السيد : محمود فايد
MSFCLIPPER@HOTMAIL.COM

ثورة فى علم البرمجة
نمط برمجة جديد اقوى من برمجة الكائنات
محيط تطوير الدبل اس (الخادم الممتاز)
والبرمجة من غير كود

DoubleS (Super Server)

DoubleS

www.sourceforge.net/projects/doublesvsoop

مقدمة :

يشهد علم البرمجة منذ بدايته تطورا سريعا متعدد الاتجاهات فهو يشدو نحو راحة المستخدم وراحة المبرمج واستقرار بيئة العمل وخلاف ذلك من المواصفات المطلوبة لتوفر بيئة تطوير مناسبة.

ورغم ذلك التطور السريع الا ان الجميع يعلم ان هناك قفزات نوعية محددة يرجع اليها هذا التطور الهائل ولعل اهم هذه القفزات ما قام بيه الدكتور الان كيى Dr Alan Kay والذى اخترع البيئة الرسومية GUI وبرمجة الكائنات OOP والحاسب المتنقل Lap Top وكان لهذه الابتكارات اثرها المعلوم على علم برمجة الحاسب.

وسوف نتطرق فى الحديث قليلا عن الثورة التى احدثتها برمجة الكائنات والتى اددت الى توفير الظروف الملائمة لتطوير التطبيقات الاكثر تعقيدا عما كان من قبل بواسطة البرمجة التركيبية او الهيكلية Structure Programming ولعل اهم ما يميز التطبيقات الحديثة الان

البيئة الرسومية كواجهة للبرنامج 1 – Graphical User Interface

نظام ادارة الاحداث 2 – Event Driven System

نمط البرمجة (برمجة الكائنات) 3 – Object Oriented

تناقل الخدمات بين اجزاء البرامج (زبون – خادم) 4 – Client – Server

ولعل الجميع قد سلم بان برمجة الكائنات هى نمط البرمجة الامثل الذى يناسب مع مستوى التعقيد فى البرمجيات الحالية ***ولكن السوال الذى يطرح نفسه ماذا بعد هذا النمط وهل من انباء عن احتياج نمط برمجة جديد ؟***

فى الواقع اذا ظل مستوى التطبيقات التى نقوم بتطويرها على ما نحن عليه اليوم فلا نجد حاجة ملحة لتطوير وابتكار نمط برمجة جديد ولكن اذا ما حدث تغير فى مستوى التطبيقات المطلوبة فان الامر سوف يندرج تحت المثل الشهير (الحاجة منبع الاختراع).

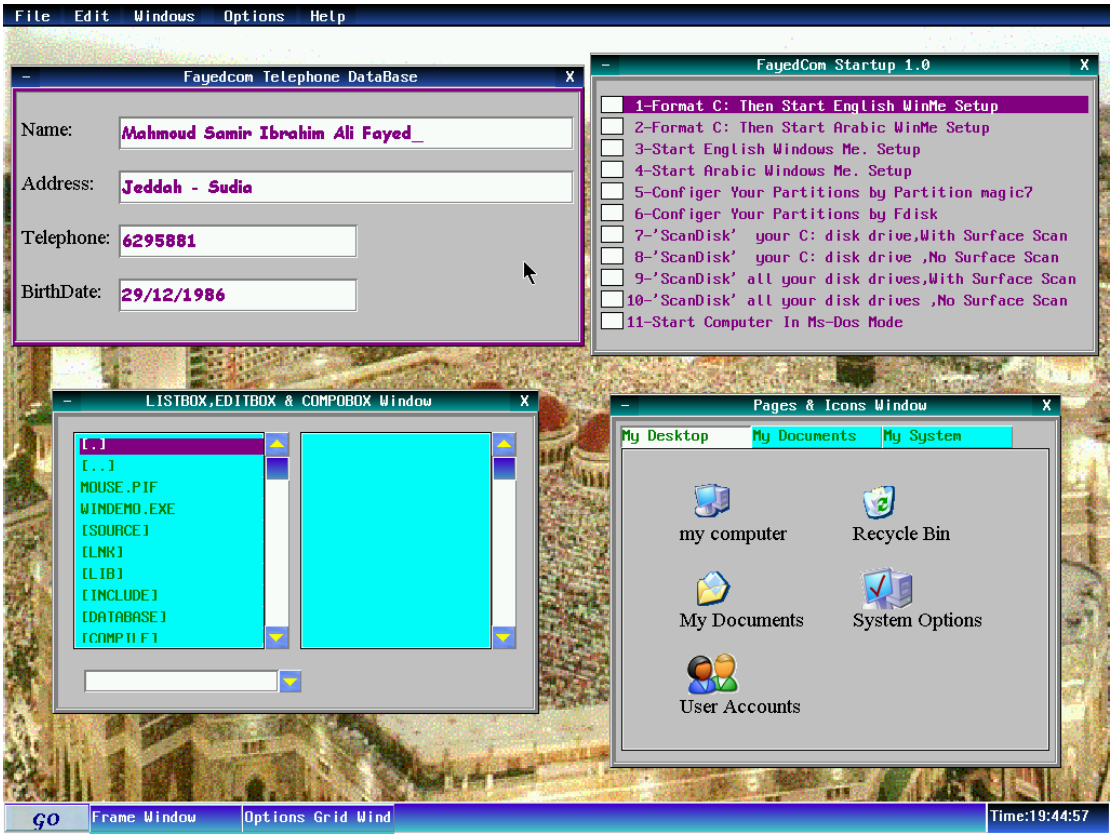
وعلى مستوى راحة المبرمج فان تطوير ادوات البرمجة يتجه نحو توفير الادوات التى تودى الى زيادة سرعة عمل التطبيقات من خلال ادوات توفر ما نسميه بالبرمجة من غير كود Programming Without Code من المحاولات فى هذا المجال برامج التصميم مثل مصمم النماذج Form Designer ومصمم التقارير Report Designer والذى يوفر الكثر من الكود بالاضافة الى المعالجات Wizards والتى توفر العديد من الحلول النمطية الثابتة بدون اى مجهود يذكر.

