XtreemOS

Enabling Linux for the Grid

Porting Applications to XtreemOS

Marjan Šterk, XLAB
marjan.sterk@xlab.si

Information Society Technologies

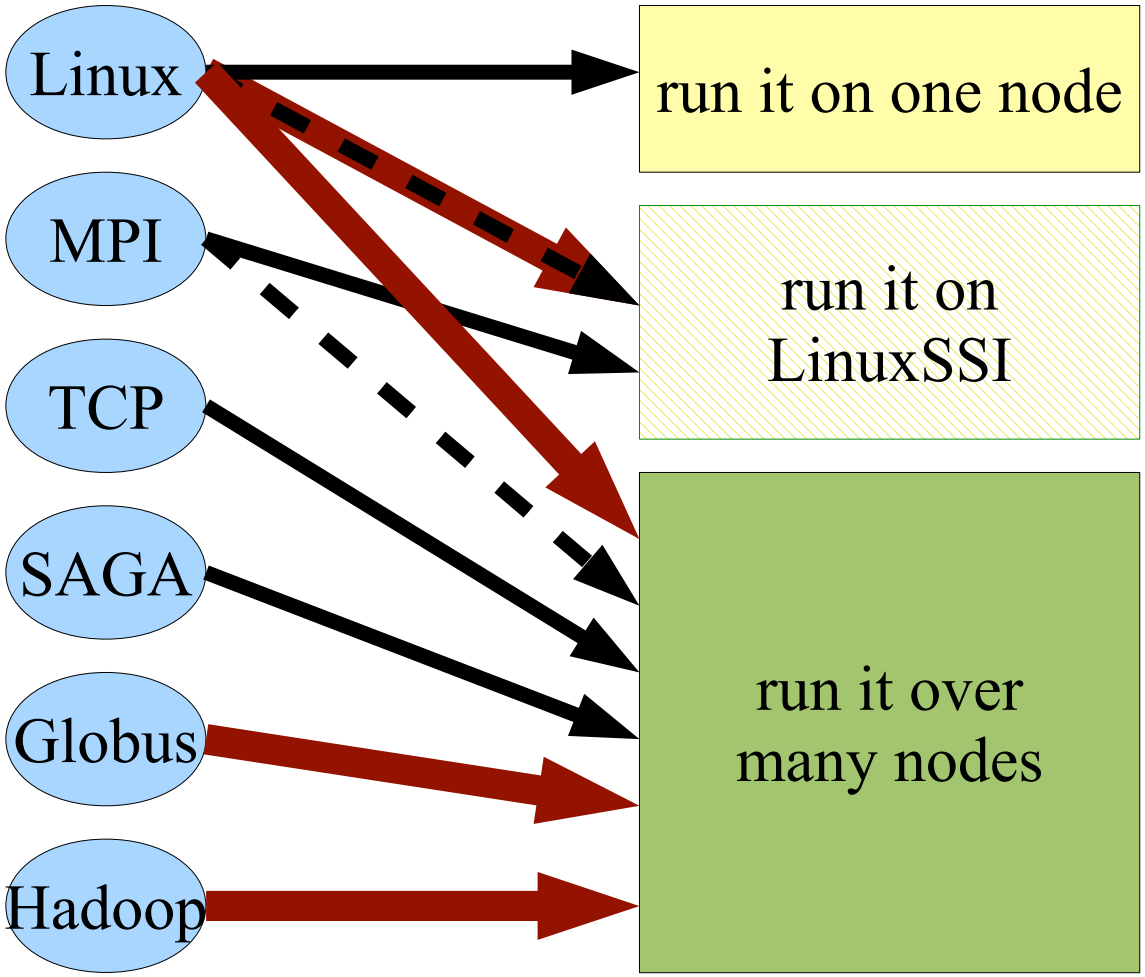
XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

What type of application do you have?

- **Ordinary Linux applications**
- **"Classical" parallel applications**
  - MPI
  - PVM
- **Grid applications**
  - Globus
  - Unicore
  - G-Lite
  - SAGA, JavaGAT...
- **Cloud applications**
  - Hadoop (Map/Reduce)

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

How do you want to run it?



Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

MPI applications

- **Built-in Support (experimental)**
  - XtreemOS job submission instead of rsh/ssh
    - resource selection, reservation etc
  - XtreemFS instead of NFS or manual file staging
- **How to run an MPI application**
  - copy it to your XtreemFS home
  - copy your personal XtreemOS configuration files and certificate to XtreemFS
  - write a skeleton JSDL containing resource requirements, e.g. x86_64 architecture
  - create a reservation:

    # xreservation ... -n <numProcs> -t <durationMin>
  - run it:

Information Society Technologies

XtreemOS IP project is funded by the European Commission under contract IST-FP6-033576

MPI - gotchas

- **Why do we say it is experimental?**
  - user certificate on XtreemFS
  - running it is still too involved
  - stdout/stderr are discarded
  - only mpich 1.2.7 is supported

- **General considerations**
  - firewall issues
  - grid network probably slower than LAN
  - no easy way to map logical network topology to physical network
  - too much processing power in your hands

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Linux and native Globus 4.0 applications

- **Your simple Globus 4.0 application (written following the tutorials)**
  - exposes parts as
    - grid services
    - command-line utilities
  - takes care of file staging (using GridFTP)
  - takes care of resource selection and scheduling
- **Your advanced Globus 4.0 application**
  - uses the plethora of different services with different interfaces to take care of certain aspects

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS-ifying Linux applications

1. **Divide it into command-line utilities**
2. **Copy application and data to your XtreemFS home**
3. **Master process**
   - bash script
     - xsub -f <job description>.jsdl
     - ssh-xos <NODE> <executable>
       - executes on the given node
   - SAGA application (C++, Java, Python)
   - native XtreemOS application
     - only if you need features not exposed by SAGA
     - still, not THAT complicated

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Hadoop (Map/Reduce)

- **"A new computing paradigm"**
  - Divide and conquer
  - Embarrassingly parallel
  - Lower ranks reporting to up the hierarchy

- **A really good framework to implement such applications**

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Hadoop vs. XtreemOS

- **Similarities**
  - Global, distributed file system (HDFS / XtreemFS)
  - No accounts required on computational nodes
  - Resource selection/scheduling

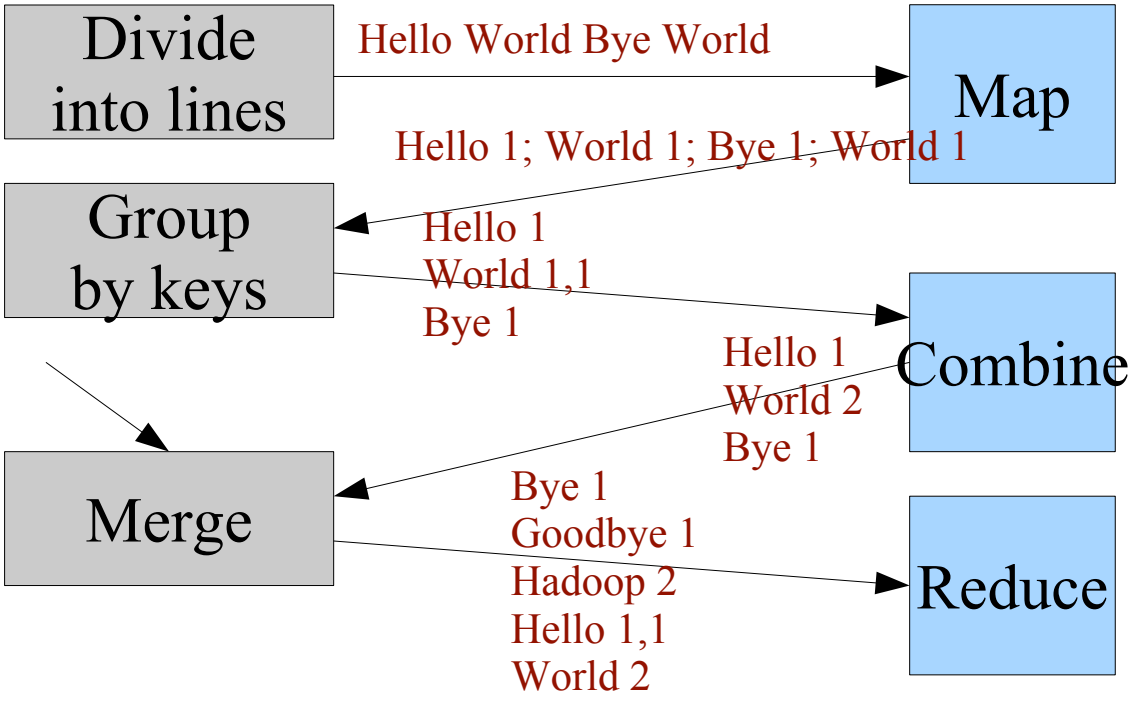
- **However**
  - XtreemOS only knows jobs and files and not:
    - How to distribute the work
    - Whether and how to restart failed jobs
  - Hadoop cannot do general parallel applications
    - Input, intermediate and final results are just independent key-value pairs

