



GAT Tutorial

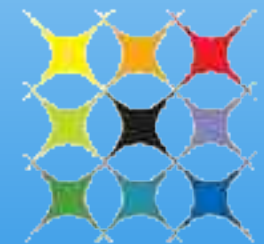


Getting started with the Grid Application Toolkit

Rob van Nieuwpoort

rob@cs.vu.nl

Vrije Universiteit
Amsterdam



vl·e



vrije Universiteit



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- Security
- Grid I/O
- Resource Management
- Application Information Management
- Monitoring

- Hands-on session



The GridLab Project



- EU Project Funded by 5th Framework
- 3 years, 5 million euros
- Partners
 - PSNC, AEI, ZIB, MASARYK, SZTAKI, ISUFI, Cardiff, NTUA, Chicago, ISI, Wisconsin, Sun, Compaq,...
 - Vrije Universiteit Amsterdam
- 12 Work Packages covering
 - Grid Portals, Mobile Users, Data Management, Applications, Testbed, ...
 - GAT: The Grid Application Toolkit



Writing Grid Applications



Information Society
Technologies

Grid Application



Writing Grid Applications



Information Society
Technologies

`File.copy(...)`



Grid Application

`submitJob(...)`





Writing Grid Applications



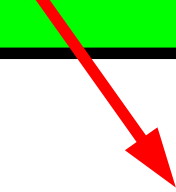
Information Society
Technologies



File.copy(...)

Grid Application

submitJob(...)



cp





Writing Grid Applications



Information Society
Technologies



File.copy(...)

Grid Application

submitJob(...)

cp
ftp



Writing Grid Applications



Information Society
Technologies

`File.copy(...)`

Grid Application

`submitJob(...)`

`cp`
`ftp`
`gridftp`



Writing Grid Applications



Information Society
Technologies

`File.copy(...)`

Grid Application

`submitJob(...)`



`cp`
`ftp`
`gridftp`
`scp`



Writing Grid Applications



Information Society
Technologies

`File.copy(...)`

Grid Application

`submitJob(...)`



`cp`
`ftp`
`gridftp`
`scp`
`http`



Writing Grid Applications



Information Society
Technologies



`File.copy(...)`

Grid Application

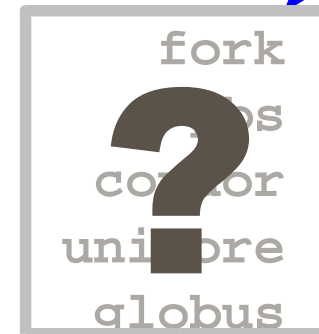
`submitJob(...)`

`cp`
`ftp`
`gridftp`
`scp`
`http`

`fork`
`pbs`
`condor`
`unicore`
`globus`



```
cp  
ftp  
gridftp  
scp  
http
```



```
fork  
globus  
com  
uni  
pre
```

- Which should you use?



```
cp  
ftp  
gridftp  
scp  
http
```

```
fork  
globus  
condor  
unipre  
globus
```