ولكن هل بالامكان ان تتطور كل تلك الادوات وتتضافر معا لكى نحصل على لغة برمجة جديدة مئة بالمئة من غير كود ؟ ام ان هذا حلم لن يتحقق !

ان عملية الحصول على لغة برمجة كاملة بدون اى اكواد وتوفر الفرصة لتطوير كافة التطبيقات التى نقوم بعملها الان امر صحيح من الناحية النظرية ولكن تطبيقه يعد غاية فى الصعوبة وقد يستحيل عمليا حتى الان لان ذلك يتطلب على الاقل مئات الالاف من المبرمج وهو ما يستحيل ان تستطيع توفيره اكبر الشركات فى العالم حتى الان .

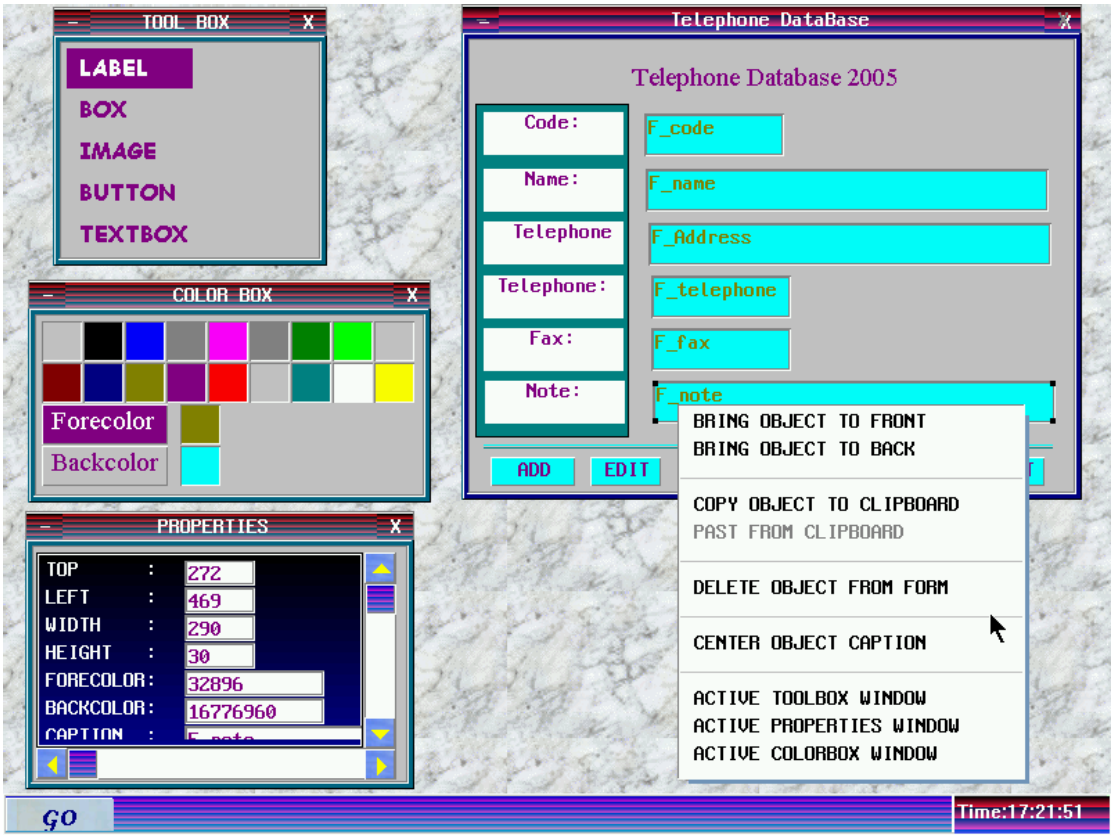
ولعل البعض يسال اذا ما كانت هناك لغة من غير كود فكيف تكون طبيعة تلك اللغة وما هو البديل للكود للحصول على كافة النتائج المطلوبة ؟
فى الواقع ان الاجابة عن هذا السوال تتوفر فى ان هذه اللغة سوف تتمتع ببئة تطوير تكاد تبدو من اول وهلة خيالية حيث انك سوف تجد تحت يدك المئات من برامج التصميم بالفارة (واستخدام لوحة المفاتيح فقط لادخال البيانات الازمة) بالاضافة الى المئات من المعالجات التى توفر كافة الاحتمالات البرمجة (باختلاف مستوى البرنامج الذى يتم تطويره هل هو نظام تشغيل ام لغة برمجة ام احد التطبيقات الاخرى).
وتوفر مثل هذه اللغة ام غاية فى الصعوبة.
منذ عدة سنوات اتخذ العديد من علماء الحاسب وقت عملهم ثمرة لكى يحصلوا على تصور للخطوات التى يمكن عملها من اجل جعل الحاسب الة ذكية تصل الى القدرة على التعامل مع الكلمات الطبيعية ومحاولة معالجتها بل ان البعض يتصور بامكانية جعل الحاسب يفهم اللغة الطبيعية مئة بالمئة حتى يمكننا برمجته بلغتنا العادية بدون الحاجة للغة برمجة ولكن ذلك التصور رغم تفائلة الا ان الاستخدام الافضل له يعد فى مجال تطوير الانسان الالى للقيام بكافة انشطة العمل المطلوبة فى اى مكان وليس فى امكانية تصور تطبيقات تجارية قد يصعب على العقلية البشرية تخيلها حيث ان تصميم التطبيقات الكبيرة يعد اكثر تعقيدا من برمجتها وهذا مايصعب على الانسان الالى الذى ان استطاع بقدرته الادراك والتعامل مع البشر فانه قد يصعب ان يعلو بمستوى تعليمه ليصبح مبرمجا.
ولهذا فان علم البرمجة ينتظر بفارغ الصبر اللغة الجديدة التى سوف تتيح للمستخدم العادى برمجة الحاسب والتعامل معه كصديق بدون مبرمج وسيط.
ولكن من سوف يقدم تلك الغة ؟
فى بداية عام ٢٠٠٦ اعلن المبرمج المصرى السيد : محمود فايد من خلال الموقع الشهير www.sourceforge.net عن المشروع DoubleS والذى يقدم نمط برمجة جديد اقوى من نمط برمجة الكائنات ويمكن استخدامه فى تطوير البرمجيات الاكثر تعقيد وان هذا النمط صمم خصيصا لكى يتم استخدامه فى عمل لغة برمجة جديدة مئة بالمئة من غير كود من خلال عدد مسموح به من المبرمجين ويمكن للشركات العظمى توفيره.
ويعنى ذلك انه سوف يتم تطبيق هذا النمط اولا بحيث يمكن استخدامه مع اللغات الحالية وبعد ذلك يتم استخدام تلك اللغات فى عمل لغة البرمجة الجديدة.
ان تلك اللغة الجديدة تعادل فى الحجم اكثر من ١٠٠ لغة برمجة حالية بالاضافة الى احتياجها الى اكثر من ١٨٠٠ مترجم صغير Simple Compiler ولكن داخل لغة واحدة مما يصعب عمليا برمجته باستخدام النمط الشهير برمجة الكائنات ولهذا توفر لنا النمط الجديد والذى يسمى الخادم الممتاز لكى يقوم بتلك المهمة.
بدا المبرمج محمود فايد تعلم البرمجة عام ١٩٩٧ على يد والده المهندس سمير ابراهيم والذى كان يتخصص فى عمل البرامج التجارية (انظمة قواعد البيانات) وفى عام ٢٠٠٠ بدا السيد محمود فايد الاتجاه بعيدا عن برمجة انظمة قواعد البيانات

