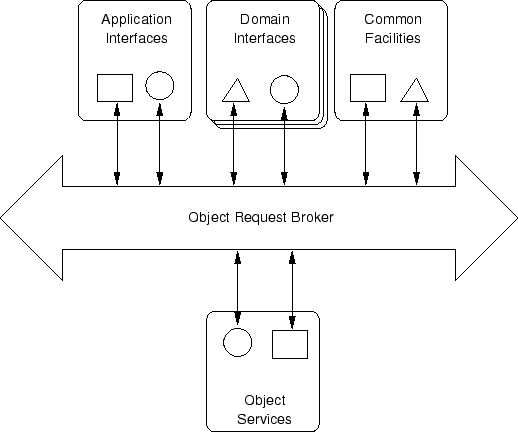
يعتبر استخدام معمارية الكائن الموزع أفضل من الخادم- العميل مع أنواع التطبيقات التي تسمح بتجديد والتطوير لان ال(c/s) يهتم كثيرا بتحديد الخدمات التي يقدمها إدارة قواعد البيانات بينما أنظمة الكائنات الموزعة غير ذالك حيث تسمح بإضافة قواعد بيانات جديدة دون حدوث أي خلل للنظام حيث يعتبرها كائن موزع جديد يحوي عدة كائنات.

**طريقة الاتصال**

يتم الاتصال بين هذه الكائنات عن طريق برنامج وسيط يقوم بالتنسيق بين الكائنات يسمى**Object Request Broker**

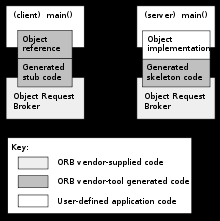
# Object Request Broker

وهو عبارة عن برنامج وسيط من اجل ايجاد حلول للتعقيدات التي تواجه الأنظمة الموزعة والالتزام بمعايير تمكننا من حل مشكلة التنوع في الكائنات الموزعة وانشاء واجهات بين الكائنات بغض النظر عن نوع الكائنان وبيئات عملها الشكل التالي يوضح الهيكل الاساسي لorb



يوضع هذا الوسيط على معيار يسمى معيار **Common Object Request Broker Architecture**

يوضح الشكل التالي المعمارية الشائعة للانظمة الوسيطةCORBA

****

وهي كلمة اختصار لـCommon Object Request Broker Architecture وهي عبارة [إنكليزية](http://ar.wikipedia.org/wiki/%D9%84%D8%BA%D8%A9_%D8%A5%D9%86%D9%83%D9%84%D9%8A%D8%B2%D9%8A%D8%A9) تعني باللغة العربية أسلوب البناء الوسيطي لطلب الأغراض العامة.

يعتبر أسلوب البناء الوسيطي لطلب الأغراض العامة CORBA معيار قياسي معرّف من قبل مجموعة إدارة الأغراض (: Object Management Group‏ OMG). يخوّل هذا المعيار المكونات البرمجية: Software components) المكتوبة [بلغات حاسوبية](http://ar.wikipedia.org/wiki/%D9%84%D8%BA%D8%A9_%D8%A8%D8%B1%D9%85%D8%AC%D8%A9) مختلفة، والتي تعمل على أجهزة متعددة، يخولها بالعمل معا.

تعتبر كوربا CORBA تقنية في البرمجيات مستخدمة في تنظيم وتوحيد طرق استدعاءات المناهج (الطرائق) method-call بين أغراض التطبيقات التي تقع في المجال العنواني نفسه للتطبيق أو تقع في مجال عنواني بعيد (أي عنوان المضيف نفسه، أو عنوان مضيف بعيد على شبكة ما). لقد تم اصدار النسخة 1.0 في شهر تشرين الأول أكتوبر من عام 1991 ميلادي.

تستخدم كوربا CORBA لغة تعريف الواجهات Interface Definition Language IDL لتحديد الواجهات التي سوف تقدمها الأغراض البرمجية للعالم الخارجي outside world. عندئذ كوربا تؤمن الانتقال mapping من IDL لغة تعريف الواجهات إلى أي لغة برمجة خاصة بالتنفيذ implementation مثل لغة C++ أو لغة جافا. هناك ارتباط قياسي standard mapping لعدد من لغات البرمجة منها لغة آدا Ada ولغة C ولغة C++ ولغة Lisp ولغة Smalltalk ولغة جافا ولغة كوبول COBOL ولغة PL/L ولغة بايثون Python. كما وهناك ارتباطات غير قياسية non-standard mapping للغات أخرى مثل لغة بيرلPerl ولغة فيجوال بيسك Visual Basic ولغة Rubyروبي ولغة Erlang ولغة Tcl منجزة من خلال الوسائط لطلب الأغراض العامة ORBs Object Request Brokers ومكتوبة من أجل تلك اللغات.