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Hadoop application

Hadoop job: count words

Hello World Bye World
Hello Hadoop Goodbye Hadoop



Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Hadoop Word Count

http://hadoop.apache.org/common/docs/current/mapred_tutorial.html

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Java Word Counter

- **Open file**
- **For each line:**
  - Split to words
  - Count words in a HashMap

- **DISCLAIMER: the source code in the next slides is only provided as illustration and thus deliberately kept compact; for one thing, it lacks any exception handling**

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Java Word Counter Source

```
import java.io.*;
import java.util.*;

public class WordCount {
	public static void main(String args[]) throws Exception
	{
		//open input file
		BufferedReader input = new BufferedReader(
				new InputStreamReader(
					new FileInputStream(new File(args[0]))));

		Map<String, Integer> map = new HashMap<String, Integer>();

		//split each line into words and count the latter
		String line;
		while (null != (line = input.readLine()))
		{
			String words[] = line.toLowerCase().split("[ _\\-()\\,\\.;'\\\"!:?]");
			for (String word : words)
			{
				if (map.containsKey(word))
					map.put(word, map.get(word)+1);
				else
					map.put(word, 1);
			}
		}

		//print results
		PrintWriter out = new PrintWriter(new BufferedWriter(
				new OutputStreamWriter(System.out)));
		for (String word : map.keySet())
			out.println(word + " " + map.get(word));
		out.close();
	}
}
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Multi-threaded WordCount

- **Divide work and start threads**
- **Each thread:**
  - Open file
  - Skip to part allocated to this thread
  - Count the words into thread's HashMap
- **When all threads finish:**
  - Reduce partial results into a single HashMap

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

Enabling Linux for the Grid

Multi-threaded WordCount Source

```
import java.io.*;
import java.util.*;

public class WordCountMultiCore {
	public static void main(String args[]) throws Exception
	{
		//parse parameters and check input file length
		File inputFile = new File(args[0]);
		int workerCount = Integer.parseInt(args[1]);
		long lastByte = inputFile.length()-1;

		//divide the work
		WordCountThread workers[] = new WordCountThread[workerCount];
		for (int i = 0; i < workerCount; i++)
		{
			workers[i] = new WordCountThread(inputFile,
				(i*lastByte)/workerCount, ((i+1)*lastByte)/workerCount);
			workers[i].start();
		}

		//wait for workers to finish the Map phase
		for (int i = 0; i < workerCount; i++)
			workers[i].join();

		//Reduce
		Map<String, Integer> map = workers[0].map;
		for (int i = 1; i < workerCount; i++)
			reduce(map, workers[i].map);

		//print results
		PrintWriter out = new PrintWriter(new BufferedWriter(
				new OutputStreamWriter(System.out)));

		for (String word : map.keySet())
			out.println(word + " " + map.get(word));

		out.close();
	}

	/*
	 * Joins two word count Maps
	 */
	private static void reduce(Map<String, Integer> cumulative,
			Map<String, Integer> single)
	{
		for (String word : single.keySet())
			map(cumulative, word, single.get(word));
	}