وبالتحديد اتجه الى برمجة الجزء الاخير من نظام التشغيل وهو ما يتعلق ببرمجة نظام ادارة البيئة الرسومية والنوافذ المتحركة ونظام ادارة الاحداث ونظام تعدد المهام وكان يستخدم نمط برمجة الكائنات فى ذلك.
وفى عام ٢٠٠٥ اتحد مع المبرمج Ferns Paanakker فى تقديم المكتبة FGLIB والتى توفر بيئة رسومية وتعدد مهام ونظام نوافذ للغة كليبر من خلال الموقع www.sourceforge.net/projects/fglib وكان ذلك اول مساهمة منه فى البرامج مفتوحة المصدر وتقدر الشفيرة المصدرية التى قام بكتابتها باكثر من ١٧٠٠٠ سبعة عشر الف سطر مقسمة الى ٤٥ فصيلة Classes وكانت المكتبة تحتوى على مصمم نماذج Form Designer حوالى ٣٠٠٠ سطر.
ومن خلال ذلك المشروع لاحظ ما يمكن ان يصنعه تغير نمط البرمجة فى الانتاجية وتاثيره على البرنامج الذى يتم تطويره حيث ان مشروع كهذا يعد غاية فى الصعوبة والتعقيد اذا تم العمل به باستخدام البرمجة الهيكلية Structure Programming ويحتاج الى كم ضخم جدا من المبرمجين فى حين ان استخدام نمط برمجة الكائنات ازال الكثير من التعقيد .
ولما كان السيد محمود فايد يعلم ان لتغير نمط البرمجة هذا الاثر وجد ان الحصول على لغة البرمجة الكاملة بدون اكواد يتطلب ثورة جديدة ولا يمكن ان يتم ذلك من خلال خطوة واحدة بل بتطلب عدة خطوات للوصل الى الهدف المنشود.
وبالفعل بدا تصميم نمط البرمجة الجديد الخادم الممتاز مستفيدا بخبرة ٩ سنوات فى تطوير انظمة قواعد البيانات وخبرة ٥ سنوات فى تطوير نظام ادارة البيئة الرسومية وتعدد المهام ليقدم لنا هذا النمط الجديد الذى يستند على مفاهيم عديدة من مجالات مختلفة مثل الرياضيات و الشبكات والخوادم وتركيب الذرة والدوائر الكهربية والتفاعل البشرى كل هذه المفاهيم المتضاربة ذات المجالات المختلفة استطاع توظيفها لتخدم مجال واحد وهو نمط البرمجة الجديد الخادم الممتاز.
ولكن كما نعلم انه كلما تواجدت المميزات فانه هناك جانب اخر قد يخفى بعض العيوب ***ومن هنا كانت الصدمة الكبيرة حيث اتهم الكثير من المبرمجين المحترفين هذا النمط بانه يتسم باعلى مستويات التعقيد مما يجعل من الصعب استخدامه فى حين ان تعلمه ليس فى غاية الصعوبة بل بحاجة الى بعض الوقت*** .
ولهذا كانت المبادرة من السيد محمود فايد بالاتحاد مع عدد من المبرمجين بتوفير محيط تطوير يخدم هذا النمط ويساعد فى عملية التصميم وهو ما يسمى بـ DoubleS Framework وهو بيئة مريحة للعمل مع النمط الجديد وتوفر واجهة للمبرمج قادرة على التغلب على صعوبات الاستخدام وتعقيد التصميم مما يسمح بتخطى تلك المشكلة.

