



## 9. Среда за разпределена обработка

Васил Георгиев

 [v.georgiev@fmi.uni-sofia.bg](mailto:v.georgiev@fmi.uni-sofia.bg)  
 [ci.fmi.uni-sofia.bg/](http://ci.fmi.uni-sofia.bg/)

## Съдържание

- Метасистеми и грид
- Сервизно-, моделно- и аспектино-базирани архитектури
- Софтуерни агенти

## Обзор на РСА

- терминални комплекси [с минимашини] – mainframe architectures
- разпределени файлови системи
- клиент-сервер
  - двуслойни РСА
  - трислойни РСА
  - $n$ -слойни РСА
- метасистеми
- грид системи
- сервизно-базирани архитектури (SOA)
  - Web-услуги

## Клъстерна обработка

- компютърните клъстери предоставят разпределен ресурс като виртуален уникompютър; типично:
  - хомогенна мултикомпютърна изпълнителна платформа в интранет
  - обикновено без апаратна специализация на възлите (евентуално функционална специализация) – симетричен мултикомпютинг
  - обща администарция и защита; вътрешно отворен достъп
  - обикновено за специализирани приложения:
    - надеждна обработка (high-availability clusters) – пенликиране/дублиране на приложенията
    - балансиращи разпределени сървери (load-balancing clusters) – най-често в комуникационните общ портал, функционална хомогенност – за бързо обслужване
    - разпределен сървер данни
    - сърверна ферма
- «фермите» (computer farms, server clusters) са кмпютърни клъстери
  - със специализирани сърверни функции на различни сърверни възли или модули (вкл. апаратна специализация) – напр. числова обработка (разпределена с работни станции или паралелна с минимашини), дисково поле, комуникационен сървер и др.
    - приложение и за отказустойчива репликирана обработка
  - в един административен домейн и в обща високоскоростна опорна мрежа
  - за интензивна числова обработка или уеб-хостинг

## Метасистеми

- метасистемите (Legion, Condor, SNIPE) – осигуряват достъп до хетерогенни разпределени ресурси
- потребителят изпраща прозрачно заданието към виртуална разпределена система без да специфицира отделни компоненти – виртуален униконпютър
- предимно се прилагат в LAN системи отколкото WAN (проблемна скалируемост)
- отсъства концепцията за потребителска услуга освен предоставяните системни функции и поради това са приложими само в професионални среди
- слаба стандартизация
- проблемна преносимост – дилемата CORBA/Java (+Jini/Jxta)

## Споделени ресурси

- разпределени среди за споделяне на ресурси **между индивидуални потребители** – volunteer computing VC
- отдалечена обработка на асинхронни заявки във фонов (нископриоритетен) режим
- иерархия на потребителите – приоритетни локални потребители и опортюнистични потребители (CPU scavenging)
- основен оптимизиращ критерий е натоварване на свободните ресурси (вместо производителност или цена на обработка  $C = p * T_p$ )
- споделянето може и да е в рамките на една организация – с цел оптимизиране на натоварването – high throughput computing – Condor – или дори във VO – Condor-G
- активни VC проекти:
  - с общо предназначение: BOINC (Berkeley Open Infrastructure for Network Computing), distributed.net
  - специализирани: SETI@home

## Грид системи

- Раслоена програмна среда от за достъп до услуги, базирани върху разпределени и мрежови ресурси, които са споделени **между различни институции**
  - кординирано споделяне на програмно-апаратни ресурси в динамична **виртуална организация** (9.7)
- списък характеристики (checklist) за особеностите на грид-средите
  - споделени ресурси
  - без централизирана администрация
  - различна административна принадлежност на потребителите
  - системните протоколи и интерфейси са стандартни, отворени и с общо предназначение (без ориентация към конкретен клас приложения)
  - по-високи функционални (напр. метаслуги) и нефункционални (напр. производителност, отказоустойчивост..) изисквания към предоставяните услуги, отколкото в класическите РС

