CLAIMS

What is claimed is:

Claim 1. A system comprising:

at least one memory having computer-readable instructions stored thereon; and

at least one processor, the at least one processor being configured, in response to the computer-readable instructions, to:

receive, via a monitor subsystem, a request from a client subsystem, wherein the request is a request for performing secondary analysis or tertiary analysis of sequencing data, wherein the request comprises a version associated with the client subsystem;

determine a first version associated with a server subsystem to be implemented for servicing the request;

determine that the version associated with the client subsystem is incompatible with the first version associated with the server subsystem currently implemented for servicing the request;

in response to the determination that the version associated with the client subsystem is incompatible with the first version associated with the server subsystem, prevent the server subsystem from being accessed by the client subsystem;

identify that the server subsystem is idle after processing each request from client subsystems associated with a compatible version;

cause the server subsystem to perform a change from the first version to a second version;

determine that the second version associated with the server subsystem is compatible with the version associated with the client subsystem; and

allow the client subsystem to access the server subsystem for servicing the request.

Claim 2. The system of claim 1, wherein the request from the client subsystem is a first request, and wherein the client subsystem is a first client subsystem, the at least one processor being further configured, in response to the computer-readable instructions, to:

receive, via the monitor subsystem, a second request from a second client subsystem, wherein the second request comprises a version associated with the second client subsystem;

while the server subsystem is operating in the first version, determine that the version associated with the second client subsystem is compatible with the first version associated with the server subsystem; and

allow the server subsystem to be accessed by the second client subsystem for the second request.

Claim 3. The system of claim 1, wherein each version of the server subsystem comprises different features for bioinformatics components associated with processing different requests for the secondary analysis or the tertiary analysis.

Claim 4. The system of claim 1, wherein different bioinformatics components are utilized by loading the different bioinformatics components into one or more field-programmable gate arrays (FPGAs).

Claim 5. The system of claim 1, wherein the change from the first version to the second version comprises a tear down and a buildup of at least one software layer in a vertical solution stack.

Claim 6. The system of claim 5, wherein the first version and the second version support different bioinformatics components to be loaded to one or more FPGAs for supporting a same or different sequencing task.

Claim 7. The system of claim 6, wherein the sequencing task comprises a task performed for the secondary analysis or the tertiary analysis of the sequencing data.

Claim 8. The system of claim 1, wherein the secondary analysis or the tertiary analysis comprises alignment, sorting, or variant calling based on the sequencing data.

Claim 9. The system of claim 1, wherein the monitor subsystem is a daemon process.

Claim 10. The system of claim 1, wherein the server subsystem is a daemon process, and wherein the server subsystem is configured to manage hardware on the system for servicing a sequencing application operating on one or more client subsystems.

Claim 11. The system of claim 10, wherein the daemon process is one of a plurality of predefined daemon processes in the server subsystem.

Claim 12. The system of claim 1, wherein the at least one processor being configured to prevent the server subsystem from being accessed by the client subsystem for the request further comprises the at least one processor being configured to:

prevent an access key from being transmitted to the client subsystem or a connection from being established between the client subsystem and the server subsystem.

Claim 13. The system of claim 1, wherein the at least one processor being configured to allow the server subsystem to be accessed by the client subsystem further comprises the at least one processor being configured to:

provide, via the monitor subsystem, an access key to the client subsystem; and

notify, via the monitor subsystem, the server subsystem that the access key has been provided to the client subsystem.

Claim 14. The system of claim 13, wherein the at least one processor being configured to notify the server subsystem further comprises the at least one processor being configured to provide the access key to the server subsystem.

Claim 15. The system of claim 13, wherein the processor is further configured, via the computer-readable instructions, to:

receive a message via the client subsystem indicating that the client subsystem has been blocked; and

wait to receive an indication via the client subsystem from the monitor subsystem indicating that the request can be processed.

Claim 16. The system of claim 13, wherein the processor is further configured, via the computer-readable instructions, to:

receive the access key via the client subsystem; and

transmit a message to the server subsystem that includes the access key for establishing a connection with the server subsystem.

Claim 17. The system of claim 1, wherein the processor is further configured, via the computer-readable instructions, to send an acknowledgement to the client subsystem in response to the request.

Claim 18. The system of claim 1, wherein the processor is further configured, via the computer-readable instructions, to receive the request from the client subsystem via a socket, and grant access to the server subsystem through the socket via an access key.

Claim 19. The system of claim 1, wherein the processor is further configured, via the computer-readable instructions, to:

monitor, by the monitor subsystem, signals from at least one child process;

detect a terminal signal via the at least one child process; and

automatically restart the at least one child process.