	/*
	 * Marks occurence(s) of the given word in the Map
	 */
	private static void map(Map<String, Integer> map, String word, int count)
	{
		if (map.containsKey(word))
			map.put(word, map.get(word)+count);
		else
			map.put(word, count);
	}

	private static class WordCountThread extends Thread
	{
		private final File inputFile;
		private final long startPos, endPos;
		public final Map<String, Integer> map;

		public WordCountThread(File inputFile, long startPos, long endPos)
		{
			this.inputFile = inputFile;
			this.startPos = startPos;
			this.endPos = endPos;
			this.map = new HashMap<String, Integer>();
		}

		public void run()
		{
			try
			{
				RandomAccessFile input = new RandomAccessFile(inputFile, "r");

				//skip to the first newline after startPos
				if (startPos > 0)
				{
					input.seek(startPos);
					while (input.read() != '\n')
					{}
				}

				//process up to the first newline after endPos
				//split each line into words and count the latter
				String line;
				while (input.getFilePointer() <= endPos
					&& (null != (line = input.readLine())))
				{
					String words[] = line.toLowerCase().split("[ _.\\-()\\,\\:;'\\\"!?:?]");
					for (String word : words)
						map(map, word, 1);
				}
			}
			catch (IOException e)
			{
				e.printStackTrace();
			}
		}
	}
}
```

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Information Society Technologies

SAGA WordCount

- **No shared memory → temporary files are used**
- **Divide work**
- **Start slave jobs using SAGA**
- **Each slave job:**
  - Process your part of file
  - Print results
- **When any slave finishes:**
  - Read its results file and add to the global HashMap

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

Enabling Linux for the Grid

SAGA WordCount Source master (left), worker (right)

```
import java.io.*;
import java.util.*;

import org.ogf.saga.task.state;
import org.ogf.saga.job.*;
import org.ogf.saga.session.*;
import org.ogf.saga.context.*;
import org.ogf.saga.url.*;

public class WordCountSaga {
    public static void main(String args[]) throws Exception
    {
        //parse parameters and check input file length
        File inputFile = new File(args[0]);
        int workerCount = Integer.parseInt(args[1]);
        long lastByte = inputFile.length()-1;

        //divide the work
        URL resourceManagerURL = URLFactory.createURL("ssh://localhost");
        JobService jobService = JobFactory.createJobService(
                resourceManagerURL);
        ArrayList<Job> jobs = new ArrayList<Job>();

        String output = "wordcount-worker-out";
        String exec = "wordcount-worker-wrk";

        for (int i = 0; i < workerCount; i++)
        {
            System.err.println("work for a worker: "
                    + args[0] + " "
                    + (i*lastByte/workerCount + " "
                    + ((i+1)*lastByte/workerCount);

            //create job description
            JobDescription jobDesc = JobFactory.createJobDescription();

            jobDesc.setAttribute(JobDescription.EXECUTABLE, "java");
            jobDesc.setVectorAttribute(JobDescription.ARGUMENTS,
                    {"WordCountWorker",
                     args[0] + " "
                     + (i*lastByte/workerCount + " "
                     + ((i+1)*lastByte/workerCount).split(" "));

            /*jobDesc.setAttribute(JobDescription.EXECUTABLE, "/bin/bash");
            jobDesc.setVectorAttribute(JobDescription.ARGUMENTS,
                    {"hello.sh 42").split(" ");}*/

            jobDesc.setAttribute(JobDescription.OUTPUT, output + i);
            jobDesc.setAttribute(JobDescription.ERROR, error + i);

            //create and run the job
            System.err.println("creating job...");
            Job job = jobService.createJob(jobDesc);
            System.err.println("...running job...");
            job.run();
            System.err.println("...OK");
            jobs.add(job);
        }

