На правах рукописи

Тарасов Алексей Григорьевич

СИСТЕМА МОНИТОРИНГА ВЫЧИСЛИТЕЛЬНОГО КЛАСТЕРА РАСШИРЕННОЙ ФУНКЦИОНАЛЬНОСТИ

05.13.11 — Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

кандидата технических наук

Иркутск — 2011

Работа выполнена в Учреждении Российской академии наук Вычислительном центре Дальневосточного отделения РАН (ВЦ ДВО РАН).

Научный руководитель:	член-корреспондент РАН, профессор **Смагин Сергей Иванович**
Официальные оппоненты:	доктор физико-математических наук **Федорук Михаил Петрович**
	доктор физико-математических наук, профессор **Корольков Юрий Дмитриевич**
Ведущая организация:	**Институт вычислительного моделирования СО РАН** (г. Красноярск)

Защита состоится **31 марта 2011 г.** в **15** ч. **00** мин. на заседании Диссертационного совета Д 003.021.01 в Учреждении Российской академии наук Институте динамики систем и теории управления Сибирского отделения РАН (ИДСТУ СО РАН) по адресу: 664033, Иркутск, ул. Лермонтова, 134.

С диссертацией можно ознакомиться в библиотеке ИДСТУ СО РАН.

Автореферат разослан **28 февраля 2011 г**.

Ученый секретарь диссертационного совета

д.ф.-м.н. А.А. Щеглова

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность. При использовании вычислительного комплекса (ВК) разработчикам необходимо решить ряд задач. Одной из них является развертывание средств управления ВК, а также системы мониторинга (СМ). Традиционно для исследования вопросов, связанных с организацией сбора данных и контроля ВК, получения статистики о работе локальных вычислительных сетей (ЛВС) и программных комплексов, разрабатывалось прикладное программное обеспечение (ПО). Исследованиями в этом направлении, в том числе практической реализацией, занимались такие авторы как R. Wolski, I. Foster, M. Genersereth и др. Значительные результаты были достигнуты и в смежных областях: в теории управления и управляющих систем, теории автоматов, системном анализе и системах поддержки принятия решений (А.А. Ляпунов, M. Morton, С.В. Яблонский, О.И. Ларичев, Н.Н. Моисеев и др.).

Тем не менее, задачи мониторинга по-прежнему являются сложными для практической реализации. Мониторинг компонент программно-аппаратного комплекса и оказание управляющих воздействий на него являются критически важными для организации высокопроизводительных распределенных вычислений. Пользователь и администратор кластера нуждаются в информации о том, как выполняется задание, какое влияние оно оказывает на вычислительную систему в целом. При этом с ростом числа вычислительных узлов в составе ВК возрастает и объем данных, которые приходится анализировать администратору, вследствие чего растет и вероятность ошибки ввиду человеческого фактора.

Многие СМ успешно и эффективно решают задачи мониторинга в частных случаях. Тем не менее, для ВК, на которых установлено устаревшее ПО мониторинга, зачастую переход на новые системы мониторинга затруднен, а устаревшее ПО уже не удовлетворяет возросшим требованиям. Несмотря на все многообразие архитектур и ПО СМ, до сих пор отсутствует возможность интеграции различных СМ между собой, затруднено добавление возможностей к уже разработанным СМ, сложен переход от контроля на низком уровне к более высокому и наоборот. Поэтому существует потребность в разработке новых эффективных подходов к архитектуре СМ.

Цель работы состоит в разработке архитектуры и реализации СМ ВК расширенной функциональности, которая представляет механизмы взаимодействия сторонних СМ и программных комплексов в рамках единой СМ. Для достижения этой цели были поставлены следующие задачи:

1. Разработать архитектуру СМ, позволяющей производить интеграцию построенных на ее основе приложений с различным ПО, в том числе сторонними СМ с учетом современных требований к ним.

2. Реализовать кросс-платформенную СМ, построенную на базе разработанной архитектуры.

3. Применить предложенные подходы для реализации программных комплексов решения задач как общего характера, так и представляющих интерес для узкоспециализированных приложений.

Объектом исследования являются теория и практика организации систем мониторинга вычислительных кластеров.

Предметом исследования являются методы и инструментальные средства контроля разнородных данных в вычислительном кластере.

Методы исследования. При решении поставленных задач использовались методы системного и прикладного программирования; методы объектно-ориентированного и модульного программирования; технологии построения вычислительных кластеров и разработки приложений, работающих в веб-среде.

Научная новизна результатов диссертации состоит в представлении подхода к архитектуре СМ, позволяющего осуществлять автоматизированный контроль ресурсов ВК и улучшить эффективность взаимодействия СМ со сторонним программным обеспечением через механизм отклика на изменения в работе ВК.

Предложен способ наращивания возможностей мониторинга, в том числе путем добавления сторонних модулей к уже используемым СМ без останавки их работы, а также подход к осуществлению совместного мониторинга и анализу данных, поступающих от различных СМ.

Для взаимодействия с независимыми модулями СМ, в том числе реализующим нейросетевые алгоритмы с целью определения текущего состояния вычислительного кластера, предложено использовать авторскую методику уведомления СМ о событиях, позволяющую в процессе работы СМ выполнять подписку на уведомления.

Эффективность предложенных подходов подтверждается тестированием на прикладных задачах, связанных с контролем ВК.