ان مواصفات كوربا تقتضي وجود ORB لكي يتمكن من خلاله التطبيق بالتعامل مع أغراض برمجية أخرى. في الواقع، يقوم التطبيق بإعطاء قيمة ابتدائية للـ ORB ويقوم بالنفاذ إلى وصيلة الغرض Object Adapter الذي يحافظ على بعض الأمور متل عداد المرجع وسياسات إنشاء الأغراض وسياسات فترة حياة الأغراض.. الخ.

يستخدم الـ Object Adapter لتسجيل الـ instances من الصفوف مولدة النص البرمجي generated code classes. تعتبر الصفوف مولدة النص البرمجي Generated Code Classes على أنها نتيجة ترجمة الكود الخاص بـ IDL المستخدم الذي يقوم بتحويل تعاريف الواجهات عالية المستوى إلى أساس صفوف خاصة باللغة وخاصة بنظام التشغيل لكي يتم استخدامها في تطبيق المستخدم. وتعد هذه الخطوة ضرورية للتأكيد على دلالات ألفاظ الكوربا، كما وتعد ضرورية لتقديم عملية صحيحة للمستخدم للمواجهة مع البنية التحتية لكوربا. هناك لغات مقابلات IDL تعتبر أكتر صعوبة من غيرها. على سبيل المثال طبيعة لغة البرمجة جافا تقتضي أن الـ IDL-Java Mapping سلس ويستفيد من كوربا بشكل بسيط في تطبيقات الجافا. بينما C++ Mapping لا تعتبر سهلة ولكنها مناسبة لجميع هيئات الكوربا مثل معالجة الاستثناءات Exception Handling. كما وأن C-mapping تعتبر أشد غرابة (لأنها ليست لغة برمجة غرضية التوجه) ولكنها منطقية وتعالج RPC semantics بشكل جيد. (لقد استبدلت Red Hat Linux و GNOME UI system الـ IPC المبني على كوربا بآخر هو DBus) إن تحقيق التقابل بين لغات البرمجة يتطلب من المبرمج أن يكتب IDL code يمثل واجهة إلى الأغراض البرمجية الخاصة به. وعادةً هناك مترجم خاص مع كوربا هو IDL compiler يحول IDL code الذي كتبه المبرمج إلى نص برنامج مولد حسب لغة البرمجة المحددة. ومن ثم يقوم مترجم تقليدي بترجمة نص البرنامج المولد لإنشاء ملفات أغراض القابلة للربط لهذا التطبيق.

بعض مواصفات كوربا هي: أنماط المعطيات، الاستثناءات Exceptions، بروتوكولات الشبكة، الزمن المحدد للتواصل communication timeouts. على سبيل المثال: يتواجد في جهة المخدم Server عادةً Portable Object Adapter (POA) يقوم بتوجيه الطلبات اما إلى الخادم المحلي local servants أو يقوم بتوجيهها إلى مخدمات أخرى servers وذلك لتحقيق موازنة الحمل.

بعض الأمور التي لا تهتم بها كوربا ولكنها موجودة في جميع الأنظمة الموزعة الأخرى هي: مدة حياة الغرض، التكرارية وتجاوز الأعطال، name semantics، إدارة الذاكرة، الموازنة الديناميكية للحمل، وفصل النماذج عن بعضها (اظهار، معطيات، تحكم). إضافة لكون كوربا تقدم للمستخدم تقنية استدعاء الاجرائيات عن بعد بغض النظر عن البيئة، فهي أيضاً تعرّف بعض الخدمات الهامة مثل المناقلات transactions والأمان والأحداث والوقت وغيرها من نماذج الواجهات حسب المجال المحدد.

**CORBA Component Model (CCM) نموذج الكوربا**

إن CORBA Component Model (CCM) هو إضافة لمجموعة تعريفات كوربا. أول ظهور له كان في CORBA 3 كما وأنه يوصّف اطار عمل framework للتطبيقات قياسي لـ CORBA Component. بالرغم من أنه لا يعتمد على Enterprise Java Beans (EJB) التي هي مستقلة عن لغة البرمجة، الا وأنه أكثر عمومية من EJB لأنه يقدم أربع أنواع من الـcomponents بدلاً من 2 التي تعرّفها EJB. كما وأنه يعرّف تجرّد من الأشياء entities التي يمكن أن تقدم وتقبل خدمات من خلال واجهات اسمية معرفة جيداً تدعى ports. لدى CCM حاوٍ للمكونات component container لكي تُنشَر المكونات البرمجية فيه. ان هذا الحاوي يقدّم مجموعة من الخدمات للمكونات. بعض هذه الخدمات هي التنبيه والتوثيق authentication والاستمرارية وإدارة المناقلات. وهذه الخدمات هي أكثر الخدمات التي تطلبها النظم الموزعة، ومن خلال نقل التحقيق لهذه الخدمات من المكونات البرمجية إلى حاوِ المكونات ينخفض تعقيد المكونات components بشكل كبير. 2-5- بروتوكول InterORB العام (General InterORB Protocol(GIOP يعتبر GIOP بروتوكول مجرد تتخاطب من خلاله الـ ORB مع بعضها البعض. المعايير القياسية المرتبطة بهذا البروتوكول متبناة من قبل مجموعة OMG. إن بنية GIOP تقدم بعض البروتوكولات متل:

* Internet InterORB Protocol (IIOP) وهو تحقيق لـ GIOP يستخدم في الإنترنت ويقدم تقابل بين رسائل GIOP و TCP/IP layer.
* SSL InterORB Protocol (SSLIOP) وهو IIOP فوق SSL ويقدم ترميز encryption وتوثيق Authentication.
* HyperText InterORB Protocol (HTIOP) وهو IIOP فوق HTTP ويقدم بروكسي شفاف جانبي.