## Грид архитектура (9.8)

- универсални потребителски услуги –
  - идентификация(authentication) и допуск (authorization) – single sign in в рамките на BO
  - създаване на процеси (и нишки) с контролиран достъп до ресурсите
  - разпределено управление на свързаните процеси:
    - хоризонтално – единно адресно пространство обща памет (семафори/ключалки, монитори, глобални променливи и променливи на състоянието) или MOM
    - вертикално – комуникационни процеси: конвейризация, потоково управление workflow
  - комуникационни протоколи (принцип на пясъчния часовник върху [TCP]/IP )
- системни функции:
  - блокиране и освобождаване на заявени ресурси
  - диспечеризация (scheduling, mapping, ;matchmaking, load balancing)
  - защита
  - управление на виртуалната памет (файлови системи, бази данни, планиране на активните страници в паметта)
  - компютърно счетоводство и одит (accounting, audit)
- настолни, леки и безжични гридове

## Open Grid Services Architecture

- OGSA – описателен стандарт за грид-архитектура на GGF за изграждане на сервисно-базирани гридове с Web-обслужване
  - развива концепцията на Web-услугите в посока на разпределено управление на ресурситена ВО в грид
  - прилага се от Globus Alliance
  - специфицира системните интерфейси и услуги
- основни интерфейси на OGSA
  - регистрация и откриване** на услуги (със сервисно представяне на ресурсите) – директория на услугите
  - сервисни интерфейси** – заредим код за достъп до услугите
  - стандартен интерфейс за изграждане на **сервисни области** (комплексни услуги) от елементарни локални услуги
  - интерфейс за **дефиниране на цели** (policies): за разпределеното и прозрачно администриране на натоварването (вкл. резервирането и алоцирането на ресурси), комуникациите (вкл. QoS), защитата
  - интерфейс за **управление на данните**: разпределение, структура (БД/файлове), достъп, резервация, транспиране, репликиране, описание и откриване, транзакции
- основни системни услуги в OGSA
  - мониторинг** – наблюдение на ресурси и услуги – осигурява информация за разпределението, планирането (алокацията и ре-алокацията) на обслужващите процеси
  - ресурсно управление** (в OGSA clustering) – планиране на ресурси и услуги за оптимизация на обработката (load balancing, съгл. различни критерии), възстановяване след грешка (disaster recovery, seamless recovery), вкл. и за бартер на услуги/данни и др.
  - потокова композиция** – workflow – осигурява изпълнението на комплексни приложения като координирано изпълнение на набор от услуги – вкл. координация на съставните услуги, пренос на данните, интерфейси между услугите
  - одит** на данни и услуги – за резервиране на данни и ресурси, за оценка (billing) на участието на административните единици в ресурсите на ВО и за защита
  - защита** – супер-протокол, картиращ защитните протоколи на различните административни области в грида за постигане на прозрачна защита (принцип на най-слабото звено)

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

9

## Грид - слоест модел (9.10)

- инфраструктура (fabric) – ресурси за изчисление, запаметяващи устройства, мрежи, сензори и др.
- съобщителен слой (connectivity – “hourglass principle”) – малък брой протоколи, имплементирани за всички компоненти на инфраструктурата
  - осигуряват комуникация и защита и в резултат – изпълнението на едно приложение в различните компоненти на инфраструктурата
  - Globus Toolkit реализира основно този слой и се състои от протоколи и APIs;
- колективни (системни) услуги (collective services) – състои се от услуги, протоколи и APIs, които
  - осигуряват взаимодействието между ресурсите от инфраструктурата
  - разширяват разнообразието от услуги чрез комбиниране на малкия брой протоколи от съобщителния слой
  - типични функции са
    - директории/регистратори/браузъри на услуги и ресурси
    - ресурсни брокери, монитори, диагностика
    - репликиране на данни и услуги,
    - балансиране на изчислителния товар, отказустойчивост
    - защита на достъпа и т.н.
- приложни услуги (applications) – потребителски и интерфейсни приложения (напр. портали), които се състоят от обръщания към модулите от долните слоеве.

