Z2 Computer Solutions

Wayne L. Atchison
1609 Lund Lane
Polson, MT 59860

Wayne@Z2cs.com www.Z2cs.com

January 29, 2016
Edited April 18, 2018

The Snippet Engine Technology

Executive Summary

The Snippet Engine is an innovative software development technology, which dramatically simplifies the task of synchronizing networked Servers, Databases, and Applications across the Internet. The Snippet Engine is a truly unique technology. This new technology innovation is designed to take full advantage of modern networked computers, each having multiple CPU cores. The business opportunities for using the Snippet Engine Technology, for both Database Servers and Applications, are infinite.

Background:

There are foundational reasons why the number of failed multi-million dollar Object Oriented Design (OOD) projects are too numerous to even count. Not only are large projects always way over budget, most simply fail to meet the project's original list of requirements. To overcome the shortfalls of the OOD methodology, the Snippet Engine was created. The Snippet Engine enhances the C++ class object and the OOD methodology. **The result is the creation of a new "Engine" to process the new enhanced Objects**.

The Snippet Engine Methodology expands the typical OOD object **into a new kind of Object**. The Snippet Engine Object is called a "**Node**" which:

1. Automatically takes full advantage of **massive parallel processing** through the multiple CPU-Cores, as each Object executes as its own Core Thread.

2. Enables any Object to **talk directly to any other Object**, on any computer in the world.

3. Allows other Objects, **whether executing locally or remotely**, to be included as designed subsets of itself. The designed purpose of an Object can include any other Objects running anywhere in the world.

Thus, the most significant limitations of the traditional OOD C++ class object are overcome. The Snippet Engine allows Object Oriented Programming Objects to be Network-integrated Objects too. Network Objects understand that other Objects reside on remote computers, and are split across space and time.

The foundational concept behind the Snippet Engine technology, is to use these new kind of Objects (Nodes) **to aggregate almost anything**. Each Object can now talk to, and aggregate (build one thing on top of another) with any other Objects running anywhere in the world.

The Snippet Engine is itself a normal application, and is a normal EXE file, compiled with these enhanced Node-Objects. Thus, the Snippet Engine is running on a computer, **just as does any normal (double clicked icon) application**.

Visualizing The Snippet Engine:

A "**snippet**" is a small piece of a larger thing. The Snippet Engine lets the developers divide any size of Application into numerous independently scheduled and self-aware smaller pieces, as Objects. The whole Application is accomplished as each of its many smaller pieces perform their individual tasks, and talk to each other.

To visualize what the Snippet Engine does, think of an ant hill. Hundreds of little ants all working independently, and collectively, to accomplish the greater task. Each of the Snippet Engine's Objects are like a single ant. The Snippet Engine enables the developers to create Objects, each like a small ant, each being independent and self-aware, and each doing their little part of the whole. Each Snippet Engine Object is doing whatever it is supposed to do, to collectively accomplish the greater task. The Snippet Engine enables hundreds/thousands of independent "software ants" ("Cyber Ants") to work both independently and collectively, to accomplish the greater goal.

The significant exception to this "ant hill" analogy, is that unlike an ant, each Snippet Engine Object, "a software ant", **can instantaneously and directly communicate to any other "software ant", anywhere in the world**.

Each "software ant" can share data, accumulate intelligence, command others, manage its own sub-tasks, work independently, think ahead, and interact collectively. Each Snippet Engine "software ant" can do its own task, accept commands to do more tasks, command other "software ants" to do their tasks, and think ahead in preparation of the next tasks.

> **No other software technology enables the developers to create high level Object Oriented Constructs, where each Object is capable of worldwide, cooperative, self-aware interaction, and infinite aggregation of intelligence**.

A Browser's AJAX interface can be used to talk directly to any Snippet Engine Node in the world. A Server's PHP fsocket-interface can be used to talk directly to any Snippet Engine Node in the world. **This means that any Internet website can directly tap into any Snippet Engine application, in real time, to fetch or accomplish anything desired**.

Imagine thousands of independent "software ants" executing all over the world, interfacing with your website. There is no limit to what can be accomplished!

To The Project Designers:

The Snippet Engine enables the designers to elevate their typical object oriented constructs, into much higher level constructs, where each elevated construct is capable of independent, autonomous, cooperative, interactive, self aware, Core-Thread level execution. Each Snippet Engine construct "Knows How To Do Something", which is their small piece of the whole project. Applications and Database Servers are designed and developed using these Snippet Engine constructs, as Objects. Any Object can talk directly to any other Object, running anywhere in the world. **These innovations allow command and control exchanges and data update synchronizations to be accomplished in real-time, around the world**.

Snippet Engine Applications and Servers are accomplished by the massive parallel processing of these "self-aware" interacting Snippet Engine Objects, executed by the multiple cores on multiple networked computers. Because any Object can directly talk to any other Object, **there are no limits to the scope and total aggregation of the project's design**.

The Snippet Engine's footprint is small enough, and executes fast enough, to be run on any kind of computer. Special Server-Computers are not required, as Snippet Engine Applications/Servers can be run on home computers and laptops.

Multi-Core 'Cache Coherency' and inter-Thread 'Atomic' synchronization:

The following capability is vital!
The Snippet Engine expects to be executing multiple Threads on multiple cores requiring inter-Thread "logic" synchronization.

