АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ТРЁХМЕРНОЙ ВИЗУАЛИЗАЦИИ МОНИТОРИНГА ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА

Д. А. Тишкин

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

В настоящее время существует огромное количество различных вычислительных комплексов, состоящих из множества различных компонент. Необходимо следить за всеми компонентами вычислительного комплекса и в случае внештатной ситуации оперативно принимать меры по их устранению. Для слежения за различными компонентами вычислительного комплекса существуют различные системы мониторинга. Каждая система следит за своими параметрами вычислительного комплекса и по запросу пользователя, выводит эту информацию на экран. Чаще всего используемые нами системы мониторинга имеют интерфейс командной строки (рис. 1)

У систем мониторинга с интерфейсом командной строки есть несколько недостатков:

- Из вывода тяжело представить внешний вид вычислительного комплекса;
- В случае сбоя оборудования, сложно узнать его физическое расположение;
- Пользователю необходимо вручную вводить команду в командную строку, чтобы увидеть актуальные данные системы мониторинга;
- Различные системы мониторинга, могут представлять информацию в разных видах;
- Сложно дать качественную оценку состоянию всего комплекса.

Для решения этих проблем была создана система трёхмерной визуализации мониторинга вычислительного комплекса.

Интерфейс системы визуализации мониторинга вычислительного комплекса показан на рис. 2. Система визуализации интерактивна, сцену можно вращать, изменять масштаб и взаимодействовать с объектами на сцене. С правой стороны находится информационная панель, в верхней части которой отображается название выбранного элемента, ниже располагается список его дочерних элементов, а в самом низу панель с управляющими кнопками: автоматическое вращение камеры, поиск элементов, отображение легенды. К основным возможностям системы визуализации можно отнести следующие пункты:

- Функция поиска, позволяет определить физическое расположение любого элемента вычислительного комплекса;