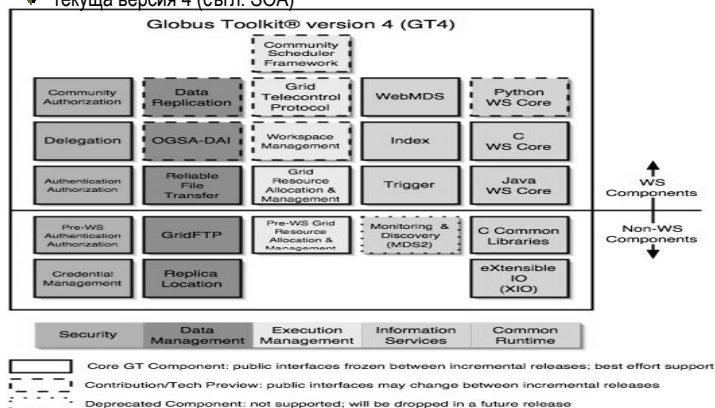
9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

10

## Globus Toolkit

- среда за изграждане на грид инфраструктура – отворен проект на Globus Alliance
- текуща версия 4 (съгл. SOA)



9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

11

## Грид приложения

- разпределен суперкомпютър (virtual unicomputer) – за обработка на големи задания във ВО
  - критерий за оптимизация – производителност (време за обработка)
- обработка по заявка (on-demand) – абонаментна обработка на инцидентно възникващи задания, за които потребителя няма локален обработващ капацитет
  - през определен период
  - критерий за оптимизация – коефициент стойност/ производителност
- пакетна фоновая обработка (high-throughput computing, htc)
- обработка на данни
  - критерий – оптимизация на 3У
- съвместна обработка (collaborative computing) – отдалечена съвместна обработка
  - критерий – функционалност (възможности за ефективно и защитено взаимодействие между потребителите)
  - проектиране на сложни системи от колективи
  - образование (вкл. интерактивни игри и симулации в реално време)
  - разпределено събиране и обработка на големи масиви от данни (астрономия, метеорология, медицина, бизнес и борси ...)

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

12

## Архитектура на грид приложенията - 9.13

- традиционни грид приложения (без услуги) – развитие на C-S в интернет
- координираният процес ползва грид инфраструктура – grid-aware
- централизирано управление на достъпа през координирания процес
- често централизирана БД за общия приложен контекст
- евентуално с Web интерфейси към локалните обслужващи процеси
- ограничена скалируемост и преносимост

## Архитектура на грид приложенията в SOA – 9.14

- приложението разделено на множество компоненти, взаимодействието между които не е централизирано
- всички бизнес-функции се дефинират като услуги; обслужващата платформа е също сервисно-базирана поради което системните услуги могат да са част от приложението (напр. за постигане на QoS):
  - бизнес-услуги
  - бизнес-транзакции, които са композиция от услуги от по-ниско ниво и системни услуги
  - системни услуги
- основни въпроси на архитектурния дизайн:
  - грануларност
  - модулност и преносимост (вкл. картиране към съществуваща грид-инфраструктура)
  - технологии и модели на разработка
- услугите са независими – принцип на черната кутия
- прозрачност на отдалечените обръщения посредством локално-заредими интерфейси, които изпълняват протокола на достъп до отдалечената услуга – обикновено в XML-формат
- така се заличава концепцията за общо (линейно) виртуално адресно пространство
- отдалечената услуга алтернативно може да е:
  - да е част от същото приложение (общо адресно пространство)
  - част от друго приложение или платформа от грида (разпределени адресни пространства – **“асиметричен мултипроцесинг”** – следователно аргументите на обръщението са стойности или абсолютни адреси, но не и локални относителни адреси)

## SOA

- бизнес- и системните функции се поддържат от дискретни изолирани процеси, дефинирани съгласно платформения стандарт за услуги
- услугите в SOA са
  - с функционална спецификация – т.е. покриват функция или група от функции прозрачно от инфраструктурата
  - грануларността им не се стандартизира (засега) – приложими са елементарни и комплексни услуги
    - технологично е елементарните услуги да се проектират като софтуерни компоненти – вж. MS Passport – компонент за идентификация при уеб-достъп – черна кутия; респ. комплексните услуги – като сива кутия (glass box)
  - независими и прозрачни – принцип на черната кутия и компонентни технологии
  - публичност и откриваемост (advertising)
  - проектират се със специализирани интерфейси за различни класове сървери, така че могат да се вграждат в съществуваща сърверна инфраструктура – SOA не изисква специализирана инфраструктура (освен при някои класове грид)
- интеграция на SOA-приложения:
  - потребителски слой (user interaction) – за [интерактивно] обслужване
  - свързаност (connectivity) – междусервизен интерфейс
  - интегриращ процес – междусервизно управление за “хореография” на потребителските сценарии
  - интеграция на данните – федериране или пренос на данни