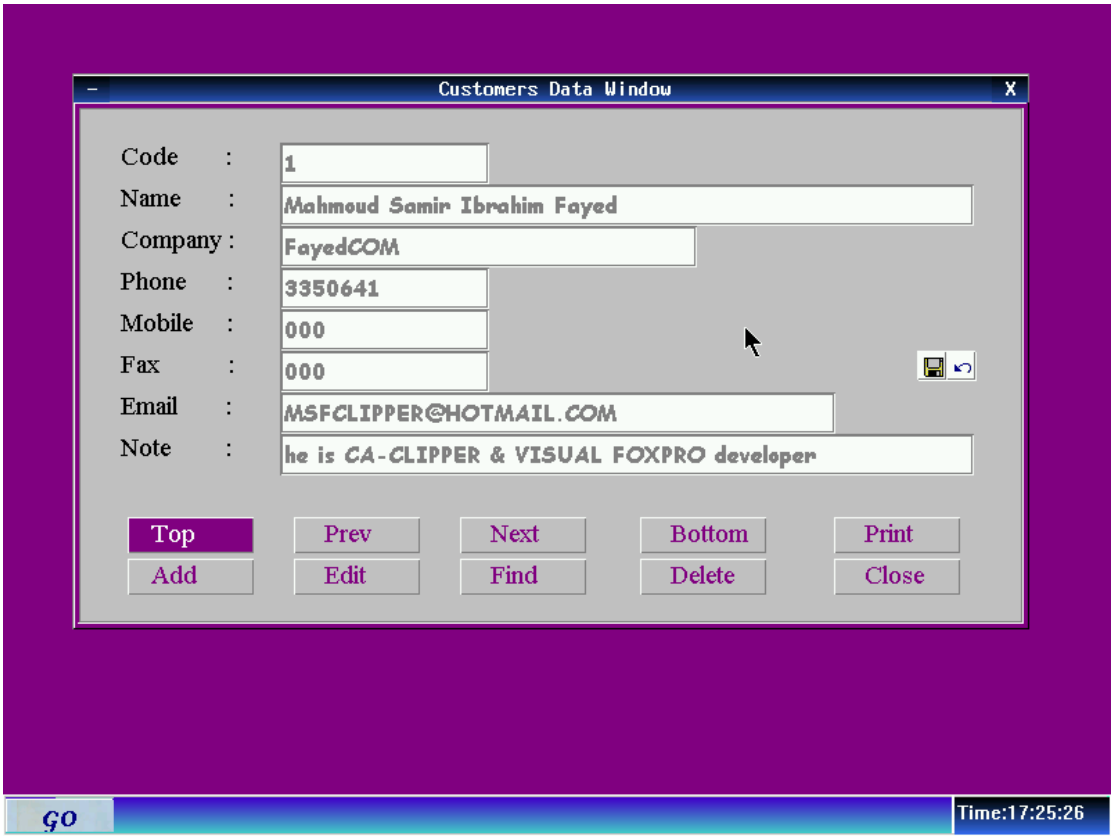


شكل ١ : نظام ادراة البيئة الرسومية- النوافذ – برمجة السيد : محمود فايد

شكل٢ : نظام ادراة البيئة الرسومية- القوائم المنسدلة – برمجة السيد : محمود فايد



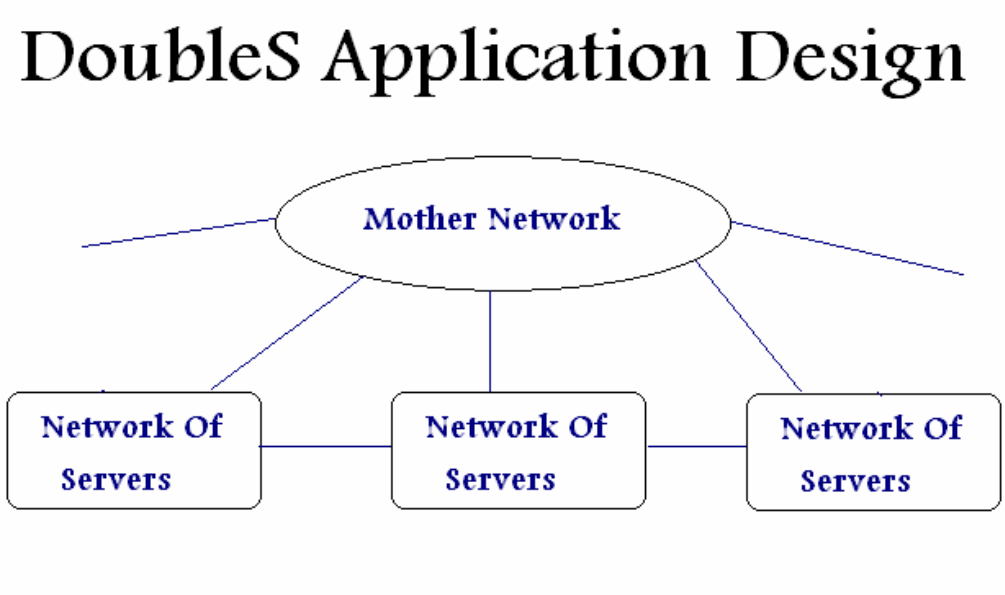
شكل ٣ : نظام ادراة البيئة الرسومية – مصمم النماذج – برمجة السيد : محمود فايد



شكل ٤ : نظام ادراة البيئة الرسومية- شاشة بيانات – برمجة السيد : محمود فايد

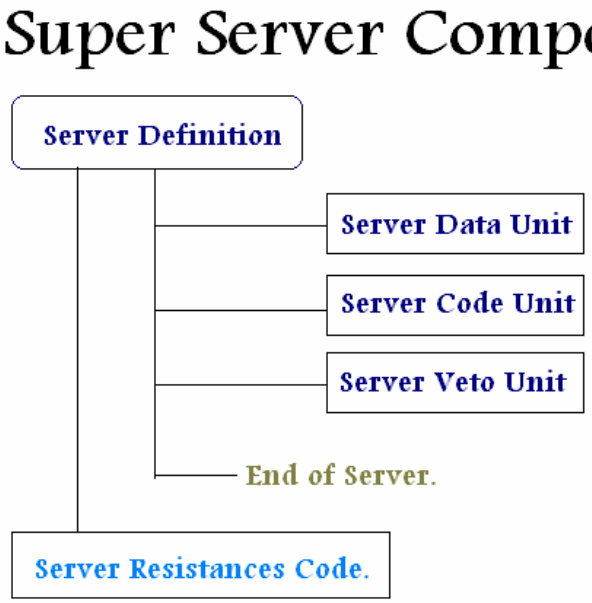
مدخل الى نمط البرمجة الخادم الممتاز :