Достоверность полученных в работе результатов и эффективность разработанной архитектуры СМ подтверждается опытом успешной практической эксплуатации разработанного автором диссертации ПО Grate в Вычислительном центре ДВО РАН при мониторинге ряда ресурсоемких прикладных задач и результатами вычислительных экспериментов.

Практическая значимость. Разработанное ПО позволяет повысить эффективность и надежность использования вычислительных кластеров.

Создание и внедрение программных средств, предложенных в диссертации, выполнялись в рамках проекта по внедрению экспериментального

кластера ВЦ ДВО РАН и работ по созданию РВС ВЦ ДВО РАН. Разработанная СМ Grate помогает выполнять мониторинг вычислительных задач, производящих расчеты по плановым научно-исследовательским работам.

Предложенная архитектура СМ может использоваться как для разработки СМ вычислительных кластеров (основная область применения разработанной архитектуры), так и для создания СМ для других областей, например для систем контроля потоков данных в локальной вычислительной сети. В процессе разработки системы автором были получены дополнительные результаты: выполнено тестирование эффективности функционирования вычислительных кластеров, узлы которых находятся под управлением гипервизора Xen; выявлены значимые для работы СМ метрики.

Соответствие диссертации паспорту специальности. Полученные в диссертации результаты соответствуют области исследования специальности 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей, включающей в себя исследования моделей, методов, алгоритмов, языков и программных инструментов для организации взаимодействия программ и программных систем (пп. 1, 3 области исследования).

Апробация. Результаты работы были представлены на XXX, XXXII и XXXIII математических школах-семинарах им. Е.В. Золотова (Хабаровск, 2005 г., Владивосток, 2007, 2008 гг.), Всероссийской научной конференции “Научный сервис в сети Интернет: многоядерный компьютерный мир” (Новороссийск, 2007 г.), Межрегиональной научно-практической конференции “Информационные и коммуникационные технологии в образовании и научной деятельности” (Хабаровск, 2008 г.), Международной научной конференции “Параллельные вычислительные технологии” (Санкт-Петербург, 2008 г.), XI Всероссийской конференции молодых ученых по математическому моделированию и информационным технологиям (Иркутск, 2010 г.), Международной научной конференции “Russia and Pacific Conference on Computer Technology and Applications” (Владивосток, 2010 г.).

Исследования по теме диссертации проводились в рамках проектов по программам ДВО РАН: “Распределенные информационные вычислительные системы для комплексных научных исследований” (№ гос. регистрации 0120.0 603769) с 2006 по 2008 гг., “Организация работы вычислительного кластера в режиме удаленного доступа” и “Развитие системного программного обеспечения вычислительного кластера”, а также в рамках федеральной целевой программы “Научные и научно-педагогические кадры инновационной России” (проект № 02.740.11.0626) и грантов ДВО РАН № 09-I-П1-01 и 10-III-В-01И-008.

Публикации. Результаты научных исследований отражены в 15-и

научных работах, включающих в себя 2 статьи в журналах, рекомендованных ВАК для опубликования результатов диссертаций, 2 препринта, 1 свидетельство об официальной регистрации программы. В перечисленных публикациях все результаты, связанные с алгоритмизацией, программной реализацией и проведением вычислительных экспериментов на высокопроизводительных системах, получены автором лично. Результаты вычислительных экспериментов интеграции системы мониторинга и компоненты искусственных нейронных сетей, полученные совместно с Писаревым А.В., являются неделимыми. Из совместных работ и публикаций с Сапроновым А.Ю., Шаповаловым Т.С., Пересветовым В.В., Смагиным С.И., Щербой С.И. в диссертацию включены только те результаты, которые принадлежат лично автору.

Личный вклад автора. Автором осуществлена выработка требований к системе мониторинга, разработка трехуровневой архитектуры и реализация системы мониторинга расширенной функциональности, получение основных результатов проведенных исследований.

Структура и объем работы. Диссертация состоит из введения, трех глав, заключения, библиографического списка из 96 наименований, глоссария и 4 приложений. Общий объем работы — 145 страниц, в том числе 28 рисунков и 3 таблицы. Результаты главы 1 опубликованы в [2, 3, 5, 13], результаты главы 2 опубликованы в [4, 11, 12, 14]. Результаты главы 3 опубликованы в [1, 6–10, 15].

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обосновывается актуальность работы, формулируются цели и задачи исследования, приводятся основные положения работы. Анализируется специфика предметной области, существующие подходы и их реализация.

В первой главе поставлены основные задачи, решаемые системами мониторинга (СМ):

1) получение от контролируемой вычислительной системы значений множества контролируемых характеристик ВК и их сохранение;

2) выявление значимых событий;

3) запуск средств реагирования (отклика) на значимое событие.

Там же сформулированы дополнительные требования, предъявляемые к СМ. СМ должна предусматривать:

- возможность уведомления приложений пользователя о значимых событиях, возникающих при работе ВК;

- корректную обработку исключительных ситуаций: задержки и ошибки при получении данных, отсутствие либо повреждение необходимой информации и т.п.;
- набор библиотек и программных интерфейсов, позволяющих интегрировать СМ в различные программные комплексы;
- возможность получения и обработки данных сторонних СМ единообразно и прозрачно для пользователя;
- надежную и стабильную работу.

Для выполнения этих требований предложена новая архитектура СМ, передача данных в которой может осуществляться лишь между соседними уровнями. Это позволяет изменять и дополнять уровни, не нарушая общей работоспособности СМ.