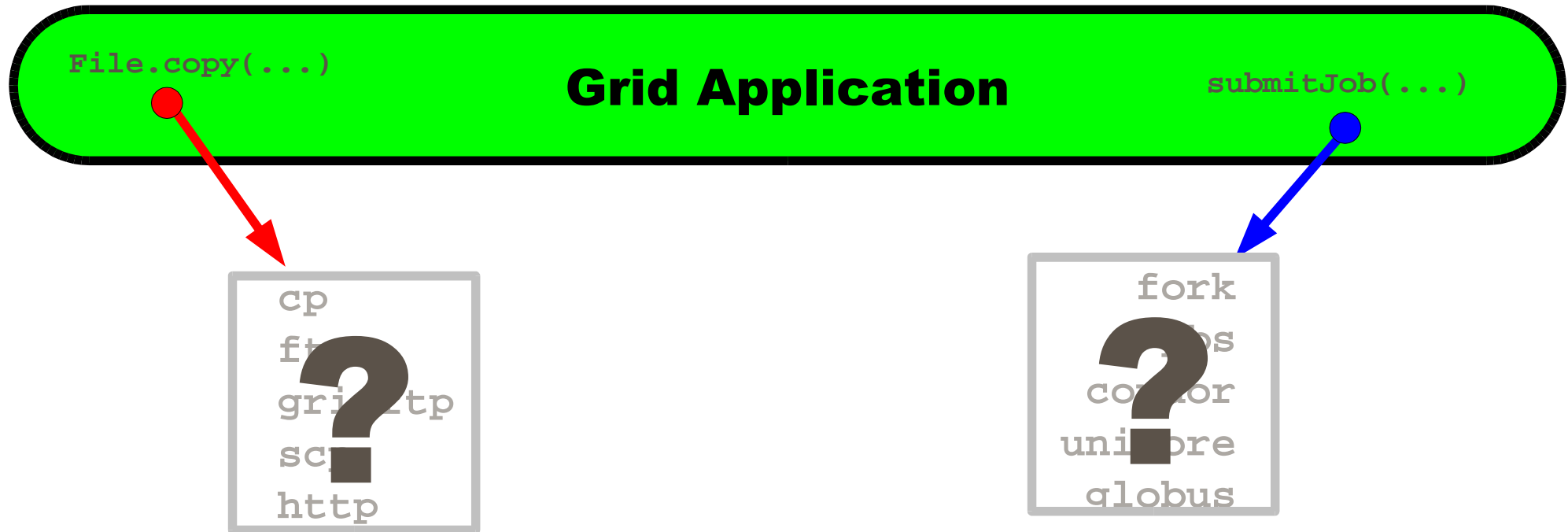
- Which should you use?
- Some might not be available on all sites



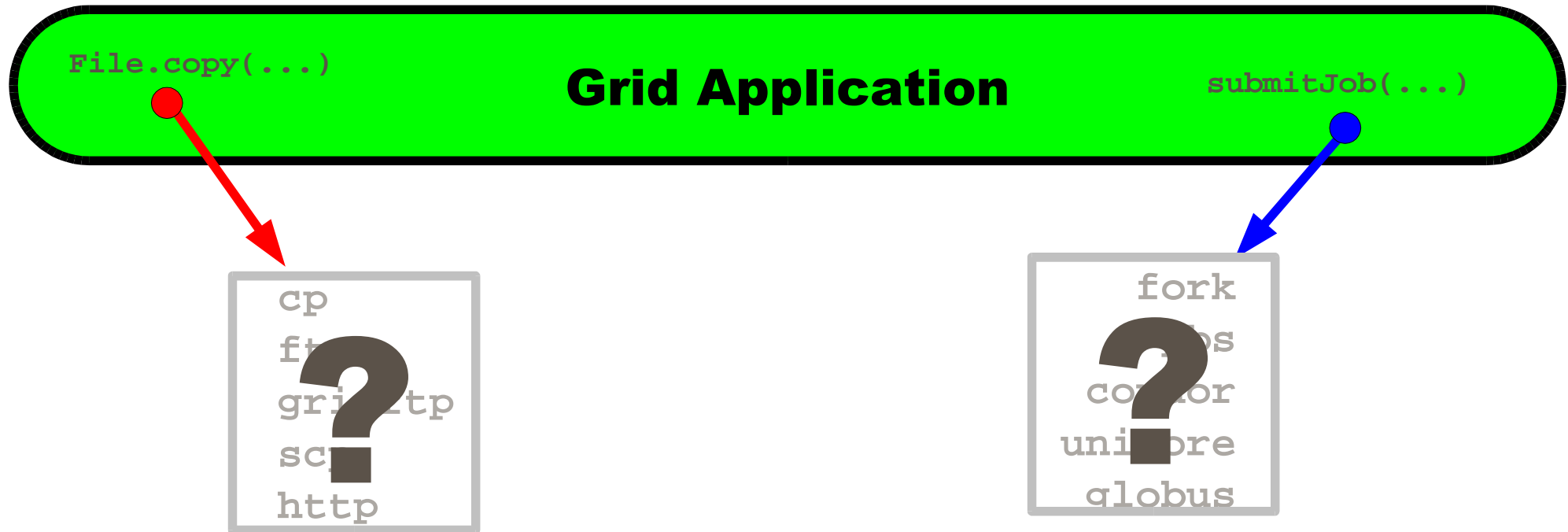
```
cp  
ft  
gridftp  
scp  
http
```

```
fork  
globus  
condor  
unipre  
globus
```

- Which should you use?
- Some might not be available on all sites
- Some may not work for all users (certificates)



- Which should you use?
- Some might not be available on all sites
- Some may not work for all users (certificates)
- Version differences (Globus changes every 2 months)



- Which should you use?
- Some might not be available on all sites
- Some may not work for all users (certificates)
- Version differences (Globus changes every 2 months)
- Exponential number of combinations!



```
cp  
ft  
grid-tp  
sc  
http
```

```
fork  
os  
color  
unipre  
globus
```

- Which should you use?
- Some might not be available on all sites
- Some may not work for all users (e.g. ftp)
- Versions differ (Globus changes every 2 months)
- Exponential number of combinations!