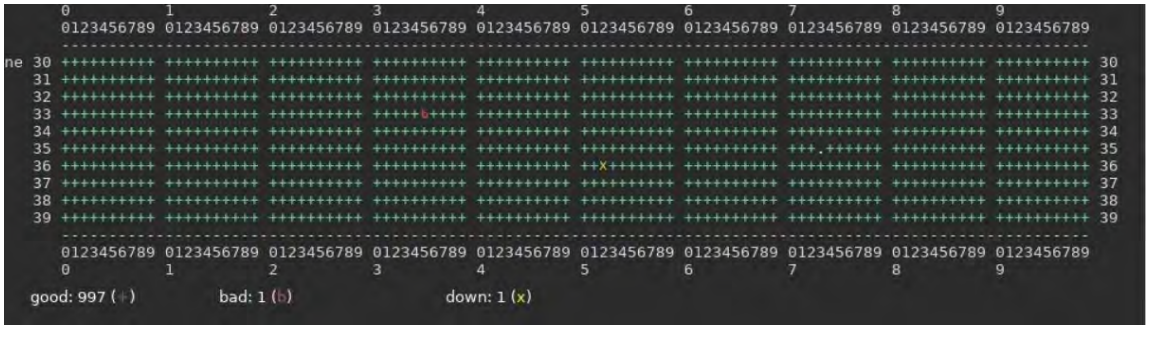

Рис. 1. Вывод состояния компонент в командной строке

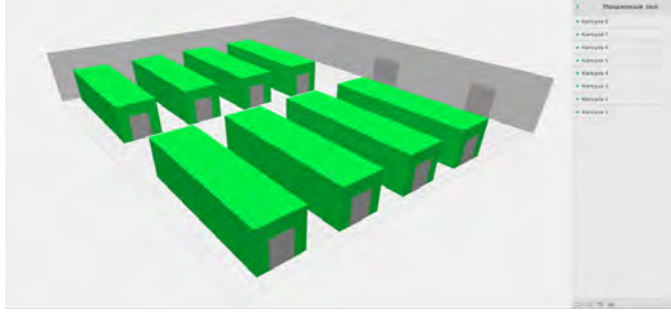

Рис. 2. Внешний вид интерфейса системы визуализации

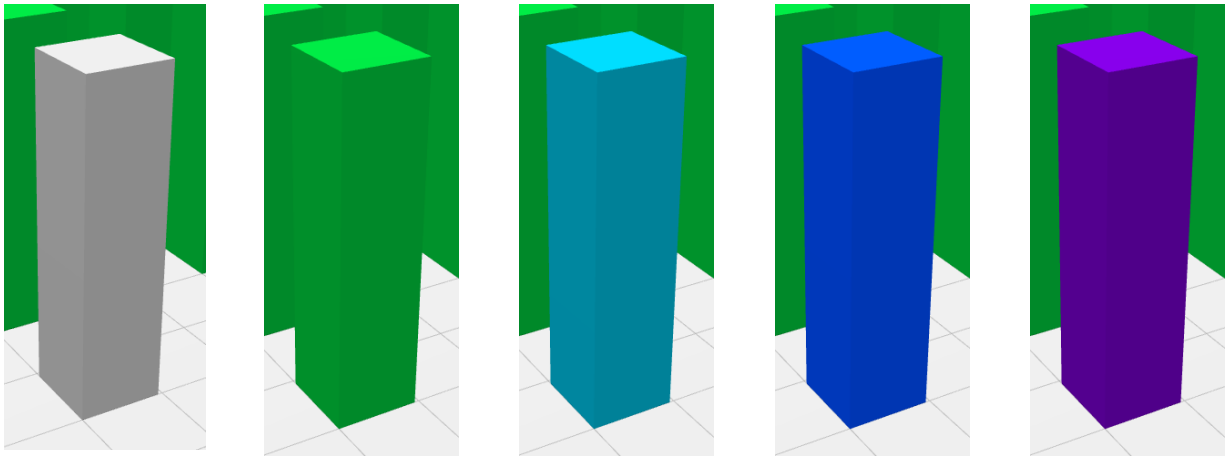

Рис. 3. Состояния элементов вычислительного кластера (на примере стойки)

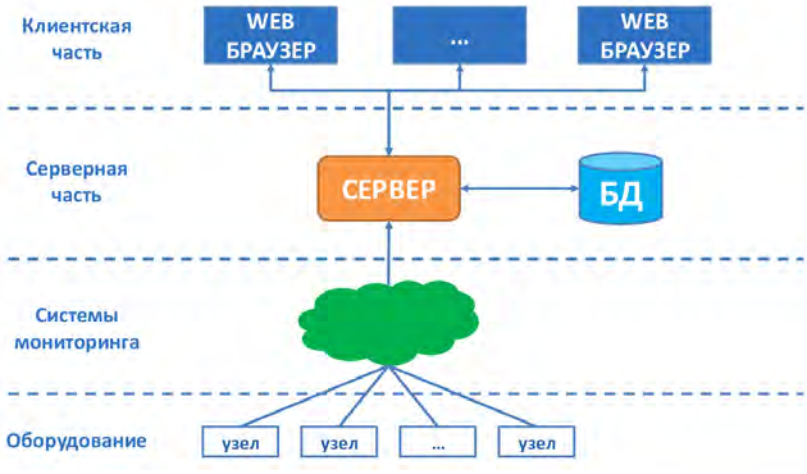

Рис. 4. Общая схема архитектуры системы

• Обновление данных и отображение изменений состояния элементов происходит автоматически без участия пользователя. Элемент может принимать одно из 5-ти состояний (unknown, ok, noncritical, critical, nonrecoverable) каждое состояние отображается определённым цветом (рис. 3);

• Система имеет возможность отображать данные от различных источников в едином формате;

• Благодаря тому, что клиентская часть выполнена в виде веб-страницы, появляется возможность пользоваться приложением в любой операционной системе с помощью любого современного веб-браузера.

Система трёхмерной визуализации мониторинга вычислительного комплекса выполнена в виде веб-приложения, схема которого изображена на рис. 4.

Клиенты, в качестве которых выступают веб-браузеры, обмениваются данными с серверным приложением. Эти данные содержат состояние компонент вычислительного комплекса, дочерние элементы выбранного элемента на сцене, значения счётчиков выбранного узла и т.п. Сервер связан с базой данных, в которой хранится информация о компонентах вычислительного комплекса. Значения состояний компонент получаются от различных систем мониторинга, которые собирают их от различного оборудования и хранят их у себя, а по запросу пользователя предоставляют её.

Обмен данными между клиентским и серверным приложением происходит с помощью протокола WebSocket. Это протокол полнодуплексной связи, то есть с его помощью можно принимать и отправлять данные одновременно. Он предназначен для обмена сообщениями между браузером и веб-сервером в режиме реального времени. Благодаря тому, что WebSocket соединение поддерживается постоянно, нет необходимости переустанавливать соединение при каждой передачи данных. В качестве обёртки над протоколом WebSocket в системе визуализации используется библиотека Socket.io. Эта библиотека предоставляет дополнительные возможности над стандартными возможностями протокола WebSocket:

• Поддержка вещания на несколько сокетов;

• Хранение данных, связанных с каждым клиентом;

Рис. 5. Взаимосвязь компонентов вычислительного комплекса

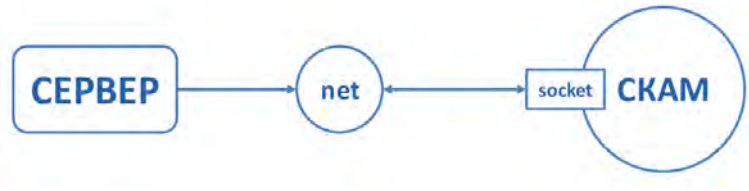

Рис. 6. Схема взаимодействия сервера и СКАМ

• Асинхронный ввод вывод;

• Использование различных протоколов передачи данных, если не доступен протокол WebSocket.