Нижний уровень ответственен за сбор и представление данных в виде метрик. Источником данных может быть какой-либо сенсор физического или логического устройства, например SNMP-статистика или данные служб сторонних СМ. *Промежуточный уровень* отвечает за сбор данных с нескольких источников данных, преобразование их к унифицированному формату, проверку простейших триггеров и выполнение соответствующих им действий. Этими задачами занимаются *центры сбора данных* (ЦСД). На этом же уровне функционируют *ретрансляторы событий*. *Уровень приложений* отвечает за анализ данных и представление их пользователю. Выполняющееся на нем ПО может работать за пределами ВК, обращаясь лишь к данным, предоставляемым промежуточным уровнем.

На рис. 1 представлена общая схема получения и обработки данных мониторинга в системе, построенной с использованием разработанной архитектуры.

Предложенная архитектура позволяет динамически формировать *иерархическую структуру данных* (ИСД), узлом которой может выступать любой определяемый источником данных программный или аппаратный компонент. При этом базовыми единицами хранения данных являются значения метрик.

Метрика — это набор значений определенного типа, отражающий текущее состояние или изменение характеристики в конкретном узле. Метрики бывают статические, т.е. неизменяемые во время работы ОС (например, объем физической памяти), и динамические (например, время работы вычислительного узла). Также они делятся на метрики с накоплением, которые обеспечивают доступ ко всем данным измерений, и одномоментные: хранят только текущее значение. СМ отслеживает непротиворечивость данных по времени, соблюдение порядка помещения данных в метри-

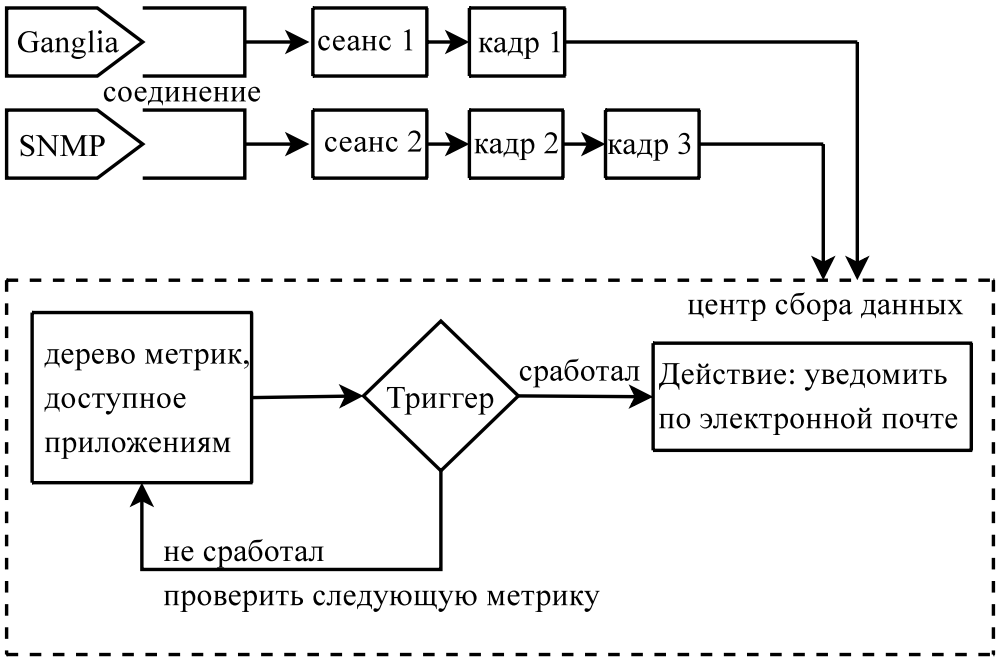

Рис. 1: Схема работы системы мониторинга, построенной по предлагаемой архитектуре

ку. Узел имеет изменяемые во времени атрибуты и набор метрик. Отличие атрибута от метрики состоит в том, что атрибут всегда хранит только одно значение.

Каждый узел, за исключением корня иерархии, всегда связан с родительским узлом, а также может иметь набор дочерних узлов. Важным свойством является возможность динамического формирования иерархической структуры, узлом которой может выступать любой определяемый источником данных программный или аппаратный компонент. При мониторинге РВС могут быть сформированы метрики РВС (например, число кластеров), метрики кластеров (число вычислительных узлов) и т.п. Узел имеет изменяемые во времени атрибуты и набор метрик. Отличие атрибута от метрики состоит в том, что атрибут всегда хранит только одно значение.

Сборку всех метрик в единое дерево осуществляет ЦСД, который может производить также агрегирование данных и генерировать значения специальной метрики на основе набора однотипных метрик. Например, метрика общей загрузки ВК сформирована на основе данных об использовании процессоров всех узлов ВК.

Каждый сенсор (источник данных) может производить измерение множества метрик, возможно неодинаковое для различных источников. Через определенные промежутки времени источник данных делает мгновенный снимок состояния метрик и передает его в ЦСД, ответственный за обработку информации.

При получении метрик от сенсоров используются соединения, сеансы и кадры. *Кадр* — это набор данных, характеризующих метрики и их зна-

чения в определенный момент времени. Кадр является логическим представлением данных, не зависящим от типа соединения, по которому данные могут передаваться в различных форматах.

При помощи *соединения* обеспечивается доступ к данным посредством сетевых коммуникаций, открытия локального файла, запроса к базе данных и т.д. Соединение можно рассматривать как реализацию механизма получения метрик в неструктурированном формате от сенсора, к которому единожды при инициализации устанавливается подключение.