Not portable!



Globus File copy (C++)



```
int RemoteFile::GetFile (char const* source, char const* target) {
    globus_url_t          source_url;
    globus_io_handle_t    dest_io_handle;
    globus_ftp_client_operationattr_t source_ftp_attr;
    globus_result_t       result;
    globus_gass_transfer_requestattr_t source_gass_attr;
    globus_gass_copy_attr_t source_gass_copy_attr;
    globus_gass_copy_handle_t gass_copy_handle;
    globus_gass_copy_handleattr_t gass_copy_handleattr;
    globus_ftp_client_handleattr_t ftp_handleattr;
    globus_io_attr_t        io_attr;
    int                     output_file = -1;

    if ( globus_url_parse (source_URL, &source_url) != GLOBUS_SUCCESS ) {
        printf ("can not parse source_URL \"%s\"\n", source_URL);
        return (-1);
    }

    if ( source_url.scheme_type != GLOBUS_URL_SCHEME_GSIFTP &&
        source_url.scheme_type != GLOBUS_URL_SCHEME_FTP &&
        source_url.scheme_type != GLOBUS_URL_SCHEME_HTTP &&
        source_url.scheme_type != GLOBUS_URL_SCHEME_HTTPS ) {
        printf ("can not copy from %s - wrong prot\n", source_URL);
        return (-1);
    }

    globus_gass_copy_handleattr_init (&gass_copy_handleattr);
    globus_gass_copy_attr_init (&source_gass_copy_attr);

    globus_ftp_client_handleattr_init (&ftp_handleattr);
    globus_io_fileattr_init (&io_attr);

    globus_gass_copy_attr_set_io (&source_gass_copy_attr, &io_attr);

    globus_gass_copy_handleattr_set_ftp_attr (&gass_copy_handleattr,
        &ftp_handleattr);

    globus_gass_copy_handle_init (&gass_copy_handle,
        &gass_copy_handleattr);
```

```
    if (source_url.scheme_type == GLOBUS_URL_SCHEME_GSIFTP ||
        source_url.scheme_type == GLOBUS_URL_SCHEME_FTP ) {
        globus_ftp_client_operationattr_init (&source_ftp_attr);
        globus_gass_copy_attr_set_ftp (&source_gass_copy_attr,
            &source_ftp_attr);
    }
    else {
        globus_gass_transfer_requestattr_init (&source_gass_attr,
            source_url.scheme);
        globus_gass_copy_attr_set_gass (&source_gass_copy_attr,
            &source_gass_attr);
    }

    output_file = globus_libc_open ((char*) target,
        O_WRONLY | O_TRUNC | O_CREAT,
        S_IRUSR | S_IWUSR | S_IRGRP |
        S_IWGRP);

    if ( output_file == -1 ) {
        printf ("could not open the file \"%s\"\n", target);
        return (-1);
    }
    /* convert stdout to be a globus_io_handle */
    if ( globus_io_file_posix_convert (output_file, 0,
        &dest_io_handle)
        != GLOBUS_SUCCESS) {
        printf ("Error converting the file handle\n");
        return (-1);
    }

    result = globus_gass_copy_register_url_to_handle (
        &gass_copy_handle, (char*)source_URL,
        &source_gass_copy_attr, &dest_io_handle,
        my_callback, NULL);

    if ( result != GLOBUS_SUCCESS ) {
        printf ("error: %s\n", globus_object_printable_to_string
            (globus_error_get (result)));
        return (-1);
    }
    globus_url_destroy (&source_url);
    return (0);
}
```



CoG/RFT File copy (C++)