Для трёхмерной визуализации в веб-браузере используется библиотека WebGL. Она предоставляет программный интерфейс для построения интерактивной 3d-графики в веб-браузере с помощью языка JavaScript. Библиотека основана на программном интерфейсе OpenGLES, который позволяет получить аппаратное ускорение для обработки 3D-графики. В настоящее время библиотека включена в поставку всех современных веб-браузеров и доступна без дополнительных плагинов.

При написании приложений с использованием библиотеки WebGL необходимо описывать все шейдерные процедуры, формировать все вершины, ребра и грани каждого объекта вручную, а так же устанавливать и описывать источники света и т. п. Поэтому для упрощения выполнения основных операций с элементами сцены, используется библиотека THREE.js. Эта библиотека предоставляет множество обёрток над низкоуровневыми функциями библиотеки WebGL.

Серверная часть приложения реализована с помощью программной платформы Node.js. Node.js – это платформа, основанная на JavaScript движке v8, которая превращает JavaScript из узкоспециализированного языка в язык общего назначения. В основе этой платформы лежит асинхронное событийно-ориентированное программирование с неблокирующим вводом/выводом. Это позволяет избежать блокировки исполнения программы, вызванной тем, что установка соединения с ресурсом и особенно чтение из него данных требуют времени. С помощью программного интерфейса, реализованного на языке C++, Node.js позволяет использовать устройства ввода-вывода, а так же подключать различные внешние библиотеки, которые написаны на разных языках программирования.

Параметры всех компонент комплекса: их координаты, размеры, названия, отношения между ними и т.п. хранятся в базе данных. Взаимосвязь компонент можно представить в виде дерева (рис. 5)

В качестве системы управления базами данных используется NoSQL база данных MongoDB. NoSQL базы данных позволяют менять структуру отдельных записей в любое время, добавлять или удалять поля. В случае с системой визуализации это важно, т. к. различные компоненты могут иметь различные специфичные параметры. В качестве языка формирования запросов в NoSQL базах данных используется JavaScript, а запросы передаются в виде JSON-строк, а так же они хорошо подходят для хранения иерархических структур данных.

В текущей версии системы реализована возможность получения информации от системы контроля аппаратных метрик (СКАМ). Она собирает значения различных сенсоров оборудования с помощью интерфейса IPMI. Взаимодействие серверной части приложения с системой СКАМ изображено на рис. 6.

Для взаимодействия с интерфейсом системы мониторинга СКАМ, сервер использует встроенный в Node.js модуль «net». С его помощью Node.js создает асинхронное соединение со СКАМ. Обмен данными между СКАМ и сервером происходит с помощью строк в формате JSON.

Результаты

Разрабатываемая система трёхмерной визуализации мониторинга вычислительного комплекса позволяет пользователю увидеть трёхмерную модель вычислительного комплекса, рассмотреть его компо-

ненты с различных сторон и узнать состояние его компонент в реальном времени с помощью любого современного веб-браузера. Встроенный поиск позволяет быстро находить физическое расположение тех или иных компонент вычислительного комплекса.

Литература

1. Официальный сайт библиотеки WebGL [Электронный ресурс]. Режим доступа: http: // www.khronos.org/webgl/;
2. Официальный сайт библиотеки Three.js [Электронный ресурс]. Режим доступа: http://threejs.org;
3. Jos Dirksen “Learning Three.js: The JavaScript 3D Library for WebGL” – Packt Publishing, 2013;
4. Стандарт протокола WebSockethttp: // tools.ietf.org/html/rfc6455[Электронный ресурс]. Режим доступа: http://tools.ietf.org/ html/rfc6455;
5. Сайт библиотеки Socket.IOhttp://socket.io/ [Электронный ресурс]. Режим доступа: http:// socket.io/;
6. Официальный сайт СУБД MongoDB [Электронный ресурс]. Режим доступа: http://mongodb.org/;
7. Официальный сайт Node.jshttp://nodejs.org/ [Электронный ресурс]. Режим доступа: http:// nodejs.org/;
8. Пауэрс Ш. Изучаем Node.js — Спб.: Питер, 2014.