Последний участвующий в получении данных от сенсоров объект — это *сеанс*, используемый для идентификации итерации процесса получения данных.

Следующими важными структурными элементами предлагаемой архитектуры СМ являются триггер и действие. *Триггер* — это логический элемент, содержащий условие или выражение, истинность которого необходимо проверить для последнего значения метрики. Если условие выполнено, то состояние триггера изменяется, что приводит к выполнению заранее определенных действий. Основная задача триггеров — проверка выполнения простых условий, накладываемых на значения метрик. Предполагается, что проверка триггеров проводится на уровне ЦСД.

Под *действием* понимается набор инструкций, реакция на срабатывание триггера. Одному триггеру может быть сопоставлено несколько действий и наоборот: одному действию может быть сопоставлено несколько триггеров.

В ряде случаев необходимо выполнять действия при одновременном выполнении нескольких условий. С этой целью в диссертации предложено использовать агрегаторы, которые включают в себя триггеры или другие агрегаторы, каждый из которых проверяет результат выполнения логических операций «И», «ИЛИ» для набора указанных элементов.

Важным средством, обеспечивающим удобную работу с триггерами, является *генератор триггеров* (или мета-триггер) — объект, создающий триггеры для метрик, подходящих под заданные условия. Необходимость этого элемента продиктована динамической структурой иерархии узлов СМ, при которой часть узлов уже может быть создана (и иметь активные триггеры), другая — отсутствовать и формироваться по мере необходимости. Мета-триггеры облегчают добавление новых узлов в вычислительную систему путем создания триггеров на базе указанного шаблона без вмешательства пользователя или администратора.

У подхода с использованием триггеров имеются недостатки: необходимость реализовывать на уровне проверки триггеров нестандартный способ уведомления для различных приложений, а также невозможность структурированной и иерархической обработки события. Поддержка уве-

домлений о событиях позволяет устранить перечисленные недостатки (см. рис. 2).

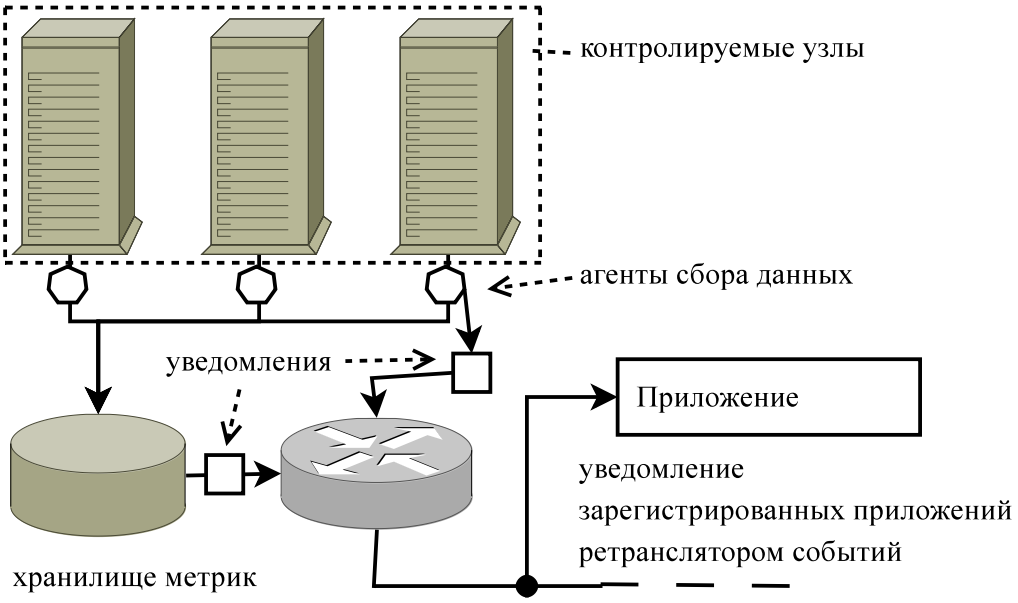

Рис. 2: Схема работы системы уведомлений

Каждый контролируемый узел СМ может генерировать уведомление о возникшем событии. В процессе работы СМ пользовательские приложения “подписываются” (регистрируются) на ретрансляторе событий, в качестве которого обычно выступает ЦСД. При подписке указывается, в уведомлениях о каких именно событиях заинтересовано приложение. ЦСД получает информацию о событии и отправляет уведомления зарегистрированным приложениям. Затем сообщение ретранслируется на другие ЦСД.

Клиентское приложение может обрабатывать события, игнорировать их или же генерировать новые события. Например, при возникновении события “отказ сетевой платы” подписавшееся приложение может сгенерировать более конкретные события: “необходимость миграции задач с узла”; “необходимость удаления узла из списка ресурсов вычислительного кластера”, о которых, в свою очередь, могут быть уведомлены другие подписчики. Если подписок на одно и то же событие несколько, то приложения получат уведомление в порядке, зависящем от приоритета, указанного при подписке.

Приложения, занимающиеся анализом метрик, выполняются на уровне приложений и принимают данные в структурированном виде от ЦСД посредством явного запроса данных. В зависимости от реализации запрос может содержать указания по агрегированию необходимых данных: осреднить данные по всем узлам, показать суммарную информацию и т.д. Возможно использование пользовательских модулей формирования отчетов для передачи значений метрик стороннему ПО в наиболее подходящем

формате.