**المساوئ والمشكلات**

- شفافية الموقع، حيث أن الأغراض البرمجية الموجودة في المجال العنواني ذاته والتي يمكن الوصول إليها من خلال استدعاء بسيط تُعامَل ذات معاملة الأغراض البرمجية البعيدة، مما يؤدي إلى تعقيد الاستدعاءات وخاصة إذا كان معظم الأغراض البرمجية المطلوبة موجودة محلياً.

- النقص في التصميم، لم يكن هناك تنظيم للعروض المختلفة المقدمة ولم يتم تنسيق المشاكل لمعالجتها ومع ذلك تم اصدار المعايير القياسية مما أدى لتعقيد المواصفات وصعوبة تنفيذها وفي بعض الأحيان كانت غامضة. كما وأدى ظهور بعض الخلافات في اللجنة المصممة إلى ظهور طبعات منقّحة من معايير كوربا كانت مستحيلة التنفيذ دون وجود اضافات خاصة لها.

- مشكلات في التنفيذ والتحقيق، غالباً كانت النسخ المحققة قليلة أو غير كاملة أو غير كافية. على الرغم من أن النسخ المحققة جيداً من كوربا كان صعب الحصول عليها سابقاً إلا أن SUN Java SDK يأتي معها كوربا ضمنياً. كما وأن هناك بعض النسخ المحققة من كوربا قد تكون معقدة أو بطيئة أو غير متوافقة أو عير كاملة.

- الجدار الناري Firewalls، كوربا وبالتحديد IIOP تستخدم اتصال TCP/IP خام لنقل المعطيات، فإذا كان الزبون client موجود خلف بيئة firewall أو transparent proxy server قوية تسمح فقط بتمرير HTTP connections عبر الميناء 80، عندئذ التواصل سيكون مستحيل إلا إذا أعطى مخدم البروكسي المطلوب السماحية لـ HTTP CONNECT method أو SOCKS connections.

لقد كان من الصعب اجبار التحقيق على استخدام port قياسي واحد فقد كان يتم اختيار عدد من الـ ports بشكل عشوائي، وللآن الـ ORBs لديهم هذه العلة. بسبب هذه الصعوبات ازداد عدد المستخدمين الذين لجئوا لاستخدام خدمات الويب web services بدلا من استخدام كوربا CORBA حيث تتواصل باستخدام XML/SOAP على المنفذ 80 الذي يُترَك مفتوح عادةً. بعض النسخ الحديثة المحققة من كوربا تدعم SSL ويمكن إعدادها بسهولة لتعمل على رقم ميناء port واحد فقط.

ومن أشهر الـ ORBS المفتوحة المصدر مثل TAO و JacORB تدعم GIOP باتجاهين مما يعطي كوربا ميزة القدرة على اتصالات استعمال الاستدعاء الرجعي callback communication بدلا من صفة polling approach الموجودة في خدمات الويب.

كما أنه أصبح هناك عدد من الـ CORBA-friendly firewalls الصديقة لكوربا متوفرة في السوق.

- لا وجود للضغط compression، إن كوربا ترتب معطياتها على شكل معطيات ثنائية (صفر وواحد) ولكنها لا تستخدم أي ضغط. وحالياً شركات IONA/Remedy IT/Telefonica يعملون على إنشاء إضافة لمعيار كوربا القياسي يسمح بتحقيق الضغط وتسمى هذه الإضافة بـ ZIOP وسوف يقدم به طلب رسمي لمجموعة OMG في اجتماعها القادم.

- تجاهل التعامل مع الـ threads.

- لا يوجد تقابل للغات برمجة متل C# حيث أن كوربا تجاهلت.Net.

وغيرها العديد من المشكلات التي أدت إلى تراجع أهمية كوربا مع الزمن، وأيضاً من أسباب اندثار كوربا هو ظهور منافسة لها من قبل عدد من التقنيات مثل DCOM التابعة لشركة مايكروسوفت و EJB وخدمات الويب Web Services.

**OpenCCM (Open CORBA Component Model**

يعتبر OpenCCM أول تحقيق عام ومفتوح المصدر لمواصفات CORBA Component Model المعرّفة حسب مجموعة Object Management Group (OMG). يسمح OpenCCM بتصميم وتحقيق وترجمة وترزيم وتجميع ونشر وتنصيب واعداد وتنفيذ وإدارة وإنشاء تطبيقات موزعة تعتمد على مكونات كوربا CORBA Component. يعتبر OpenCCM مشروع مجاني ومفتوح المصدر معرّف تحت رخصة جنو LGPL ومستضاف من قبل Object Web Constium.