يفترض هذا النمط ان البرنامج الذى تقوم بعمله مقسم الى مجموعة من الشبكات Networks بحيث ان كل شبكة تحوى بداخلها مجموعة من الخوادم Servers



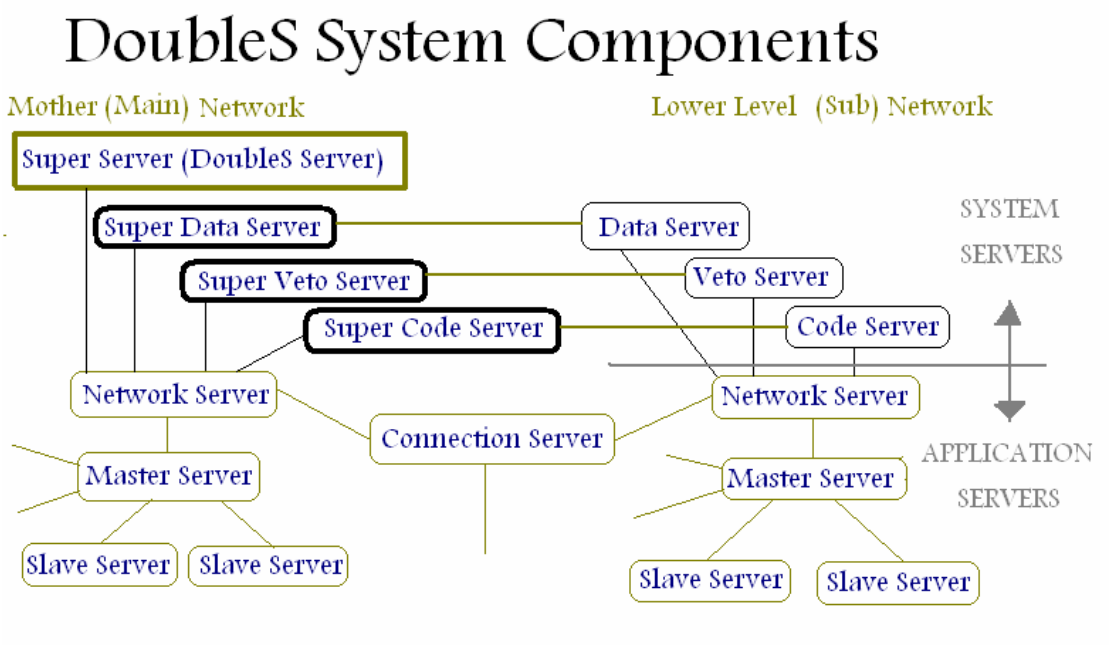
شكل ٥ : مكونات البرنامج فى نمط الدبل اس

ويشترط ان يكون بالبرنامج شبكة رئيسية بالطبع تكون هى بداية العمل ومدخل البرنامج (شى مشابه ل Main() فى لغة السى)
ومن هنا يتضح ان البرنامج يعود بمكوناته الى مجموعة من الخوادم Servers بحيث ان كل خادم مكون من مجموعة من الوحدات وبالتحديد ٣ وحدات واحدة للبيانات Data Unit والاخرى للكود Code Unit والثالثة للتصويت Veto Unit



شكل ٦ : مكونات الخادم الاساسية

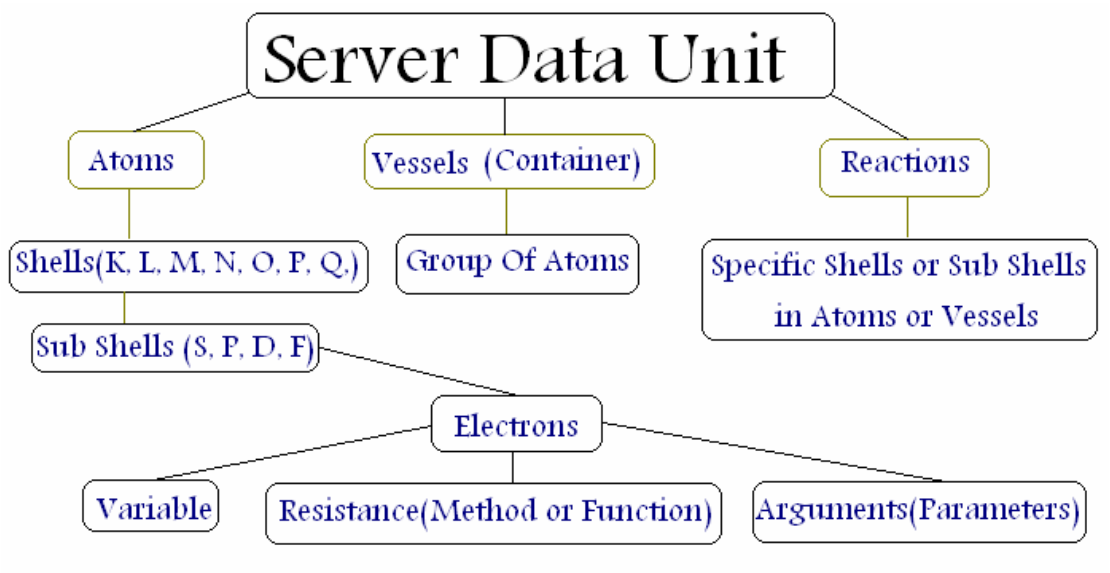
ولا يقتصر الامر على ذلك بل ان الخوادم انواع وهى فى الواقع ١٣ نوع لكل منها وظيفة محددة داخل المنظومة والتى يطلق عليها نظام دبل اس