Такая архитектура решает все задачи мониторинга, сформулированные в начале первой главы. Разделение элементов СМ по трем уровням позволяет реализовать опрос различных сенсоров единообразно, тем самым обеспечив совместную работу нескольких СМ. Применение механизма подписки обеспечивает взаимодействие СМ со сторонним ПО.

Вторая глава демонстрирует особенности авторской программной реализации СМ Grate, структуру модулей и интерфейсов, которые необходимы для интеграции СМ в более сложные программные комплексы. Описывается ряд интерфейсов и классов языка программирования Java, объединенных в пакеты для более удобного взаимодействиях со сторонним программным обеспечением.

Нередко ВК состоят из комплектующих от разных производителей или различных по своим характеристикам компонент. Поэтому в такой гетерогенной среде важно достичь межплатформенной переносимости программ. Это явилось одним из важных факторов при выборе для создания практческой реализации языка программирования Java, который является объектно-ориентированным, независимым от архитектуры машины, многопотоковым и динамическим языком программирования.

СМ Grate представляет собой библиотеку классов языка Java, обеспечивающую базовый набор возможностей, аналогичных таким широко распространенным системам, как Ganglia, Nagios и т.д. Этот набор упрощает взаимодействие различного ПО мониторинга друг с другом. При минимальных изменениях возможно использование разработанного ПО совместно с более специализированными СМ. СМ Grate состоит из нескольких независимо функционирующих модулей: grated, grape (Java-апплет, выполняющийся в среде web-браузера), grate-client (приложение для простой визуализации и контроля данных). Модули используют общую библиотеку интерфейсов и классов, позволяющую им работать с метриками, триггерами и прочими объектами нашей архитектуры СМ. Все разработанные классы и интерфейсы объединены в 14 пакетов, отвечающих за различные аспекты работы СМ. В разделе 2.1 вводится соглашение о наименовании, которому следуют все разработанные интерфейсы, классы и их методы.

В разделе 2.2 раскрываются особенности реализации классов сбора данных мониторинга, поддержки сетевых соединений, определений интерфейсов кадра и сеанса. Большое внимание уделено реализации метрик, в частности, описывается механизм смены типа метрик для более эффективного использования вычислительных ресурсов.

Следующий раздел второй главы посвящен программной реализации механизма отклика, т.е. поддержке триггеров, действий и уведомлений о событиях. Дано описание основных классов, используемых в нашей реа-

лизации архитектуры расширенной функциональности. Приведен пример записи события и запроса подписки об уведомлениях в формате XML.

Раздел 2.4 содержит определения типов значений метрик, классов поддержки вывода графиков и табличного представления данных для облегчения интеграции разработанного ПО в сторонние GUI-приложения. CM Grate поддерживает основные элементарные типы данных и тип последовательности двоичных данных. Последний используется для хранения неизвестных типов данных, полученных от сенсоров. Визуализация данных может осуществляться в виде таблицы или 2d-графиков (см. рис. 3).

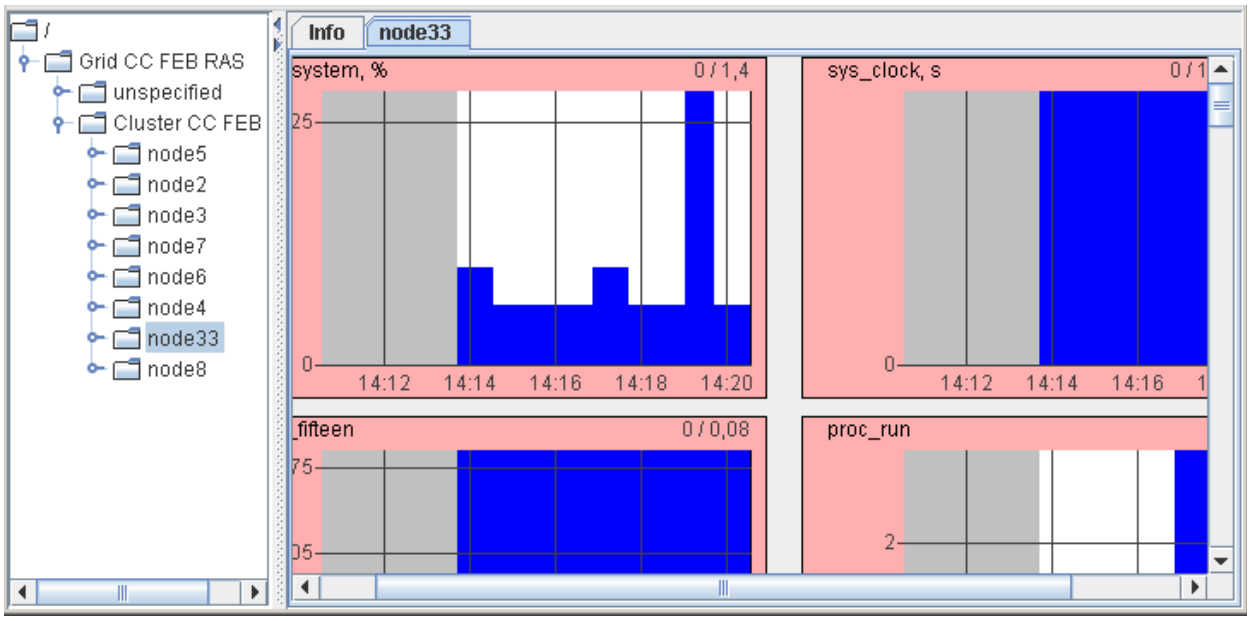

Рис. 3: Отображение значений метрик при помощи GPlateGraph

Завершает главу сравнение разработанного в диссертации ПО с доступными в настоящее время CM. Показаны преимущества Grate в сравнении с другими CM, указаны характерные особенности авторской CM. Существенным достоинством Grate является ориентация на интеграцию различных CM в рамках единого программного комплекса и возможность использования независимых программных продуктов в качестве модулей расширения сторонних CM.