يعتمد OpenCCM على الجافا مما يسمح له بالعمل على عدة منصات كما ويعمل على منتجات كوربا التالية:

* Borland Enterprise Server (BES)
* JacORB
* ORBacus
* The Community OpenORB

كما ويعمل OpenCCM تقريبا على كل أنظمة التشغيل مثل:

* Linux
* Linux Familiar
* MacOS X
* Solaris
* Windows NT/2000/XP
* Windows NT/2000/XP with CYGWIN
* Windows CE for Personal Data Assistants (PDA)

ولا زال OpenCCM قيد التطوير حيث أنه لا يتمتع بجميع ميزات CCM المعروفة.

**معماريات حديثة في الانظمة الموزعة**

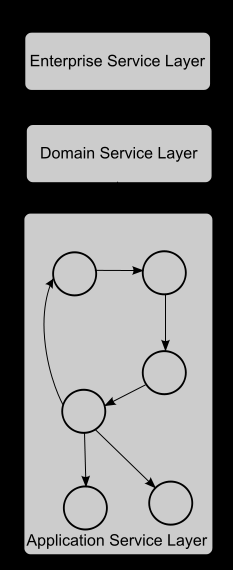
* معمارية الند ب الند

وتقوم فكرة نظام الند للند Peer-to-Peer على مبدأ تجميع مجموعة من الحواسيب المتصلة فيما بينها بواسطة شبكة

 Network بحيث يعمل كل حاسب من هذه الحواسيب كمزود للحواسيب البقية وتزويدهم بالبيانات التي يحتاجونها، وفي نفس الوقت يعمل هذا الحاسب كزبون عند طلب بيانات من الحواسيب البقية

* + معمارية التوجيه بالخدمه**[Service Oriented](http://www.google.com/url?sa=t&source=web&cd=34&ved=0CCcQFjADOB4&url=http%3A%2F%2Fwww.iritn.com%2FITshow-news-9942-%25D9%2585%25D8%25B9%25D9%2585%25D8%25A7%25D8%25B1%25D9%258A%25D8%25B3%25D8%25B1%25D9%2588%25D9%258A%25D8%25B3%2B%25DA%25AF%25D8%25B1%25D8%25A7%2BService%2BOriented%2BArchitecture%2B%25DA%2586%25DB%258C%2B%25D8%25B3%25D8%25AA%25D8%259F.htm&ei=nmE3TZeNCJKn8QPQvr3ADA&usg=AFQjCNHdGdgquEdCXOyEVtkwtUR1-dto8A)**[Architecture](http://www.google.com/url?sa=t&source=web&cd=34&ved=0CCcQFjADOB4&url=http%3A%2F%2Fwww.iritn.com%2FITshow-news-9942-%25D9%2585%25D8%25B9%25D9%2585%25D8%25A7%25D8%25B1%25D9%258A%25D8%25B3%25D8%25B1%25D9%2588%25D9%258A%25D8%25B3%2B%25DA%25AF%25D8%25B1%25D8%25A7%2BService%2BOriented%2BArchitecture%2B%25DA%2586%25DB%258C%2B%25D8%25B3%25D8%25AA%25D8%259F.htm&ei=nmE3TZeNCJKn8QPQvr3ADA&usg=AFQjCNHdGdgquEdCXOyEVtkwtUR1-dto8A)

اغلب الأنظمة الحديثة تتبع ما يسمى SOA أو Service oriented architechture وهي معمارية تصميم تطبيقات تتلخص في اعتماد أجزاء البرنامج كخدمات بدلا من أن تكون في شكل دوال ، أي أن النظام المتكامل يتكون من خدمات انترنت web services تقوم بحساب وأداء الأعمال المطلوبة منها ومن ثم تسليم النظام النتائج، مثلا، تخيل معي نظام حسابات في مؤسسة لها أفرع كثيرة، برنامج كل قسم حسابات في كل فرع يقوم بإعداد تقرير الحسابات النهائي في الفرع، ومن ثم يسلمه إلى الرئاسة التي طلبته عبر نداء دالة الweb service الخاصة بالفرع، لتقوم حسابات الرئاسة بإعداد التقرير النهائي لحسابات لكن يبقى الأهم هو ضمان سرية البيانات المنقولة عبر الweb service ، وضمان أن من سيستخدم الweb service مصرح له بالاستخدام.