شكل ٧ : مكونات نظام دبل اس

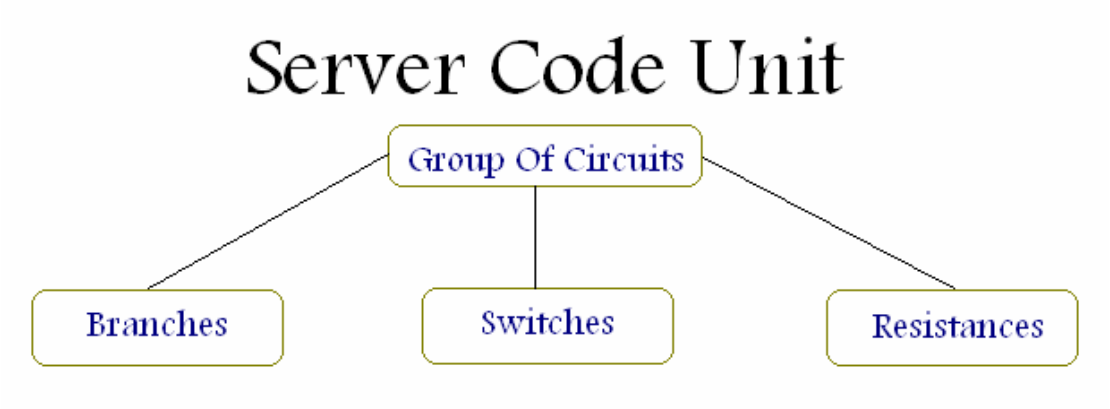
وفكرة هذه المنظومة فى ديناميكة عمل مجموعة من الخوادم معا لتودى وظيفة محددة حيث يتم داخل هذه المنظومة ليس فقط تنفيذ البرنامج بل انه يراقب نفسه البرنامج يراقب عملية تدوال البيانات وتنفيذ الكود واعطاء الخدمات وهو بذلك يتعاون مع نظام التشغيل ولغة البرمجة

وسوف نبدا الان التوضيح من خلال استعراض وحدة البيانات داخل الخادم Data Unit



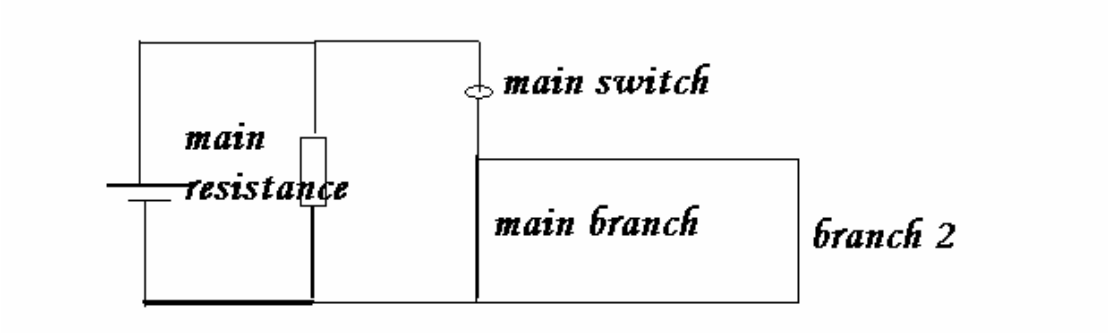
شكل ٨ : مكونات وحدة البيانات داخل الخادم

تتكون وحدة البيانات داخل الخادم من نظام كيميائى Chemical System متطور تقوم وظيفته على توفير نظام ادارة قاعدة بيانات تخيلى بمعنى Virtual DBMS مما يجعل لدى المبرمج قاعدة بيانات علاقية قوية بصورة مباشرة داخل الخادم Server مسئولة عن هياكل البيانات Data Structure داخل الخادم.
حيث تمثل الذرة ملف بيانات Data File لوكن تخيلى فى الذاكرة Ram وتتكون هذه الذرة من مجموعة من المدارات التى تحوى مدارات فرعية ومن ثم الالكترونات حيث ان الالكترون قد يكون المتغير (Variable) الذى نعرفه وقد يكون عبارة عن دليل Reference لتنفيذ حدث معين وقد يكون عبارة عن كتلة بيانات وليست متغير ويمكن الوصول لكتلة البيانات هذه من خلال اسم الذرة والمدار والمدار الفرعى
الوعاء Vessel/Container هو حاوى لمجموعة من الذرات لتجهيزها لعمل عملية محددة اما التفاعل Reaction فهو كفلتر Filter ولكن للالكترونات داخل Vessels او Atoms من خلال المدارات Shells او المدارات الفرعية Sub Shells
تمكن مميزات وحدة البيانات فى دعمها لتطوير التطبيقات التى تتسم ب Complex Data Structure من خلال توفير Virtual DBMS والسر الاكبر فى وحدة البيانات يرجع الى الالكترون الذى قد يمثل وحدة بيانات من نوعية غير معروفة وبحجم غير محدد
والان سوف ننتقل لاستعراض وحدة الكود :



