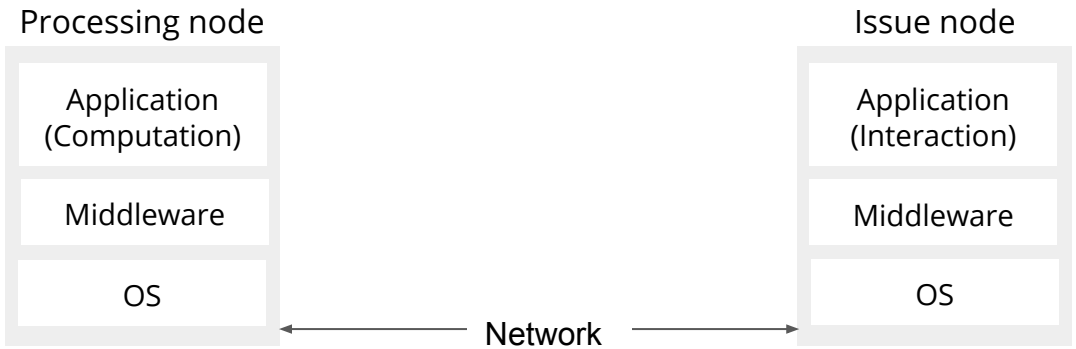
Computational Resource Pooling

Bachelor thesis

Idea

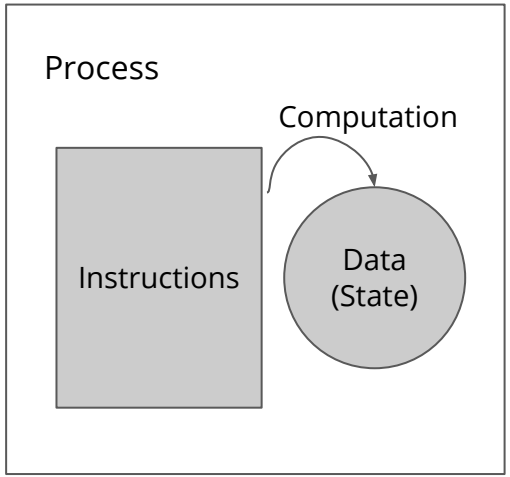
Create a virtual environment where a process issued by a node could use any compute resource in a LAN or Network.

Based on the below diagram, the issue node is where the user runs the process. The middleware we build sophisticates the computation of that process on another[Processing node], while the interaction[I/O] of the process remains in the Issue node. In this way, the entire architecture could remain transparent to the user.



Theory

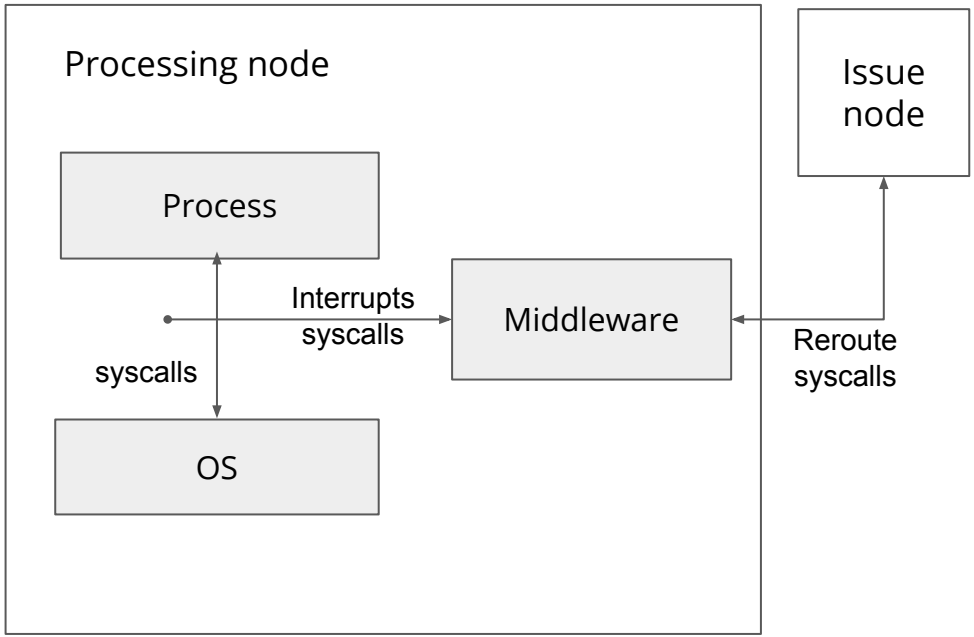
```
void main(){
    int x;
    x=0;
    x=x+5;
    printf("value of x is %d",x);
}
```



In the represented C code snippet, X is the data. Its start as 0 [Initial state]. Then X+=5 occurs. So, change in the state of X. This change of data is called computation. On the whole, a process starts with the initial state and executes series of instructions [computation]. This computation is the same irrespective of the system it executes.

Consideration of system matter, when the process executes system calls. Like “printf” in the snippet. Rerouting of these system calls to the issue node makes the middle layer transparent.

Middleware implementation



Linux provides the ptrace system calls, which the parent process could use to trace the child process.

Middleware could execute the process as its child and trace it. If a ***selective system calls**** occurs from the child, MW reroutes to the host node.

Selective system calls: Not all system calls are required to execute in the issue node. Some could/should execute at the processing node. The selective system calls include read, write, open, etc.

Middleware conclusion

Once user hits execute a process, MW must ship the executable to the processing node and execute it as its child.

Any selected system calls made by child process must be rerouted to issue system and the middleware must execute it in the host node.

The results must be sent back to the processing node and this communication and execution must be made transparent to the process and user.

So, on the whole any node that has this middleware their computational resource are pooled virtually.