## SOA - реквизит

- Web-услуги (9.16.1): използват се като устойчиви услуги в среда за публикуване, търсене/откриване и изпращане на заявки към съответната Web-услуга
- Web-базираните интерфейси (9.16.2): WSDL, SOAP, UDDI, XML форми (вместо RMI, Jini, CORBA, DCOM) – за преносимост, модулност, разширимост и глобален достъп
  - типично върху HTTP като приложен протокол (tunneling) за универсално разпространение в интернет (за преодоляване на защитни стени и прокси-процеси); но и други (SMTP)

## SOA - реквизит

- WSDL (Web Services Definition Language) – за **описание на услугите** като [колекции от] точки за достъп – т.е. мрежови портове и дефиниции на интерпретираните съобщения
  - портовете се дефинират с мрежов адрес и типизация (абстрактна колекция от поддържащите операции от услугата, протокола на обмен и формата на данните)
  - съобщенията се дефинират чрез формата на съдържанието им дефиниция на тип SOAP (Simple Object Access Protocol) – протокол за XML-спецификация на **отдалечени заявки** (вкл. обектни аргументи – контекст) към обектни методи
  - спрямо RPC/RMI
    - се различава по своята пълна платформена и езикова независимост
    - въвежда по-голям комуникационен свръхтовар (поради усложнената нотация с марки (tags))
    - си прилича по фиксирания модел CS (не напр. P2P – поради транспорта върху HTTP)
  - структурира съобщенията в обвивка/заглавие/тяло (envelope/header/body)
  - клиентските имплементации се вграждат в Java, Python, Perl и MS.Net приложения, а UDDI-директориите се поддържат от повечето сървени платформи – Apache, SAP, Windows Server 2003, Oracle и т.н.
- UDDI (Universal Description, Discovery and Integration Service) (9.17) – глобална платформено-независима **директория на услугите** (registry, service broker) с XML-записи в три дяла – бели, жълти и зелени "страници" респ. с адресна, бизнес- и техническа категоризация на услугите – общо предназначение вкл. и за Web-услуги
  - интерфейсът ѝ обслужва SOAP-съобщения
  - достъп до WSDL-описания на протоколите за свързване със съотв. услуга и на формата на съобщенията, които се интерпретират от услугата
- XML-формите: използват се във всички изброени стандартни протоколи, а също могат да бъдат използвани извън стандарта или в разширения като различни дескриптивни езици (напр. за описание на услуги като с WSDL)

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

17

## Моделно-базирани архитектури

- [MDA – Model-driven architecture]
- SW-архитектура, изградена в съответствие с принципите на моделното проектиране (Model-driven engineering)
- MDA представлява допълнение или средство за автоматизирано проектиране на приложения, използващи обичайни архитектурни подходи като OO, COTS или SOA
- абстрахира се инфраструктурата (QoS) както и системните или общите функционалности – приложението се разработва на базата бизнес модел
- детайлността на модела предопределя използването на автоматизирано проектиране
- широко се прилага формално специфициране с UML
- моделите допускат прилагането на индустриални стандарти, както и въвеждането на стандартизация в моделирането
- разработваните моделни стандарти (напр. от OMG – Object Management Group) всъщност формират SW-архитектура на базата на абстрактни (в смисъл на бизнес-ориентирани) модели
- спецификацията на моделните архитектури има предварителна моделна фаза – по което се различава от останалите SW-архитектури
- стандартната спецификация на абстрактни бизнес модели позволява модела да съдържа изпълними семантика и съответно прилагането на автоматизирана генерация на приложения
- моделната спецификация е платформено-независима (PIM – platform-independent model) и е ориентирана към проблемната област
- за генерация на изпълним код освен PIM е необходима и дефиниция на изпълнителните платформи на компонентите на модела (PDM – platform definition model: напр. CORBA, .NET, Web), при което моделът де транслира автоматично до Java, C#, PHP, Python
- автоматичната генерация на код при МБА може да се съчетае и с прилагане на различни други проектни подходи – най-често с ползване на програмни шаблони (design patterns)