Третья глава посвящена применению разработанных подходов для решения ряда задач, тесно связанных с мониторингом ВК. Там же приводятся примеры других задач, решение которых может быть упрощено при использовании предлагаемых подходов.

В разделе 3.1 описывается типичная схема использования разработанных классов. Ядро разработанного программного комплекса составляет класс, реализующий интерфейс GWorld. Экземпляр этого класса осуществляет контроль поступления данных и создание остальных важных для работы комплекса классов, инкапсулируя всю доступную приложению

информацию от СМ.

В начале работы создается экземпляр необходимого класса, реализующего интерфейс GWorld. В зависимости от конкретного класса могут потребоваться вызовы дополнительных методов экземпляра, однако обычно инициализация происходит через метод init() глобального объекта G, через статическое поле theWorld которого все созданные классы могут обращаться после его инициализации.

Для того чтобы создать объект, реализующий интерфейс GSentinel, клиентская часть приложения вызывает метод createConnection(). Этот объект при инициализации получает классы соединения и кадра. В процессе работы GSentinel с указанной периодичностью создает новые сеансы, используя объект соединения и передавая полученный поток данных на обработку в объект кадра. В объекте кадра происходит обработка поступающих данных и вызов методов объекта theWorld для создания новых узлов и метрик. В результате новые значения поступают в метрики. Если объем памяти, хранящей эти значения, превышает некоторый предел, определяемый типом метрики, происходит передача данных во внешнее хранилище с помощью интерфейса GMetricCacheman, запрашиваемого у объекта theWorld.

При успешной обработке поступивших данных вызывается метод GWorld.commit(), чтобы сообщить об изменениях в метриках объектам СМ, которые, возможно, зарегистрированы для уведомления о событии изменения в наборе данных. Приложение пользователя также может проверять уведомления о событиях. Для этого необходимо создать экземпляр класса GEventClientUDP и зарегистрировать объект, методы которого вызываются при получении нового уведомления.

В разделе 3.2 приведен подход к расширению функциональных возможностей установленной СМ для штатного отключения вычислительных узлов при их значительном перегреве, что актуально для неспециализированных комплектующих. Модуль grated проводил проверку триггеров (по одному на каждый узел), расширяя тем самым базовые возможности используемой СМ Ganglia. Визуализация выполнялась grate-client на основе данных, поставляемых от grated, тем не менее веб-сервис СМ Ganglia также мог визуализировать данные, получаемые от grated.

Важным преимуществом описанного подхода является расширение функциональных возможностей установленной СМ без остановки ее работы и без перенастройки источников данных. Замена агента сбора данных позволяет получить сравнимую или лучшую производительность при более широких возможностях итоговой системы. Получить аналогичный результат с использованием других доступных СМ было невозможно. При этом результаты применения СМ Grate показывают, что разработанное ПО по-

требляет меньше вычислительных ресурсов, предоставляя при этом больше возможностей по сравнению с CM Ganglia.

Уровень приложений может быть применен и для более сложных управляющих операций над контролируемой системой. В качестве примера приведена задача повышения эффективности использования ВК. В разделе 3.3 главы приведен подход к решению подобной задачи для вычислительного кластера, узлы которого работают под управлением гипервизора системы виртуализации Xen. Системы виртуализации позволяют запускать несколько копий операционных систем на одной машине.

При использовании Xen на экспериментальном кластере была добавлена возможность миграции выполняющихся процессов с одного вычислительного узла на другой, что позволило достичь полноценной утилизации ресурсов физического вычислительного узла несколькими виртуальными вычислительными узлами. Для тестирования взаимодействия Xen с CM Grate был создан отдельный логический кластер. Во время тестирования осуществлялся процесс миграции выполняющейся копии гостевой ОС с запущенными вычислительными задачами с одного из физических узлов на другой, обладающий большими доступными ресурсами на момент миграции.

Программный комплекс на базе CM Grate анализировал использование ресурсов на вычислительном узле, обрабатывая данные, предоставляемые промежуточным уровнем CM. При необходимости триггером инициировалось выполнение shell-скрипта с командами управления Xen. При этом не происходило значительного простаивания вычислительного процесса. Как показали тестовые замеры, отраженные на рис. 4, для малого числа узлов, участвующих в расчете параллельной задачи с интенсивной передачей данных между узлами, потери производительности были малы. Для задач, в меньшей мере зависящих от интенсивности обмена данными между вычислительными узлами, потери производительности будут меньше в силу меньшего времени простоя ВК в ожидании завершения вычислений на узле, подвергшемся миграции. При отсутствии миграций использование Xen не приводит к заметным потерям производительности (11 ГФлоп является реально достижимой производительностью согласно тесту Linpack для экспериментального кластера).

