Introduction

ShockWaves is a framework for realtime data processing in a decentralized environment.

What doing ShockWaves?

The ShockWave processes any data incredibly quickly using numerous computers and thousands of computational modules created by community members.

Why are computations necessary?

Computers are essentially involved in processing data.

When you use a smartphone or any internet service, you interact with simple visual interfaces. However, to make this work, a vast infrastructure of data centers exists where your data is stored, and your requests are processed.

Here are a couple more examples:

- **To save a video stream from surveillance cameras to a hard drive:** The computer breaks down the video stream into a sequence of frames and applies various computational algorithms to compress the video stream.

- **To 3D print an object:** The computer first builds a 3D model based on initial vertices, then slices it into layers. For each layer, it creates a trajectory.

Purpose of ShockWaves

The future of engineering development lies in the synthesis of design through AI or high-performance computing.

Computations span diverse tasks. Machine learning includes PCB tracing, CNC handles 3D printing, and software deals with program configurations. Generative design optimizes 3D printing. Physical process simulation models electromagnetics, materials, and mechanics. Electronics simulation covers circuit operations. Data analysis processes sensor and log

data for insights. These tasks collectively define the broad spectrum of computational applications.

Mission

The mission of ShockWaves is to make the use of open-source computational algorithms simple and provide community members with access to computational power that is typically available only to large corporations.

Difference

Architect

Built upon computational microkernels, high-performance queues, and task routing using Directed Acyclic Graphs (DAGs). At its core, ShockWaves utilizes a proprietary implementation of DAGs for data stream processing.

Clustering

In developing ShockWaves, our focus has been on scalability and compatibility. The ShockWaves platform always operates in a Kubernetes cluster, even if it's running on a single computer.

- Each ShockWaves node can be used for computations.
- Computations are managed through message queues, allowing for automatic load distribution across cluster nodes.

Differences from Kubernetes

Kubernetes is a platform for container orchestration, providing tools for deploying containers in a cluster and configuring their network interactions through virtual networking. Typically, Kubernetes employs a microservices architecture, where containers interact directly by establishing connections via HTTP or TCP protocols.

ShockWaves, on the other hand, is a platform designed to process real-time data within Kubernetes clusters. It deploys its infrastructure consisting of multiple containers within

clusters. Interaction between platform components occurs through high-performance message queues, allowing for easy load balancing and distribution across numerous nodes without the need for establishing direct connections.

Advantages

Performance

The ShockWaves framework is capable of processing millions of tasks per second and handling data streams at hundreds of gigabits per second in real-time.

Designed to operate without bottlenecks, ShockWaves encompasses thousands of data processing modules. These modules are capable of functioning on a single computer or across clusters of thousands of servers.

Scalability

ShockWaves is scalable and can be deployed on a range of devices, from a basic computer like a Raspberry Pi to a cluster of hundreds of servers, or on an HPC (High-Performance Computing) cluster in the cloud.

Efficiency and Reliability

Efficient use of computing resources and reliability are built into the architecture of ShockWaves. Maximum efficiency in the use of computing power and disk space is achieved by utilizing all nodes for distributed computing and data storage.

Compatibility

ShockWaves is a meta-framework, which means it relies on the higher-level Kubernetes platform. Kubernetes, in turn, has numerous implementations for various tasks. This approach provides high compatibility and flexibility in using ShockWaves.

ShockWaves can be deployed on:

- Any Kubernetes cluster (k3s, k8s, etc.).

- Any cloud-based Kubernetes service (AWS, GCP, Azure, etc.).
- Any cloud-based HPC cluster managed by Kubernetes (k3s, k8s, etc.).
- Supports processor architectures including AMD64 and ARM64.

Advantages of ShockWaves for Digital Manufacturing

- High fault tolerance due to operating in cluster mode.
- Automatic configuration and firmware updates for thousands of hardware devices.
- Automatic processing and scheduling of tasks, such as 3D printing tasks.
- Management of multiple devices as one entity, whether it's a single 3D printer or a hundred.
- Centralized log storage and notification of staff about failures.
- The ability to continue working in the event of individual node failures.