تعتبر معمارية SOA مجموعة من مبادئ التصميم المرنه التي تستخدم في مرحلة تطوير البرمجيات وتكاملها ,البرمجيات التي تعتمد على هذه المعمارية تقسم وظيفيا بحيث تكون كل وظيفة قابلة للنقل وممكن استخدامها من قبل انظمة اخرى يبين الشكل

الذي في اليمينLayer interaction in service-oriented Architecture

**متطلبات معمارية خدمات الويب**  
  
لكي تتمكن من استخدام خدمات الويب بكفاءة، يجب أن تلبي معماريتها المتطلبات التالية  
     \* التشغيل المتبادل بين الأنظمة المختلفة ولغات البرمجة التي توفر الأساس للتكامل بين التطبيقات على منصات عمل مختلفة من خلال بروتوكول الاتصال.   
     \* الرغبة في إنشاء اتحاد للموارد

**بنية خدمة الويب تتطلب ثلاث عمليات أساسية هي :**

* نشر الخدمة
* إيجاد الخدمة
* طلب الخدمة

وتعمل وفق بروتوكولات قياسية تتعاون فيما بينها للوصول إلى خدمة web متكاملة

**اولاً:نشر الخدمة**

يقوم مقدم خدمة الوب بنشر الخدمة على شبكة الإنترنت لدى موزع خدمات الوب (قد يكون الموزع يقدم خدمات وب مطورة من قبل عدد من الشركات ) , إن عملية النشر تجري بمساعدة تقنية (web services description language) WSDL يحدث فيها توصيف لخدمة الوب المقدمة , كآلية طلب هذه الخدمة , ونوع متحولات الدخل التي تتطلبها , والخرج الذي ينتج عنها , الخ . أي إنها تعبر الاتفاق الذي يحكم آلية التعامل مع خدمة الوب , تجري كتابة WSDL باستخدام لغة XML

**ثانياً :إيجاد الخدمة**

عندما يحتاج أحد المستخدمين إلى خدمة معينة , يقوم بعملية بحث عن الخدمات المتوفرة لدى موزعي الخدمة , تجري هذه العملية باستعمال تقنية : UDDI (Universal description, Discovery and integration) التي تعمل دليلاً يجري فيه تسجيل خدمات الوب , ولاحقاً يجري فيه البحث عنها . يحوي كل تسجيل عمل UDDI المكونات الثلاث التالية :

* الصفحات البيضاء : Address,Contact,Know indentifiers
* الصفحات الصفراء : تصنيف صناعي مستند على قياس taxonomies
* الصفحات الخضراء : معلومات تقنية حول خدمات معروضة للاستخدام

**ثالثاً : طلب الخدمة**

هو عبارة عن وسيلة لطلب خدمة معينة فبعد إيجاد الخدمة المطلوبة يقوم طالبها بطلبها من مقدم الخدمة إن عملية تبادل المعلومات تجري بواسطة البروتوكول SOAP (Simple object access protocol) وهو معيار يعتمد على لغة XML ويحقق للبرامج تبادل المعطيات بفضل بروتوكول HTTP وهذا يجعله يتميز بما يلي :

* مصمم ليعمل على شبكة الإنترنت
* مستقل عن نظام التشغيل
* مستقل عن لغات البرمجة المستخدمة لتطوير التطبيقات

يقوم كلاً من UUDDI , WSDL , SOAP بالتعاون فيما بينهم للحصول على عمل متكامل وقياسي (Web services stander ) وفقاً لمل يلي :

* تحديد لغة التخاطب WSDL بين التطبيقات التي تريد التواصل مع بعضها وينشأ ما يدعى بيئة الخدمة services interfaces ولغة التخاطب القياسية هي لغة XML , إن عملية النشر للخدمات تجري بمساعدة تقنية . WSDL (Web services Description language)
* الخطوة التالية هي استخدام UDDI (Universal description , Discovery , and integration ) لتسجيل توصيف الخدمة المقدمة من قبل مزود خدمات التطبيقات وذلك باستخدام Register , look – up protocol
* أما الآن فيتم تفعيل التخاطب بين التطبيقات وذلك بتمكين التطبيق الزبون ( Requester Application ) من استخدام (Object ) المعروضة من قبل التطبيق المزود للخدمة (Provider Application ) وذلك باستخدام SOAP (Simple Object Access Protocol ) ويبدا التخاطب عن طريق تمرير (object ) فيما بينهما.