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

18

## Аспектно-базирани архитектури

- [Aspect-Oriented System Architecture]
- SW-архитектура, изградена в съответствие с принципите на аспектно-базираното проектиране (AOSD – Aspect-oriented software development)
- "**аспекти**" (или още **crosscuts**) на разпределеното приложение са общи функционалности, които се прилагат от няколко от специализираните (и централни) бизнес-модули
  - напр.: дневник (log), оторизиране и защита
- поради това декомпозицията т.е. модулността на "аспектните" функционалности представлява проблем с много варианти на решение
- основен проблем е елиминирането (или стройната организация) на разпръснати и обркъани спецификации на "аспекти" поради разделянето на тези общи функционалности между модулите, които специфицират централните бизнес-функции
- обект на разработка за изграждане на цялостен архитектурен подход са стандартизацията, модуларизацията и шаблонизацията на различни категории функционални и нефункционални "аспекти"
- за целта се използват специализирани езици (**aspect oriented languages**) и платформи (тъй като аспектите не могат – или не е технологично – да бъдат модулариизирани с обичайните архитектурни абстракции – CS, p2p, OO, шаблони); тези средства подпомагат проектирането или ре-дизайна на общата архитектура с оглед на "чисто" аспектно проектиране
- AOSA се фокусира на идентифицирането, локализацията (или картирането) и спецификацията на аспектите в явен вид на етапа на проектиране на SW архитектурата

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

19

## Софтуерни агенти - 9.20

- Автономни (самоинициативни) процеси, изпълняващи задания съвместно с други [отдалечени] агенти; свойства:
  - **автономност и активност** – самоинициатива за въздействие върху дадени параметри на средата
  - **реактивност** – сканиране (с евентуална реакция) на средата
  - **комуникативност** – интерфейс/и към потребители и други агенти – по-специално съвместни (collaborative) агенти
  - **непрекъснатост (continuity)** – продължително и многократно изпълнение на функциите, не за всички типове агенти
  - **мобилност** – миграция между възли (може и при ниска мобилност на кода), не за всички; напр. агенти за извличане на информация от разпределени документни системи
  - **адаптивност** – еволюция на реакциите при еднакви параметри на средата, не за всички
- Изолирани класове агенти (освен общия клас Софтуерни агенти):
  - интерфейсни – потребителски достъп до приложения и други агенти с адаптивност (самообучение); напр. брокерски агент
  - информационни – подобно на интерфейсите – за управление на постъпваща информация

9. Среди за разпределена обработка

ФМИ/СУ \* ИС/СИ \* РИТАрх/РСА

20

## Технология на софтуерните агенти

- среда за поддържане на агенти (агентна платформа) – предоставя основните свойства на агентите като middleware-услуги
- FIPA модел на агентна платформа (на Foundation for Intelligent Physical Agents) – обслужва създаването и закриването на агенти, междуагентните комуникации и откриването на агенти
- агентната платформа за разпределена система включва модулите (фиг. 9.21.)
  - за управление (създаване, идентификация, съединение)
  - за директория (локална) – таблица идентификатори – атрибути; достъпна за отдалечени агенти
  - комуникационен канал – ACC (Agent Communication Channel) – обмен на съобщения (подредени и проверени) между агентските съединения напр. по Internet Inter-ORB Protocol (IIOP – при хетерогенни МС)

## Езици за междуагентни комуникации

- междуагентния обмен е на приложно (най-високо) ниво по протокол, възприеман като език – ACL (Agent Communication Language – FIPA)
- различава стандартизирана цел на съобщението (purpose) – табл. 9.22
- стандартна структура на ACL съобщенията –
  - цел – стандартна `∴ INFORM`
  - изпр. агент `∴ tsetsa@http://fanc1ub-karizma.abv.bg:7269`
  - приемащ агент `∴ miro@iiop://stulbica.bg:6217`
  - кодов език `∴ Prolog`
  - онтология – опция за интерпретация на съдържанието `∴ discography`
  - съдържание – съгл. кодирането [и онтологията]