В разделе 3.4 рассматривается другой подход к повышению эффективности использования ВК. С ростом числа узлов и контролируемых характеристик системному администратору становится все сложнее в непрерывном режиме следить за негативными изменениями в значениях опрашиваемых характеристик ВК, в связи с чем возникает потребность в автоматизации рутинных операций. При этом многие изменения в работе ВК могут быть обнаружены и в ряде случаев предотвращены с использованием

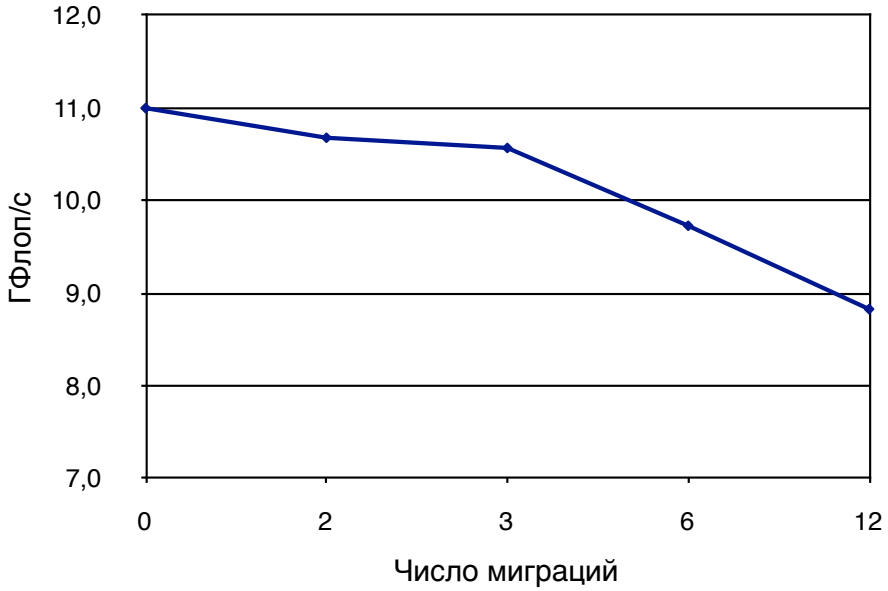

Рис. 4: График падения производительности при миграциях гостевой ОС

различных систем автоматизированного контроля.

Примером обнаруживаемых изменений могут служить последствия неэффективно работающего приложения, например, из-за простаивания на операциях ввода-вывода (рис. 5, область A).

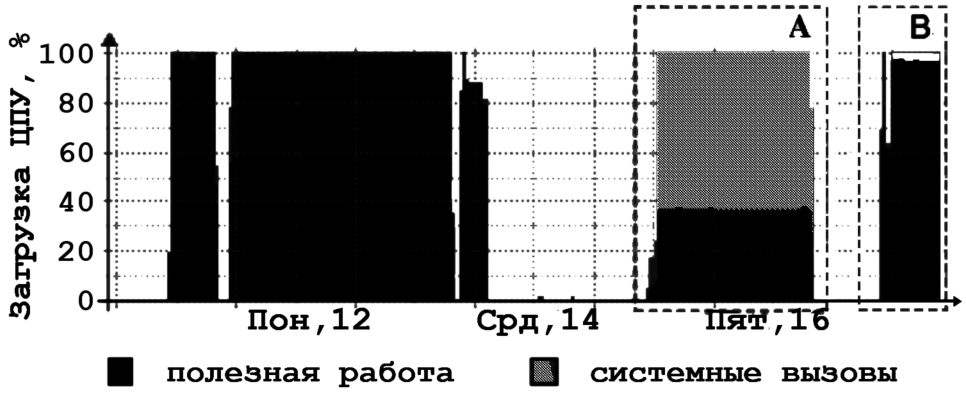

Рис. 5: График неэффективного использования ресурсов приложением пользователя

Такая ситуация хорошо распознается обученной искусственной нейронной сетью (ИНС) и может быть предотвращена при своевременном вмешательстве администратора ВК или пользователя (рис. 5, область B). Разработанная CM Grate позволяет использовать стороннее ПО для расширения функциональных возможностей, что позволило осуществить взаимодействие с модулем ИНС для определения текущего состояния вычислительного кластера на основе данных, поставляемых CM. Обратный отклик модуля был реализован через механизм уведомлений о событии. В экспери-

менте участвовали четыре компонента: вычислительный кластер, система мониторинга, приложение-обработчик событий и модуль ИНС.

Приложение-обработчик события и модуль grated СМ Grate запускались для исполнения на управляющем узле, модуль ИНС выполнялся на отдельной вычислительной станции. Для посылки уведомления и получения значений метрик модуль ИНС обращался к модулю grated.

Имея на входе полученные системой мониторинга данные метрик, ИНС встречного распространения при 12 нейронах и 30 циклах обучения верно относит текущее состояние к одному из ранее определенных во время обучения классов состояний. При изменении класса модуль ИНС посылает уведомление о событии в СМ, которая в свою очередь передает его приложению-обработчику.

В тестировании использовались ИНС с относительно несложной топологией для определения очевидных изменений в работе ВК в автоматизированном режиме. Тем не менее, развитие модуля ИНС позволяет расширить возможности СМ, не переписывая при этом исходного кода Grate. Более того, уведомления о событиях и данные мониторинга могут обрабатываться одновременно несколькими приложениями, что позволяет более полно анализировать состояние ВК с применением не только ИНС, но и других методов и алгоритмов.