الفصل الثالث

**الحوسبة السحابية**

**الحوسبة السحابية:Cloud Computing**

هي تكنولوجيا تعتمد على نقل [المعالجة](http://ar.wikipedia.org/wiki/%D9%85%D8%B9%D8%A7%D9%84%D8%AC%D8%A9) ومساحة التخزين الخاصة [بالحاسوب](http://ar.wikipedia.org/wiki/%D8%AD%D8%A7%D8%B3%D9%88%D8%A8) إلى ما يسمى السحابة وهي جهاز [خادم](http://ar.wikipedia.org/wiki/%D8%AE%D8%A7%D8%AF%D9%85_(%D9%85%D8%B9%D9%84%D9%88%D9%85%D8%A7%D8%AA%D9%8A%D8%A9)) يتم الوصول اليه عن طريق [الانترنت](http://ar.wikipedia.org/wiki/%D8%A7%D9%84%D8%A7%D9%86%D8%AA%D8%B1%D9%86%D8%AA). بهذا تتحول برامج [تكنولوجيا المعلومات](http://ar.wikipedia.org/wiki/%D8%AA%D9%83%D9%86%D9%88%D9%84%D9%88%D8%AC%D9%8A%D8%A7_%D8%A7%D9%84%D9%85%D8%B9%D9%84%D9%88%D9%85%D8%A7%D8%AA) من [منتجات](http://ar.wikipedia.org/wiki/%D9%85%D9%86%D8%AA%D8%AC%D8%A7%D8%AA) إلى [خدمات](http://ar.wikipedia.org/wiki/%D8%AE%D8%AF%D9%85%D8%A7%D8%AA). من أهم فوائد هذه التكنولوجيا هي إبعاد مشاكل صيانة وتطوير برامج تقنية المعلومات عن الشركات المستخدمة لها وبالتالي يتركز مجهود الجهات المستفيدة على استخدام هذه الخدمات فقط.

تعتمد البنية التحتية للحوسبة السحابية على [مراكز البيانات](http://ar.wikipedia.org/wiki/%D9%85%D8%B1%D9%83%D8%B2_%D8%A8%D9%8A%D8%A7%D9%86%D8%A7%D8%AA) المتطورة ولتي تقدم مساحات تخزين كبيرة للمستخدمين كما أنها توفر بعض البرامج كخدمات للمستخدمين. وهي تعتمد في ذلك على الإمكانيات التي وفرتها تقنيات [ويب 2.0](http://ar.wikipedia.org/wiki/%D9%88%D9%8A%D8%A8_2.0).

مع تطور التقنيات المتاحة من خلال شبكة الويب بظهور الويب 2,0 والويب 3,0 والزيادة المطردة في سرعات الانترنت المتاحة للمستخدمين اتجهت العديد من المؤسسات إلي إتاحة تطبيقاتها للاستخدام من خلال شبكة الانترنت باستخدام الحوسبة السحابية، حيث أتاحت هذه التقنية لمستخدميها مميزات أفضل مثل توفير النفقات أو إتاحة خدمات لقطاع اكبر من المستفيدين. ولم تكن المكتبات في منأى عن الاستفادة من تلك التقنيات فاتجهت بعض المكتبات إلي الاشتراك في مشاريع الحوسبة التي أتيحت من خلال بعض المؤسسات التي تخصص خدماتها للمكتبات مثل مشروع Dura Cloud .

**تاريخ**

السحابة هو تعبير كان يستخدم في البداية للإشارة إلى الإنترنت وذلك في مخططات الشبكات. وجاءت فكرة البرامج كخدمات في الستينيات من القرن العشرين حيث ان [جون مكارثي](http://ar.wikipedia.org/wiki/%D8%AC%D9%88%D9%86_%D9%85%D9%83%D8%A7%D8%B1%D8%AB%D9%8A_(%D8%B9%D8%A7%D9%84%D9%85)) قد عبر عن الفكرة بقول "قد تنظم الحوسبة لكي تصبح خدمة عامة في يوم من الأيام". الحوسبة السحابية (cloud computing) هي أحد اساليب الحوسبة، يتم فيها تقديم الموارد الحاسوبية كخدمات، ويتاح للمستخدمين الوصول إليها عبر شبكة الإنترنت (”السحابة“)، دون الحاجة إلى امتلاك المعرفة، الخبرة، أو حتى التحكم بالبنى التحتية التي تدعم هذه الخدمات. يمكن النظر إلى الحوسبة السحابية على أنها مفهوم عام يشمل البرمجيات كخدمة (Software as a Service)، الويب 2.0، وغيرها من التوجهات الحديثة في عالم التقنية التي تشترك في فكرة الاعتماد على شبكة الإنترنت لتلبية الاحتياجات الحوسبية للمستخدمين.

نشرت مجلة Technology Review مؤخرا مقالا يتحدث عن أدوات الحوسبة السحابية مفتوحة المصدر. سنستعرض في هذه البحث الحاجة إلى مثل هذه الأدوات، ومالذي بإمكانها تقديمه.الحاجة إلى أدوات حوسبة سحابية مفتوحة المصدر

منصات الحوسبة السحابية التجارية مثل [Amazon EC2](http://ar.wikipedia.org/w/index.php?title=Amazon_EC2&action=edit&redlink=1)، Microsoft Azure Services Platform، وGoogle App Engine تعطي العديد من الشركات مرونة الوصول إلى الموارد الحاسوبية اللازمة، وتساهم في تقليل تكاليف البنية التحتية للشركات الناشئة، لكن الاعتماد عليها لا يخلو من الشوائب. فكثير من المستخدمين يثيرون مواضيع مثل الخصوصية، الأمن، والموثوقية عندما يدور الحديث عن “السحابة“. لذلك، هناك اهتمام متزايد بأدوات الحوسبة السحابية مفتوحة المصدر، التي تمكن الشركات من بناء وتخصيص “سُحُبهم” الحوسبية لتعمل بجانب الحلول التجارية الأقوى.