        //wait for workers to finish the Map phase
        //and do the Reduce
        Map<String, Integer> map = new HashMap<String, Integer>();

        while (jobs.size() > 0)
        {
            System.err.println("waiting for a job to finish...");
            Job doneJob = waitForAnyJob(jobs);
            System.err.println("...got one");
            jobs.remove(doneJob);
            reduce(map, doneJob.getJobDescription().getAttribute(
                    JobDescription.OUTPUT));
        }

        //print results
        PrintWriter out = new PrintWriter(new BufferedWriter(
                new OutputStreamWriter(System.out)));
        for (String word : map.keySet())
            out.println(word + " " + map.get(word));
        out.close();

        System.exit(0);
    }

    /*
     * Waits for any job to finish and returns it.
     * Unfortunately SAGA does not provide this.
     */
    private static Job waitForAnyJob(Iterable<Job> jobs)
        throws org.ogf.saga.error.SagaException
    {
        while (true)
        {
            for (Job job: jobs)
            {
                if (job.getState() != State.RUNNING
                        && job.getState() != State.SUSPENDED)
                    return job;
            }
            try
            {
                Thread.sleep(200);
            }
            catch (InterruptedException e)
            {}
        }
    }

    /*
     * Joins a result file to existing Map
     */
    private static void reduce(Map<String, Integer> cumulative,
            String resultsFilename) throws IOException
    {
        System.err.println("reducing " + resultsFilename + "...");

        //open input file
        BufferedReader input = new BufferedReader(
                new InputStreamReader(
                    new FileInputStream(resultsFilename)));

        //parse and reduce lines
        String line;
        while (null != (line = input.readLine()))
        {
            String tokens[] = line.split(" ");
            map(cumulative, tokens[0], Integer.parseInt(tokens[1]));
        }

        System.err.println("...done");
    }

    /*
     * Marks occurence(s) of the given word in the Map
     */
    private static void map(Map<String, Integer> map, String word, int count)
    {
        if (map.containsKey(word))
            map.put(word, map.get(word)+count);
        else
            map.put(word, count);
    }
}
```

```
import java.io.*;
import java.util.*;

public class WordCountPartial {
  public static void main(String args[]) throws Exception
  {
    //parse parameters
    File inputFile = new File(args[0]);
    long startPos = Long.parseLong(args[1]);
    long endPos = Long.parseLong(args[2]);

    //open file & skip to first newline after startPos
    RandomAccessFile input = new RandomAccessFile(inputFile, "r");
    if (startPos > 0)
    {
      input.seek(startPos);
      while (input.read() != '\n')
      {}
    }

    //process up to the first newline after endPos
    //split each line into words and count the latter
    Map<String, Integer> map = new HashMap<String, Integer>();
    String line;
    while (input.getFilePointer() <= endPos
      && (null != (line = input.readLine())))
    {
      String words[] = line.toLowerCase().split("[ _\\-()\\,\\.;'\\\"!:?]");
      for (String word : words)
      {
        if (map.containsKey(word))
          map.put(word, map.get(word)+1);
        else
          map.put(word, 1);
      }
    }