Третья глава завершается обсуждением возможных путей совершенствования разработанной системы мониторинга. Указаны потенциальные недостатки реализации на языке программирования Java, отмечены свойства архитектуры, которые могут помешать ее масштабированию при росте числа контролируемых узлов. Приводятся варианты улучшения реализации разработанной архитектуры.

В заключении сформулированы основные результаты диссертационной работы.

РЕЗУЛЬТАТЫ, ВЫНОСИМЫЕ НА ЗАЩИТУ

1. Разработана новая архитектура СМ, обладающая широким набором функциональных возможностей. Создана ее программная реализация на языке высокого уровня Java.

2. Предложен и исследован механизм взаимодействия СМ со сторонним ПО посредством подписки и уведомлений о событиях. Показана эффективность такого взаимодействия в задачах автоматизированного контроля ВК при решении прикладных задач, общих для широкого круга вычислительных кластеров.

СПИСОК ПУБЛИКАЦИЙ ПО ТЕМЕ ИССЛЕДОВАНИЯ

1. Тарасов А.Г. Интеграция системы мониторинга расширенной функциональности и нейросетевого модуля // Информатика и системы управления. 2010. № 4 (26). С. 104–115.

2. Тарасов А.Г. Практический подход к взаимодействию систем мониторинга вычислительных кластеров со сторонним программным обеспечением // Вычислительные методы и программирование. 2011. Т. 12. С. 1–8.

3. Тарасов А.Г. Расширяемая система мониторинга вычислительного кластера // Вычислительные методы и программирование. 2009. Т. 10. С. 147–158.

4. Тарасов А.Г. Grated: Свидетельство о регистрации программы для ЭВМ № 2010615286 от 17.08.2010. РОСПАТЕНТ.

5. Тарасов А.Г. Трехуровневая система мониторинга расширенной функциональности // Параллельные вычислительные технологии: Тр. Междунар. конф. (Санкт-Петербург, 28 января – 1 февраля 2008 г.). Челябинск: Изд-во ЮУрГУ, 2008. С. 464–469.

6. Тарасов А.Г., Писарев А.В. Модульная нейросетевая система мониторинга вычислительного кластера // Материалы Межрегион. конф. “Информационные и коммуникационные технологии в образовании и научной деятельности”. Хабаровск: ТОГУ, 2009. С. 289–296.

7. Тарасов А.Г. Совместное применение системы мониторинга вычислительного кластера и системы виртуализации XEN // Информационные и коммуникационные технологии в образовании и научной деятельности: Материалы конф. Хабаровск: Изд-во ТоГУ, 2008. С. 304–310.

8. Тарасов А.Г. Опыт интеграции системы мониторинга Grate со сторонними программными комплексами // Материалы Междунар. науч.-практ. конф. “Суперкомпьютеры: вычислительные и информационные технологии” (Хабаровск, 30 июня – 2 июля 2010 г.). Хабаровск: Изд-во Тихоокеан. гос. ун-та, 2010. С. 125–132.

9. Tarasov A.G. Integration of computing cluster monitoring system // Proc. of First Russia and Pacific Conf. on Computer Technology and Applications (Vladivostok, Russia, September 6–9, 2010). Vladivostok: IACP FEB RAS, 2010. P. 221–224.

10. Пересветов В.В., Сапронов А.Ю., Тарасов А.Г., Шаповалов Т.С. Удаленный доступ к вычислительному кластеру ВЦ ДВО РАН // Вычислительные технологии. 2006. Т. 11, спец. вып. С. 45–51.

11. Пересветов В.В., Тарасов А.Г., Сапронов А.Ю., Шаповалов Т.С. Организация работы вычислительного кластера в режиме удаленного доступа. Хабаровск, 2007. 34 с. (Препринт / ВЦ ДВО РАН; № 110).

12. Пересветов В.В., Тарасов А.Г., Сапронов А.Ю. Вычислительный кластер бездисковых рабочих станций. Хабаровск, 2005. 50 с. (Препринт / ВЦ ДВО РАН; № 83).

13. Смагин С.И., Тарасов А.Г., Сапронов А.Ю., Шаповалов Т.С., Пересветов В.В. Web и GRID технологии обеспечения доступа к ресурсам вычислительного кластера ВЦ ДВО РАН // Материалы Межрегион. конф. “Информационные и коммуникационные технологии в образовании и научной деятельности” (Хабаровск, 21–23 мая 2008 г.). Хабаровск: Изд-во ТоГУ, 2008. С. 69-76.

14. Тарасов А.Г., Сапронов А.Ю., Шаповалов Т.С. Доступ к ресурсам вычислительного кластера через сеть Интернет // VII Всерос. конф. молодых ученых по математическому моделированию и информационным технологиям. Красноярск: Изд-во НГУ, ИВТ СО РАН, 2006. С. 93–95.

15. Тарасов А.Г., Сапронов А.Ю., Шаповалов Т.С. Применение системы виртуализации XEN на вычислительных кластерах // VIII Всерос. конф. молодых ученых по математическому моделированию и информационным технологиям. Новосибирск: Изд-во НГУ, ИВТ СО РАН, 2007. С. 106–107.

Редакционно-издательский отдел
Учреждения Российской академии наук
Института динамики систем и теории управления СО РАН
664033, Иркутск, ул. Лермонтова, д. 134
Подписано к печати 25.02.2011 г.
Формат бумаги 60×84 1/16, объем 1,2 п.л.
Заказ № 3. Тираж 120 экз.

Отпечатано в ИДСТУ СО РАН