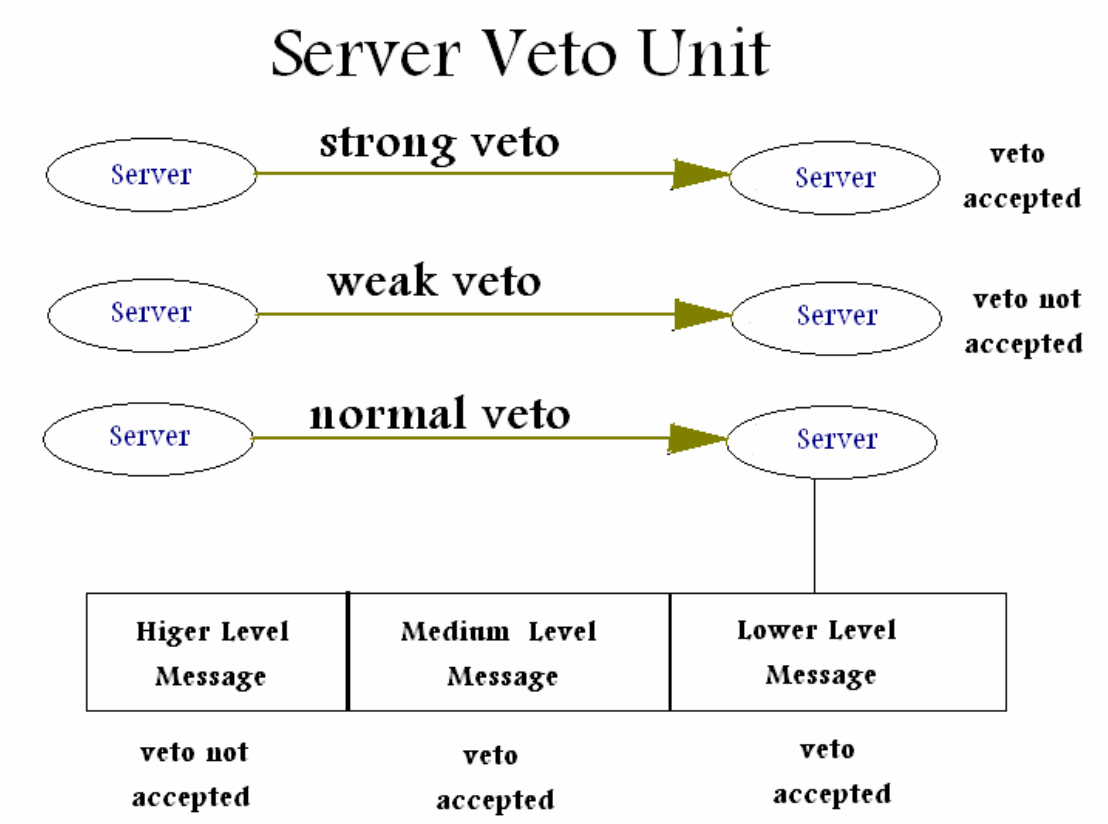
شكل٩ : مكونات وحدة الكود داخل الخادم

توجد هذه الوحدة لكى تتولى عملية هيكلة الكود داخل البرنامج على هيئة مقاوامات (البديل للوظائف او الدوال Functions) توجد داخل فروع Branches تتواجد فى دائرة كهربية والهدف من ذلك التحكم التلقائى فى ترتيب تنفيذ العمليات من خلال المفهوم الاساسى بان التيار (وليكون المعالج) سوف يمر (ينفذ) المقاومات (الدوال) المتصلة على التوالى ثم المتصلة على التوازى ويراعى ان فى كل فرع مفتاح يتحكم فى عملية تنفيذ مقاومات (دوال) الفرع .
اى ان وحدة البيانات تضع فى الاعتبار نظام ادارة الاحداث Event Driven System.



شكل١٠: دائرة الاحداث فى البداية داخل وحدة الكود داخل الخادم

والان سوف ننتقل الى وحدة التصويت Veto Unit :



شكل١١: ميكانيكية عمل وحدة التصويت داخل الخادم

يوضح الشكل ١١ دعم الخادم لتطبيقات Client – Server من خلال عمل تحكم على Response of server اى استجابة الخادم للطلبات التى تصل اليه ولا تقتصر الوحدة على ذلك بل تحتوى على ما يسمى ب Connections للربط بين الخوادم و Channels لاحتواء البيانات المتداولة بين الخوادم.

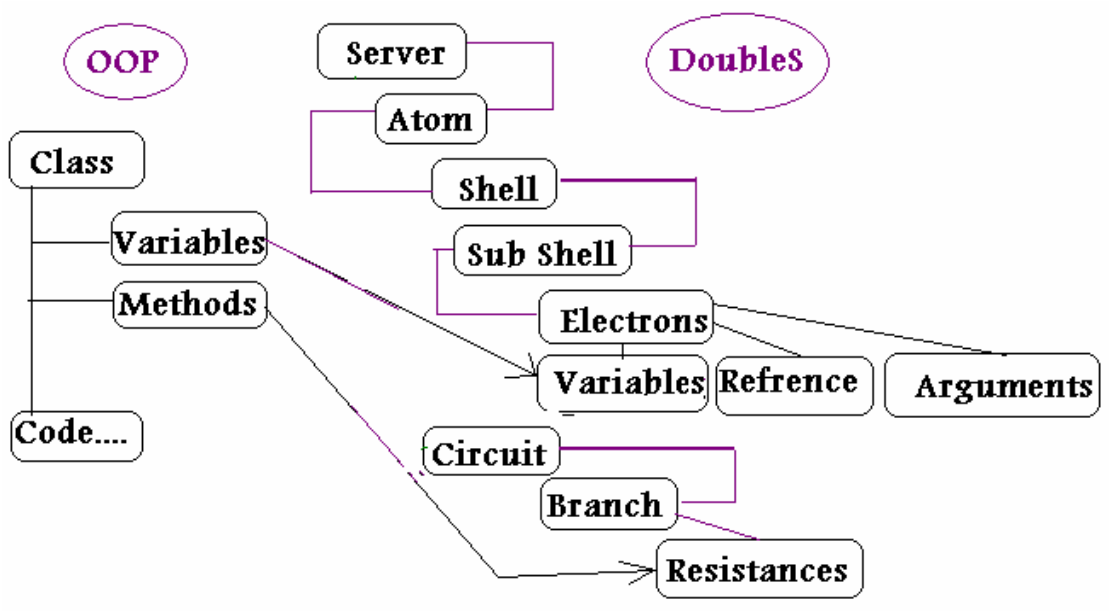
والان نقدم توضيح كيف ان نمط برمجة الخادم الممتاز يقدم بديل امثل لكل من البرمجة الهيكلية وبرمجة الكائنات

Structure programming	DoubleS
Software Program	Network
Procedure	Server
Function	Resistance
Calling procedure	Connection to server
Calling function	Sending veto to server

جدول ١ : المقابل فى الخادم الممتاز للبرمجة الهيكلية

OOP	DoubleS
Instantiation(new object)	Connection (new client)
Polymorphism	Polymorphism (the same)
Delegation	Sending veto
Encapsulation	More encapsulation
Inheritance	Server type : Connection