**فوائد الحوسبة السحابية**

\* تُمكن المستخدم من الدخول على ملفاته وتطبيقاته من خلال هذه السحابة دون الحاجة لتوفر التطبيق في جهاز المستخدم , بالتالي تقل المخاطر الأمنية وموارد العتاد المطلوبة وغيرها . ومنها الاستفادة من الخوادم الضخمة جداً في إجراء عمليات معقدة قد تتطلب أجهزة بمواصفات عاليه .

\* توفر الكثير من المال اللازم لشراء البرمجيات التي يحتاجها المستخدم فكل ما يحتاجه هو جهاز حاسب متصل بخط انترنت سريع وأن يكون متصل بأحد المواقع التي تقدم البرمجيات التي يحتاجها .

\* سهولة الوصول إلي التطبيقات المتاحة من خلال تلك التقنية من خلال أي حاسب متصل بشبكة الانترنت .

\* توفير عدد العاملين في علي صيانة النظام والبرمجيات .

**الشركات المقدمة للخدمة**

من أمثلة الشركات المقدمة للخدمة : [جوجل](http://ar.wikipedia.org/wiki/%D8%AC%D9%88%D8%AC%D9%84)، [مايكروسوفت](http://ar.wikipedia.org/wiki/%D9%85%D8%A7%D9%8A%D9%83%D8%B1%D9%88%D8%B3%D9%88%D9%81%D8%AA), [صن](http://ar.wikipedia.org/wiki/%D8%B5%D9%86)، [أمازون](http://ar.wikipedia.org/wiki/%D8%A3%D9%85%D8%A7%D8%B2%D9%88%D9%86_(%D9%85%D9%88%D9%82%D8%B9)). الحوسبة السحابية في تطور مستمر، فمعظم الشركات تحاول جاهدة لتقديم خدمة افضل

**عيوب الحوسبة السحابية**

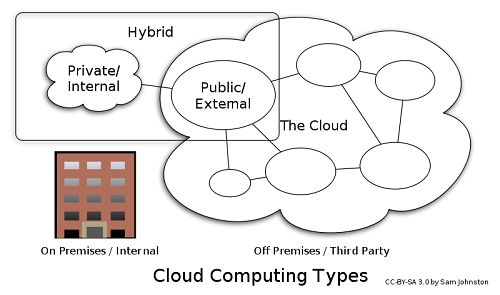
\*تعد مشكلة توافر الانترنت هي أحد المشاكل الرئيسية خصوصاً في الدول النامية حيث تتطلب الخدمة توفر الاتصال بشبكة الإنترنت بشكل دائم أثناء استخدام تلك الخدمة .

\* تعد مشكلة حماية حقوق الملكية الفكرية أحد المشاكل التي تثير مخاوف مستخدمي تلك الخدمات، فلا يوجد ضمانات بعدم انتهاك حقوق الملكية الفكرية للمستخدمين

\* أيضا مشكلة أمن وخصوصية المعلومات، فبعض المستخدمين يتخوفون من احتمالية اطلاع أفراد آخرين على معلوماتهم .

لم تعد هناك شركة بفروعها الا ومتصلة بالأنترنت لتصبح شبكة الانترنت هي الوسيط الذي يربطها جميعا مع نفسها ومع الاخرين ،ومن هنا جاءت فكرة استغلال هذا الوسيط لتوفير خدمات مختلفة للشركات بفروعها المختلفة والموزعة على مستوى العالم وبالتالي مساعدة الشركات على توفير مبالغ ماليه ضخمه كان سيتم صرفها لتوفير تلك الخدمات داخليا بالشركه  فمثلا كانت الشركة ستصرف هذه المبالغ على شراء معدات وبرمجيات وتوفير طاقم من المهندسين والمحترفين لصيانتها و ادارتها بل و حتى دفع فواتير التشغيل من كهرباء و الخ  . مما كان يزيد التكلفة الإجمالية لامتلاك تقنيه ما total cost of ownership TCO\* .

اذن الحوسبة السحابية هي توفير خدمات مختلفة و متعددة للشركات عبر الانترنت و يكون الدفع حسب الطلب وحسب الخدمة المقدمة من موفر الخدمات هذه و الذي نسميه موفر الخدمة السحابية Cloud Service Provider  مما يعجل في ROI return on investments\*\*  ويساهم بشكل كبير كحل من حلول معالجة الازمه الاقتصادية العالمية. تتوفر ال cloud  في 3 اشكال هي Private,Public,Hybrid.



ال Cloud نفسها ما هي الا مجموعة  Hardware , Networks , Storage Units , Services , Software يمكن توفيرها للعملاء من شركات مختلفة التخصص عبر الانترنت  و كل هذا بالإضافة لما توفره الغيمة من قدرة على التوسع Scability حسب رغبة العميل في اي وقت و مرونة Flexibility في الاداء.