    //print results
    PrintWriter out = new PrintWriter(new BufferedWriter(
        new OutputStreamWriter(System.out)));
    for (String word : map.keySet())
      out.println(word + " " + map.get(word));
    out.close();
  }
}
```

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Information Society Technologies

XtreemOS

Enabling Linux for the Grid

Porting COMP Superscalar - hmmpfam from JavaGAT to XtreemOS AEM and SAGA

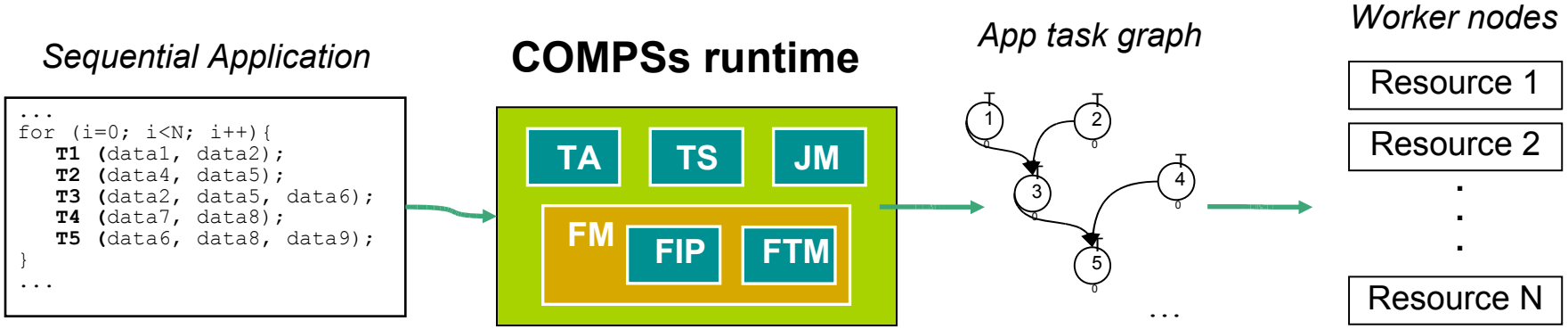
Courtesy of Enric Tejedor, BSC
enric.tejedor@bsc.es

XtreemOS-IP project
is funded by the European Commission under contract IST-FP6-033576

Information Society Technologies

COMP Superscalar (COMPSs)

- Framework to ease the development of Grid-unaware Java applications
- Simple programming model: Grid as transparent as possible
- Runtime that optimises the performance of the application (exploiting possible concurrency)



XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

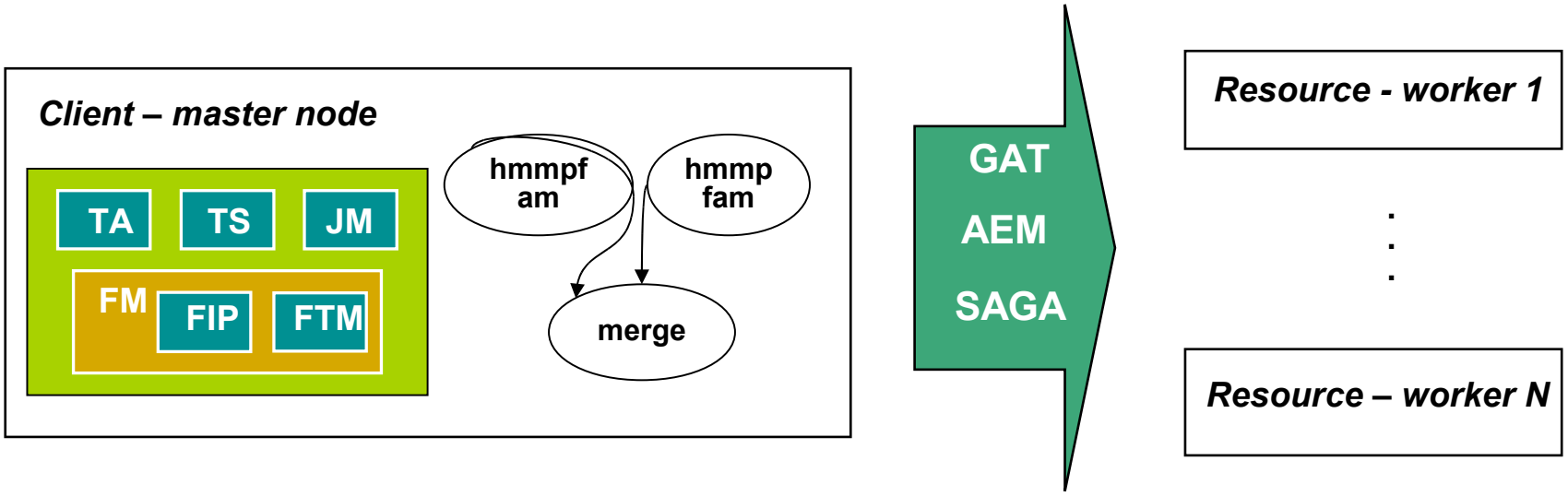
HMMER Java hmmpfam

- HMMER: analysis suite for protein sequence analysis
- hmmpfam
  - Compares sequences of aminoacids against models of protein families searching for significant sequence matches
  - Computationally intensive and embarassingly parallel
  - Executed in parallel by means of COMP Superscalar

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

Execution

- A client node runs the COMPSs runtime (master)
- Resource nodes run the tasks (workers)
- Two kinds of task: *hmmpfam* and *merge*
- GAT / AEM / SAGA used for job submission and monitoring



XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

Enabling Linux for the Grid

COMPSs - JavaGAT

Enric Tejedor, BSC
enric.tejedor@bsc.es

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

JavaGAT Resource Management

JavaGAT does not support resource reservation

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS
Enabling Linux for the Grid

JavaGAT Job Management – Job Submission