جدول ٢ : المقابل فى الخادم الممتاز للرمجة الكائنات



شكل ١٢ : جعل الخادم يمثل محتويات فصيلة

شكل ١٢ يوضح ميزة هامة من مميزات الخادم وهى Integeration حيث ان الخادم الواحد يمكن ان يقوم بعمل العديد من الفصائل Classes وان يقوم بتمثيل نظام OOP كامل حتى ولو كان يحتوى على وراثة Inheritance

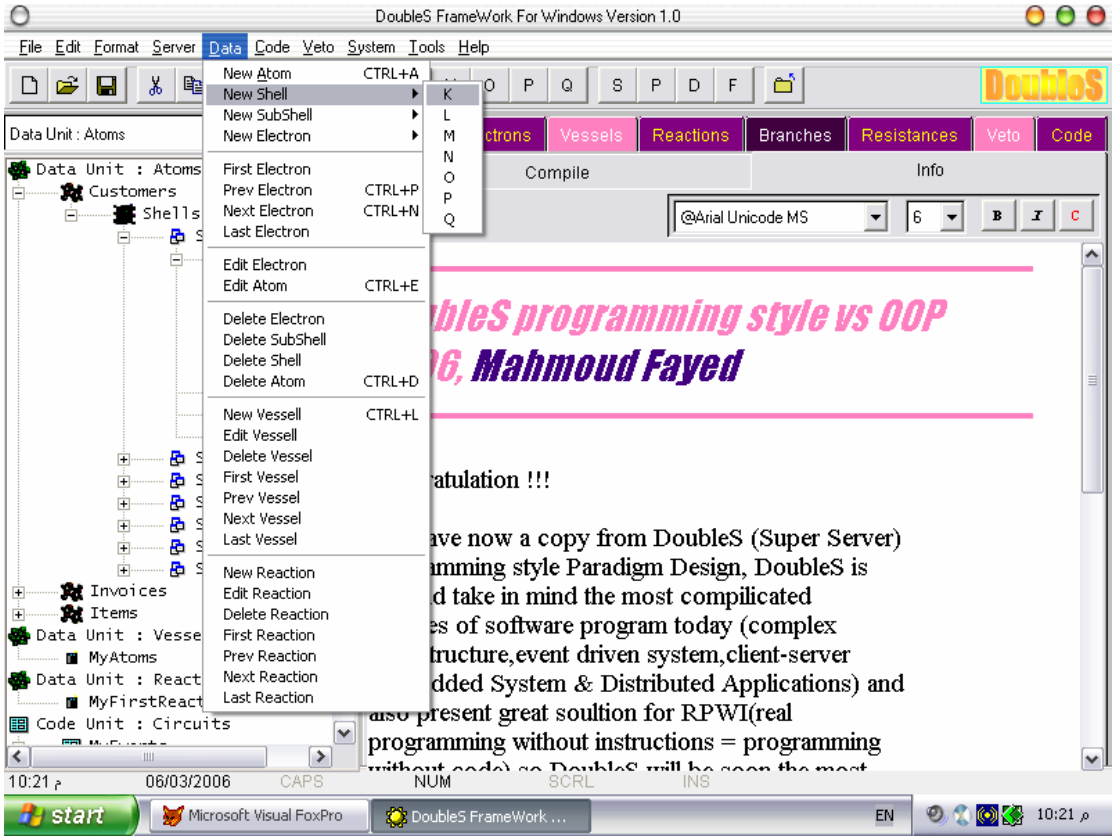
جمل المقاومات Resistance Statements :

حيث ان المقاومة فى الخادم هى البديل للدالة Function وحيث ان هيكلة الخادم تبدو معقدة فلابد وان هناك جمل للاستفادة من هذا التعقيد والحصول على النتائج المرجوة

- add network <network name>
- ignore network <network name>
- <logical var. name> = is network <network name>
- contain network <network name>
- login <server name>
- send data <var. name>
- logout <server name>
- add server <server name>
- ignore server <server name>
- <logical var. name> = is server <network name>
- contain server <server name>
- <var. name > = new client <server name>
- <client var. name>.getcontrol
- <client var. name>.addtosystem
- <client var. name>.resistancename()
- select atom < atom name >,<atom2 name>,...
- select shell < sheel name >,<shell2 name>,...
- select subshell < subshell name >,<subshell2>,...
- select address <atom>-<shell>-<subshell>
- select reaction <reaction name>,<reaction2 >,....
- select vessel <vessel name>,<vessel2 name>,.....
- goto first electron
- goto last electron
- goto next electron
- goto prev electron
- put electron in <variable name>
- delete electron
- replace electron with <variable name>
- create electrons list <list name>
- delete electrons list <list name>

- add electron to list
- delete electron from list
- set domain <list name>,<list2 name>,.....
- close domain
- stop circuit <circuit name>
- continue circuit <circuit name>
- do circuit <circuit name>
- delete circuit <circuit name>
- stop branch <circuit name>-<branch name>
- continue branch <circuit name>-<branch name>
- do branch <branch name>
- delete branch <circuit name>-<branch name>
- do resistance <resistance name>
- delete resistance <resistance name>
- set network <network server name>
- set connection <connection server name>
- new channel <channel name> , <resistance name>
- select channel <channel name>
- close channel <channel name>
- delete channel <channel name>
- send veto <veto name> to <server name>

محيط التطوير دبل اسى :
يستخدم فى تصميم الخادم وبعد الانتهاء من التصميم يقوم باستخراج ملف نصى يحتوى على الجمل الازمة للتعبير عن التصميم



شكل ١٣ : محيط تطوير الخادم الممتاز – برمجة السيد : محمود فايد

فى نهاية الامر كانت تلك مجرد مقدمة وارجو ان تكون مفيدة ومدخل جيد لهذا الموضوع الشيق وللحصول على كافة المعلومات فهى متوفرة من خلال الموقع www.sourceforge.net/projects/doublesvsoop ولكن باللغة الانجليزية

مع اجمل تحياتى .

محمود سمير فايد
كلية الهندسة الالكترونية – جامعة المنوفية
C++, Clipper & Visual FoxPro Developer
Embedded GUI Management system Designer
The main author of DoubleS programming style