Types of Accelerators

MicroKernels can interact with various types of accelerators, provided the necessary drivers are available. Here are some examples:

- **CPU** - Standard Processor
- **GPU** - Graphics Processing Unit
- **FPGA** - Field-Programmable Gate Array
- **NPU** - Neural Processing Unit
- **DSP** - Digital Signal Processor
- **ASIC** - Application-Specific Integrated Circuit

Architect

MicroKernel Applications

A ShockWaves application is composed of multiple separate applications (micro-kernels) that, during assembly, are packaged into Linux containers. The deployment of these cellular applications can occur in a single cluster or across a multi-cluster decentralized environment.

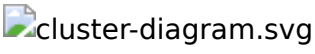
Cellular applications are either built from existing open-source programs or developed from scratch. These programs are segmented, modified for data exchange via queues, and then encapsulated into containers (micro-kernels). From these containers, deployment configurations for cellular applications in Kubernetes clusters are created. This modular approach allows for flexibility and scalability in deployment.

Micro-Kernels

A micro-kernel is a part of a program that contains one or several resource-intensive data processing functions. If the kernel includes multiple functions, they are logically unified by a single concept. This segment of the program is packaged into a container and can be deployed in a Kubernetes cluster.

The functions within these micro-kernels are stateless between calls. Once a computational task is completed, all resources are freed. When a function is invoked, all the resources of the container are dedicated solely to that task. This design ensures efficient use of resources and optimal performance for each task.

Framework Components



Microservices - Modules designed to handle user requests on the backend. They operate as containers within a lightweight Kubernetes cluster. Users on the Solenopsys platform can create and publish their own microservices, which are then immediately available for use by other network participants.

Data Storage Services - Converged uses data storage services instead of traditional databases with regular drivers. These services act as wrappers for database drivers,

enabling asynchronous interaction with the database via a message queue.

Cache - A distributed cache that operates at the cluster level, storing the states of data processing workflows.

FlowControl - A micro-program that manages data and command streams.

StreamGates - Programs designed to receive data streams at the cluster's entry point, transform them, and distribute the processed streams to HostRouters. eBPF technology can be employed for high-performance packet conversion within StreamGates.

ClusterRouter - A program responsible for the deployment and scaling of cellular applications across clusters.

HostRouter - Manages the computational process of tasks. They enable interaction between multiple micro frontends and microservices, directing message flows via WebSockets, crucial for real-time event display.

Microkernels - Specialized containers for data processing, akin to AWS lambda functions. ZeroMQ technology is used for data exchange between microkernels. Within a single node (computer), data exchange can reach speeds of three million messages per second with a transfer delay of approximately 5 microseconds.

SHOCK Protocol - Frontend and backend data exchange is conducted through message streams. This involves using WebSockets for the frontend and ZeroMQ within the cluster.

Advantages of MicroKernel Architecture

- **Easy Scalability** - Microkernels can be deployed flexibly, adapting to changing load demands.
- **Resource Limitation** - Each microkernel can be restricted in terms of resource usage.
- **Resource Usage Monitoring** - Allows tracking of computational resources expended during task processing.
- **Language Independence** - Different parts of the program can be written in various programming languages.

- **Code Reusability** - Microkernels can be reused across different applications, enhancing efficiency.

Purpose

Example

Imagine you have a 3D printing business with several dozen 3D printers, each controlled by its embedded computer (like a Raspberry Pi). When using the ShockWaves platform, they automatically form a cluster. The disk spaces of these printers are also clustered, ensuring reliability, greater computational power, and large memory capacity for data storage.

Thus, slicing before printing can be done almost instantly, as it occurs across dozens of computers instead of just one.

Types of Computations Examples

- **Machine Learning**
  - PCB (Printed Circuit Board) tracing
  - Component placement
  - Electronic chip synthesis
  - Optimization
  - Machine vision

- **CNC Program Generation**
  - Slicing for 3D printing
  - CNC machine programs

- **Software Compilation and Computer Logic Synthesis**
  - Program compilation

    - FPGA configurations
    - Electronic chip configurations

- **Generative Design**
    - Generative design for 3D printing
    - Finite element analysis
    - Parametric optimization
    - Design optimization for 3D printing

- **Physical Process Simulation**
    - Electromagnetic modeling
    - Material flow modeling
    - Heat transfer modeling
    - Computational chemistry
    - Mechanical process modeling

- **Electronics Simulation**
    - Electronic circuit operation modeling
    - High-frequency data transmission line matching on PCBs
    - PCB tracing
    - Computer logic verification

- **Data Analysis and Processing**
    - Sensor data processing
    - Log processing

- Streaming data transformation