[**نوعية الخدمات التي يوفرها مقدّم الخدمة السحابية**](http://www.arabhardware.net/services/distributed-computing/663-cloud-computing.html?start=1)

وجد 3 انواع رئيسيه من الخدمات يمكن توفيرها من قبل موفر الخدمة السحابية للعملاء (cloud services delivery models)   او XaaS :

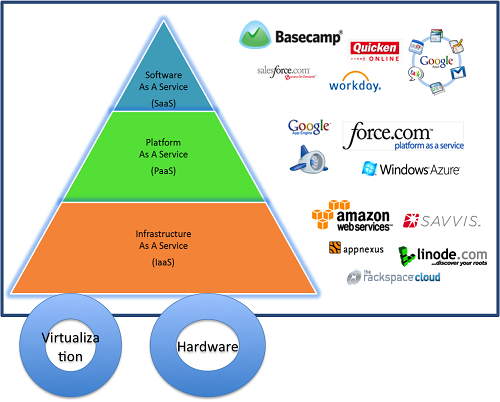
**1.** Infrastructure as a Service IaaS  
**2.**Platform as a Service PaaS  
**3.** Software as a Service SaaS

الان دعونا نتعرف بشكل بسيط على كل نوع :

**IaaS**هي توفير تقنيات شبكيه وعتاد و مراكز بيانات للعملاء و يتضمن هذا استخدام تقنيه ال Virtualization  وتوفير نظم تشغيل يمكن الدخول اليها عبر الانترنت و يعتبر AMAZON EC2 من موقع امازون نموذج لهذا النوع و الذي يوفر حتى نظم تشغيل افتراضيه و خدمات اخرى عديده و كذلك نموذج اخر هو شركات ايجار الروترات والسويتشات لدراسي ال CCIE مما يخلق مصطلح الحوسبة عند الطلب او On Demand Computing  .

**PaaS**وهي توفير كل ما يحتاجه المطوريين لبناء تطبيقات و برمجيات و خاصة Web Based Applications  من خلال توفير أدوات تطويريه في بيئة قياسيه standard .

**SaaS**و في هذا النوع نسمي موفر الخدمة السحابية ب ASP او Application Service Provider ويتم فيها توفير برمجيات أدارة علاقات العملاء CRM على سبيل المثال و يمكن اعتبار موقع salesforce.com نموذج لهذا النوع و مثال عليه  و في الحقيقة يوجد نماذج اخرى عديده تعرفونها جيدا مثل facebook , ebay , skype  فمثلا الفيس بوك يوفر شبكه اجتماعيه للموظفين في شركه بدلا من استخدام برنامج خاص على الشبكة الداخلية ليتبادلو الآراء في امور العمل او تبادل ملفات العمل الهامه عموما في هذا النوع نجد حضور هام لتقنيات التأمين و اهتمام من ال ASP بتوفيرها بداخل خدماتهم لحماية بيانات العملاء عبر ال VPN  وغيرها



تم استحداث انواع اخرى للحوسبة السحابية و منها على سبيل المثال وليس الحصر :

**Communication as a service CaaS**

وفيها يطلب العميل من موفري الخدمة بتعهيد Outsourcing حلول خدمات اتصالات , فمثلا توفير خدمات VOIP  ,real time presence و خدمات المؤتمرات الفيديويه و على موفري الخدمة السحابية من هذا النوع   بضمان توفير جودة خدمه تطابق مع ال QoS المذكورة في أتفاقية مستوى الخدمه بين الموفر و العميل service level agreements   SLA وضمان ادارتها بشكل مستقل عن قسم ال IT لدى العميل .  يمكن اعتبار  نموذج على هذا النوع حلول WEBEX من شركة سيسكو .

**Monitoring as a Service MaaS**  
توفير خدمات مراقبه وحماية البيانات تتماشى مع يريده العميل والعميل بدوره قد يحتاج هذه الخدمات بناء على متطلبات حكومية لنوعية الشركات التي يمثلها هذا العميل و يقوم موفر الخدمة هنا بتوفير باقات مختلفة للشركات منها توفير الاكتشاف المبكر لنقاط الضعف Vulnerabilities  لدى العميل ونظم تشغيله .   
  
اخيرا مجرد وجود صفحه للشركة (العميل) لدى موقع الفيس بوك هو تطبيق ونموذج للحوسبة السحابية في أبسط صورها وساعدت  الحوسبة السحابية على التعجيل في الوصول لحل للأزمة الاقتصادية العالمية من خلال توفير مبالغ ماليه ضخمه كبيره على الشركات و في نفس الوقت توفير فرص عمل و دخل أكبر لشركات ال IT الموفرة للخدمات السابقة الذكر ،وخلال الفترة القادمة سيظهر  جيل جديد من مواقع الانترنت نسميه الجيل الثالث او WEB 3.0 وستكون مواقع تفاعليه وخدميه اكثر تطورا وسترتبط بالحوسبة السحابية  بشكل كبير مما قد يغير من مفهومنا لشبكة الانترنت .