```
// Create context and resource broker
GATContext context = new GATContext();.
ResourceBroker broker = GAT.createResourceBroker(new GATContext(), new URI(destHost));

// Create Job description : software description + hardware description
SoftwareDescription sd = new SoftwareDescription();
sd.setExecutable(WORKER_SCRIPT);
String[] arguments = [...]  // arguments setting
sd.setArguments(arguments);

Map<String,Object> attributes = new HashMap<String,Object>();
attributes.put("machine.node", destHost);
ResourceDescription rd = new HardwareResourceDescription(attributes);

// Submit job (and register for callbacks)
Job job = broker.submitJob(new JobDescription(sd, rd), this, "job.status");
```

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

JavaGAT Job Management – Job Monitoring

```
// Method that processes callbacks from GAT – job state changes
public void processMetricEvent(MetricEvent value) {
        Job job = (Job)value.getSource();
        JobState newJobState = (JobState)value.getValue();

        if (newJobState == JobState.STOPPED) {
                // Job finished OK
                ...
        }
        else if (newJobState == JobState.SUBMISSION_ERROR) {
                // Job finished with error
                ...
        }
        ...
}
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

Enabling Linux for the Grid

COMPSs - AEM

Enric Tejedor, BSC
enric.tejedor@bsc.es

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

AEM Resource Management – Create reservation

```
// Read user cert
Security.addProvider(new BouncyCastleProvider());
X509Certificate caCert = Utils.readX509Certificate(cdaFilename, new char[0]);

// Prepare jsdl : set number of resources to reserve
String jsdl = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>" +
                "<JobDefinition xmlns:jsdl=\"http://schemas.ggf.org/jsdl/2005/11/jsdl\">" +
               "<JobDescription>" +
            "<Application>" +
        "<POSIXApplication xmlns:ns1=\"http://schemas.ggf.org/jsdl/2005/11/jsdl-posix\">" +
                 "<Executable>dummy</Executable>" +
          "</POSIXApplication>" +
            "</Application>" +
           "<Resources>" +
          "<TotalResourceCount>" +
               "<exact>" + numResources + "</exact>" +
            "</TotalResourceCount>" +
            "</Resources>" +
           "</JobDescription>" +
          "</JobDefinition>";

// Discover resources
ArrayList<CommunicationAddress> nodes = XResMng.getResources(jsdl, caCert, null);
```

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

AEM Resource Management – Create reservation (2)

```
// Set a reservation request for each resource
ArrayList <ReservationRequest> aRR = new ArrayList <ReservationRequest>();
GregorianCalendar startTime = new GregorianCalendar();
GregorianCalendar endTime = (GregorianCalendar)startTime.clone();
endTime.add(GregorianCalendar.MINUTE, resMinutes);
for (CommunicationAddress ca : nodes) {
        ReservationRequest request = new ReservationRequest();
        request.nodeAddress = ca;
        request.localRequest = new Request();

        TTElm ttelm = TTElmFactory.createBasic(startTime, endTime, SharingValues.MUTUAL);
        TTElmFactory.addOwnerInfo(ttelm, new OwnersInfo("COMPSs", "COMPSs"));
        TTElmFactory.addAttribute(ttelm, new CurrentAmount(1));
        TTElmRequest elmreq = new TTElmRequestAdd("CPU0", ttelm);
        request.localRequest.add(elmreq);

        aRR.add(request);
}

// Create the reservation
String reservationId = XReservationManager.createReservationExplicit(aRR , caCert);
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

AEM Resource Management – Release reservation