Information Society
Technologies

```
package org.globus.ogsa.gui;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.net.URL;
import java.util.Date;
import java.util.Vector;
import javax.xml.rpc.Stub;
import org.apache.axis.message.MessageElement;
import org.apache.axis.utils.XMLUtils;
import org.globus.*
import org.gridforum.ogsi.*
import org.gridforum.ogsi.holders.TerminationTimeTypeHolder;
import org.w3c.dom.Document;
import org.w3c.dom.Element;

public class RFTClient {
    public static void copy (String source_url, String target_url) {
        try {
            File requestFile = new File (source_url);
            BufferedReader reader = null;
            try {
                reader = new BufferedReader (new FileReader (requestFile));
            } catch (java.io.FileNotFoundException fnfe) { }
            Vector requestData = new Vector ();
            requestData.add (target_url);
            TransferType[] transfers1 = new TransferType[transferCount];
            RFTOptionsType multirftOptions = new RFTOptionsType ();

            multirftOptions.setBinary (Boolean.valueOf (
                (String)requestData.elementAt (0)).booleanValue ());
            multirftOptions.setBlockSize (Integer.valueOf (
                (String)requestData.elementAt (1)).intValue ());
            multirftOptions.setTcpBufferSize (Integer.valueOf (
                (String)requestData.elementAt (2)).intValue ());
            multirftOptions.setNotpt (Boolean.valueOf (
                (String)requestData.elementAt (3)).booleanValue ());
            multirftOptions.setParallelStreams (Integer.valueOf (
                (String)requestData.elementAt (4)).intValue ());
            multirftOptions.setDcau(Boolean.valueOf(
                (String)requestData.elementAt (5)).booleanValue ());

            int i = 7;
            for (int j = 0; j < transfers1.length; j++)
            {
                transfers1[j] = new TransferType ();

                transfers1[j].setTransferId (j);
                transfers1[j].setSourceUrl ((String)requestData.elementAt (i++));
                transfers1[j].setDestinationUrl ((String)requestData.elementAt (i++));
                transfers1[j].setRftOptions (multirftOptions);
            }
        }
    }
}
```

```
TransferRequestType transferRequest = new TransferRequestType ();
transferRequest.setTransferArray (transfers1);

int concurrency = Integer.valueOf
    ((String)requestData.elementAt(6)).intValue();

if (concurrency > transfers1.length)
{
    System.out.println ("Concurrency should be less than the number"
        "of transfers in the request");
    System.exit (0);
}
transferRequest.setConcurrency (concurrency);

TransferRequestElement requestElement = new TransferRequestElement ();
requestElement.setTransferRequest (transferRequest);

ExtensibilityType extension = new ExtensibilityType ();
extension = AnyHelper.getExtensibility (requestElement);

OGSIServiceGridLocator factoryService = new OGSIServiceGridLocator ();
Factory factory = factoryService.getFactoryPort (new URL (source_url));
GridServiceFactory gridFactory = new GridServiceFactory (factory);

LocatorType locator = gridFactory.createService (extension);
System.out.println ("Created an instance of Multi-RFT");

MultiFileRFTDefinitionServiceGridLocator loc
    = new MultiFileRFTDefinitionServiceGridLocator();
RFTPortType rftPort = loc.getMultiFileRFTDefinitionPort (locator);
((Stub)rftPort)._setProperty (Constants.AUTHORIZATION,
    NoAuthorization.getInstance());
((Stub)rftPort)._setProperty (GSIConstants.GSI_MODE,
    GSIConstants.GSI_MODE_FULL_DELEG);
((Stub)rftPort)._setProperty (Constants.GSI_SEC_CONV,
    Constants.SIGNATURE);
((Stub)rftPort)._setProperty (Constants.GRIM_POLICY_HANDLER,
    new IgnoreProxyPolicyHandler ());

int requestid = rftPort.start ();
System.out.println ("Request id: " + requestid);

}
catch (Exception e)
{
    System.err.println (MessageUtils.toString (e));
}
}
```



Why GAT?



- The situation today:
 - Grids: everywhere
 - Grid applications: nowhere
- Why is this?
 - Application programmers accept the Grid as a computing paradigm only very slowly.
- Problems:
 - Interfaces are NOT simple
 - Portability / interoperability
 - Different and evolving interfaces to the 'Grid'
 - Versions, new services, new implementations, WSDL, web services do not solve all problems at all
 - Environment changes in many ways



What is GAT?