It is a well-hidden fact that modern Pentium computers hardware absolutely fail during same-clock-cycle writes by three or more Threads running on multiple cores. Such failures are documented, and easily demonstrated in tests.

The whole reason for multiple Threads to be using a common 'address', is so that their independent "logic" can be synchronized with the other Threads on other cores. The Snippet Engine uses **its own 'lock-less software' to guarantee** that inter-Thread 'Atomic Operations' are fully 'Cache Coherent', even with concurrent writes on multiple cores, regardless of the 'Cache Coherency Protocol' of the hardware.

Further, blocked Threads wait in a First-In-First-Out Queue, guaranteeing the "logical order" of the application's inter-Thread "logic" synchronization.

Thus, the Snippet Engine works on all computers, from low-cost Laptops to expensive Servers.

One of the primary advantages of using Snippet Engine Objects is that they **dramatically reduce the total number of software interfaces** within the delivered application. This reduction in the number of software interfaces saves considerable time and cost in writing test-cases, in testing path permutations, in integration, in documentation, and in future maintenance. Another advantage is that Snippet Engine Objects can be reused in future projects.

Another primary advantage is that there is no separate “Integration Phase” in a Snippet Engine Project. Snippet Engine Projects are developed “**from the bottom-up**”. As each Node is “Unit Tested”, it must use those Nodes below it to do their preliminary work, so then those Nodes are being fully integrated. Thus, all integration occurs during “Unit Testing”. This means that large projects will experience at least **a 40% reduction in development cost**. This is because a Snippet Engine Project does not require the integration-level test cases to be written, nor the formal integration testing, nor the formal integration bug-tracking.

To The Project Programmers:

Think of the Snippet Engine as normal C++ code, that knows how to manage Snippet Engine Objects. **It is a normally compiled EXE**, that is executed as the application, with the SnEn Nodes compiled within. The Snippet Engine tracks the interrelationships, interactions, and scheduling of the Snippet Engine Objects. All Objects are autonomously executed as Threads at the core-level of the Operating System. Executing as Threads, the more CPU-cores a computer has, the more of the asynchronous “software ants” will be executed concurrently, in mass parallel processing. **Each Object will perform its duties as an autonomous self-aware “software ant”, talking to any other “software ant” anywhere in the world**.

Each “Software Ant” (an encapsulated Snippet Engine Object):

1. Has its own **unique Thread-level C++ code** (your code executes within a system lock, it cannot be interrupted until done, and your data is always isolated for concurrency),

2. Will **direct its own scheduling** (your code is always “alive”, self aware, and can think ahead in anticipation),

3. Has **its own command queue** (your code can be told to do things by others),

4. Can use any kind of communications interface to **talk to anything else** (your code can command, store, and fetch from other code running anywhere in the world, even using differing encryptions),

5. Has **its own data store** (your code controls its own piece of the Database), and

6. Can **be created, saved, and deleted as required** (your code can create, command, and delete other Objects, in order to do things for you, anywhere in the world).

These capabilities mean that each “software ant” executes as an independent Thread, but behaves as if it were a high-level application-subtask. The Snippet Engine will manage thousands of these “software ants” as independent self-aware workers. Thus, each “software ant” is a self-aware, autonomous, command-driven, data storing worker for the greater goal.

Because Snippet Engine software-constructs are independently created, opened, closed, deleted, and executed at the lowest level of a CPU-core, each Snippet Engine Object can be designed to do any desired task, large or small, just as if they were a separate sub-program of the Application.

Because each “software ant” is scheduled and executed at the Thread-level, each enjoys a guaranteed system-lock on whatever data it is managing. This system-lock eliminates huge amounts of software overhead, typically found in most other Database Networks.

Because each “software ant” has its own communications queue, it can instantaneously process commands and communicate its Database changes and progress to any other “software ants” running anywhere in the world. This data exchange is not only instantaneous; it is direct, fast, and uncomplicated. Further, **any type of encryption protocol may be used**, even using different protocols per type of Snippet Engine Object.

Saved Objects retain their data stores and status on disk, which then combine to form the Application’s entire Database. In this way **a Snippet Engine Application is a Database Server too**. That is, some of the Objects are designed to manage their own assigned Table of data. For example, one Object knows how to manage the Table for “myCustomers”, and another Object knows how to manage the Table for “myOrders”.

The project's design can have the Object for "myCustomers" running on a Server in Chicago, and the Object for "myOrders" running on a Server in Seattle. The two Snippet Engine Objects will talk directly to each other, in real time, as needed.

SUMMARY

The Snippet Engine Technology and its enhanced Objects allow very large networked projects to be developed successfully. Using the Snippet Engine means that any number of worldwide Databases and Applications can be kept fully coordinated, synchronized, and aggregated in real time. Snippet Engine Networks will cost far less, be much simpler to manage and maintain, and will be astronomically faster than typical Server Networks. Snippet Engine Networks can provide 24/7 redundancy, fail safe reliability, infinite scalability, and infinite aggregation of intelligence.

Imagine thousands of your own independent "software ants" executing all over the Internet. There is no limit to what you can accomplish! There is no limit to the revenue stream created!