```
// Release reservation
XReservationManager.releaseReservation(reservationId, caCert);
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

AEM Job Management – Job Submission

```
// Create JSDL for the job: executable + arguments
JsdlObject jodl = new JsdlObject(null, null, WORKER_SCRIPT);
jodl.addArgument(...);
...
jodl.addArgument(...);

// Create job
String xosJobId = XJobMng.createJob(jodl.toString(), false, reservationId, caCert);

// Submit job
XJobMng.runJobRes(xosJobId, reservationId, SingleCommAddress.getFromHostPort("/" +
destHost + ":60000"), caCert);
```

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

AEM Job Management – Job Monitoring

```
// Poll for the state of jobs
while (!endOfApplication) {
        ...
        String jInfo = XjobMng.getJobsInfo(xosJobs,  // List<String> containing ids of submitted jobs
                                                TypeOfInfo.BASIC.val() |TypeOfInfo.NO_BUFFER.val(),
                                                InfoLevel.JOB.val() ,
                                                null,
                                                caCert);

        jil = new JobInfoList (jInfo);
        for (String xosJob : jil.getJobs()) {
                String status = jil.getMetricValue(xosJob,"jobStatus").getValue();
                if (status.equals("Done")) {
                        // Job finished OK
                }
                else if (status.equals("Failed")) {
                        // Job finished with error
                }
        }

        Thread.sleep(POLLING_PERIOD);
}
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

XtreemOS

Enabling Linux for the Grid

COMPSs - SAGA

Enric Tejedor, BSC
enric.tejedor@bsc.es

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

SAGA Resource Management – Create reservation

```
// Create Resource Service
URL rm = URLFactory.createURL("xos://localhost:60000");
ResourceService rs = ResourceFactory.createResourceService(rm);

// Discover resources
ResourceDescription query = ResourceFactory.createResourceDescription();
List<String> ids = rs.discover(query);
List<String> ids_reservation = [...] // take here numResources resources from the ids list

// Reserve resources
GregorianCalendar startTime = new GregorianCalendar();
Date start = startTime.getTime();
Date end = new Date(start.getTime() + resMinutes);
int startSec = (int)(start.getTime() / 1000);
int endSec = (int)(end.getTime() / 1000);
Reservation reservation = rs.reserve(ids_reservation, startSec, endSec);
String reservationId = reservation.getAttribute(Reservation.RESERVATIONID);
```

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

SAGA Resource Management – Release reservation

```
// Release reservation
rs.cancel(rs.getReservation(reservationId), TIME_TO_WAIT);
```

Information Society Technologies

XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

SAGA Job Management – Job Submission

```
// Create Job Service
URL u = URLFactory.createURL("xos://localhost:60000");
JobService js = JobFactory.createJobService(u);
XOJobService xojs = (XOJobService)js;

...

// Create Application Description: executable + arguments
ApplicationDescription ad = ResourceFactory.createApplicationDescription();
ad.setAttribute(ApplicationDescription.EXECUTABLE, WORKER_SCRIPT);
String[] arguments = [...]  // arguments setting
ad.setVectorAttribute(JobDescription.ARGUMENTS, arguments);

// Create job
Job job = xojs.createJob(ad, reservationId);

// Submit job
job.run();
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576

SAGA Job Management – Job Monitoring

```
// Poll for the state of jobs
while (!endOfApplication) {
        ...
        List<Job> jobs = [...] // List containing submitted jobs

        for (Job job : jobs) {
                switch (xosJob.getState()) {
                        case DONE:
                                // Job finished OK
                        case FAILED:
                                // Job finished with error
                }
        }

        Thread.sleep(POLLING_PERIOD);
}
```


XtreemOS IP project
is funded by the European Commission under contract IST-FP6-033576