- GAT: Grid Application Toolkit
 - API and Toolkit for developing and running portable grid applications independently of the underlying grid infrastructure and available services
- GAT is used by applications to access grid services
- **Simple** API
- GAT Adaptors (“plugins”)
 - Connect GAT to grid services
 - Allow for multiple providers (Globus, Unicore, ProActive, ...)
- GAT Engine
 - Provides runtime delegation of GAT-API calls to adaptors



GAT Philosophy (1)



Information Society
Technologies

- GAT does not aim to replace existing “grid infrastructure.”
- GAT aims to provide a simple, clear interface to many different infrastructures
 - GRAM
 - Condor
 - Unicore
 - GridFTP
 - RFT
 - ...
- Open source, BSD-like licence



GAT Philosophy (2)



- Applications make GAT-API calls for grid operations
 - Applications link against GAT
- Applications run irrespective of available infrastructure
 - GAT Engine loads all available adaptors **at runtime**
 - Upon a call to the GAT-API the GAT Engine determines which adaptor(s) provide the “grid operation”
 - Upon “grid operation” failure another adaptor may be called
- There exist a set of default adaptors which provide default local capabilities
 - Grid applications can thus be compiled, linked, and tested without any available grid services
 - The same application executable can run in a “full grid environment.” **No recompilation / linking**



The GAT API



- A language independent, object oriented specification
- Language bindings
 - C, C++, Java, python, ...
- Work in progress
 - perl, .net, fortran
- Adopt the “look-and-feel” of the target language
- focus on well-known programming paradigms
 - for a file provide a file API
 - Not (web)services to services to files. . .
 - expect open, close, read, write, seek. Do not introduce fancy things like the need to ask a service discovery service to tell me the location of an service which is able to tell me the location of my file...



GAT API features



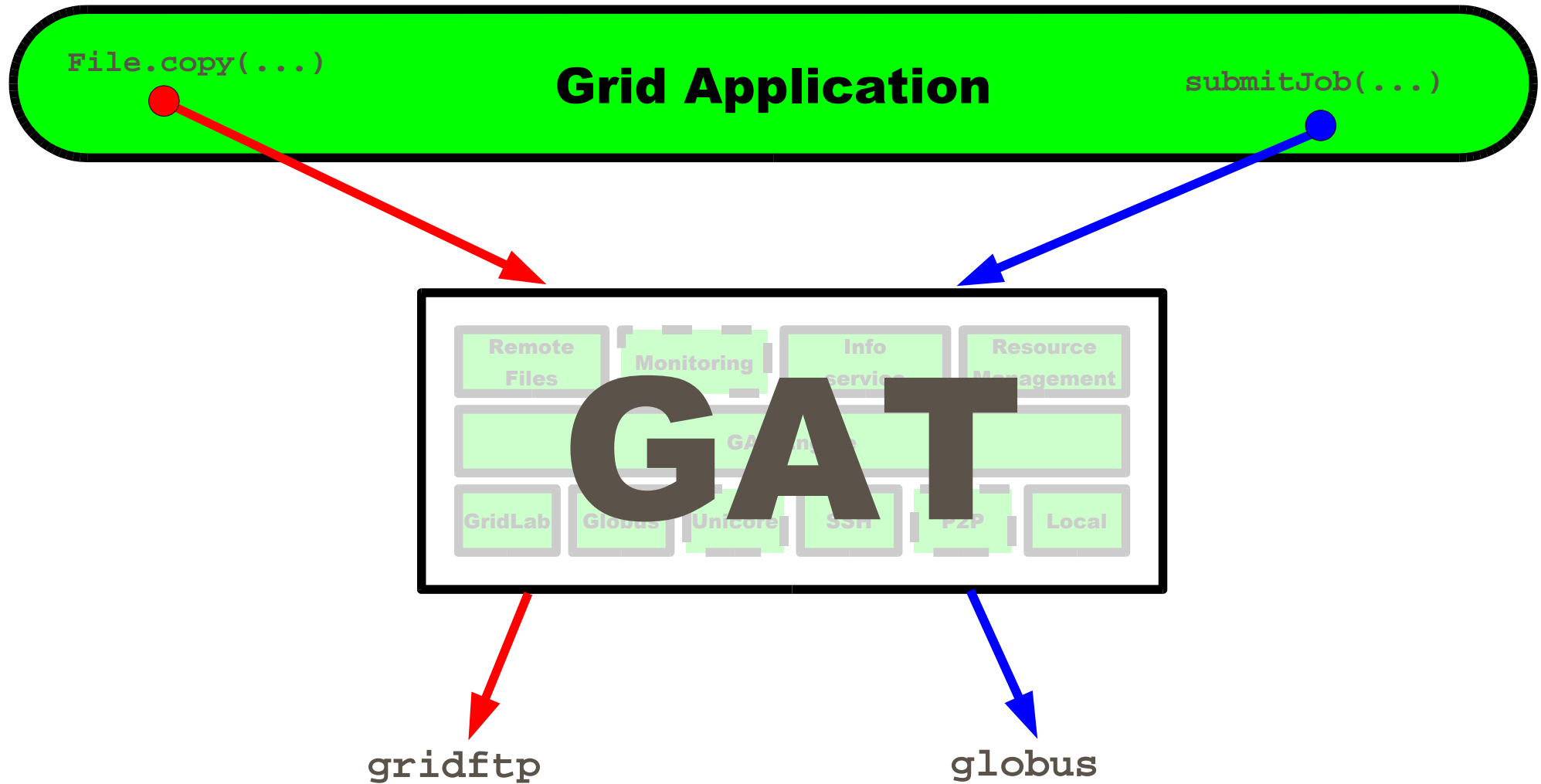
- Security (deal with passwords, credentials, etc)
- Grid I/O
 - File operations, remote file access, file replication
 - Inter-process communication
- Resource Management
 - Resource brokering
 - Forking grid applications, job management
- Application Information Management
 - Global repository for application specific information
 - Query this information repository
- Monitoring
 - Grid monitoring
 - Application monitoring and steering



Grid Applications with GAT



Information Society
Technologies





File Copy with GAT (C++)



Information Society
Technologies

```
#include <GAT++.hpp>

void RemoteFile::GetFile (
    GAT::Context context, std::string source_url,
    std::string target_url) throws GAT::Exception {
    GAT::File file (context, source_url);
    file.Copy      (target_url);
}
```



File Copy with GAT (C++)



```
#include <GAT++.hpp>

void RemoteFile::GetFile (
    GAT::Context context, std::string source_url,
    std::string target_url) throws GAT::Exception {
    GAT::File file (context, source_url);
    file.Copy      (target_url);
}
```



Overview



- What is GAT and why do we need it?
- **JavaGAT overview and structure**
- Security
- Grid I/O
- Resource Management
- Application Information Management
- Monitoring

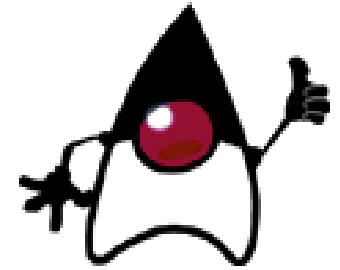
- Hands-on session



Why Java?



- Java is widely used, object oriented.
- Secure (language, sandbox)
- Java is “write once, run everywhere”.
 - Compile application on your desktop machine.
 - This creates machine independent bytecode.
 - Copy application files and the GAT to any grid site (portal typically does this for you).
 - Just run it. No recompilation / configuration.
- Performance of current JITs is good.
 - Compiled (just-in-time)
 - Runtime, profile-driven optimizations
 - Applications are typically 10% slower than C.
- Ideal for grid computing?





GAT API



- GAT API is an object oriented specification, but language independent.
- Tension between existing language features and the GAT specification.
 - Programmers do not want to learn new APIs for features already in the language.
 - GAT should “feel” the same across platforms.
- Find the tight trade-off



Java GAT Structure



Grid Application

Files

Monitoring

**Info
service**

**Resource
Management**

GAT Engine

GridLab

Globus

Unicore

SSH

P2P

Local

Legend:

Java

Done

W.I.P



Java GAT Structure



Grid Application

API

Files

Monitoring

Info
service

Resource
Management

GAT Engine

GridLab

Globus

Unicore

SSH

P2P

Local

Legend:

Java

Done

W.I.P



Java GAT Structure



Grid Application

API

Files

Monitoring

Info
service

Resource
Management

GAT Engine

ADAPT.

GridLab

Globus

Unicore

SSH

P2P

Local

Legend:

Java

Done

W.I.P



Java GAT Structure



Information Society
Technologies

`File.copy(...)`



Grid Application

API

Files

Monitoring

Info
service

Resource
Management

GAT Engine

ADAPT.

GridLab

Globus

Unicore

SSH

P2P

Local

Legend:

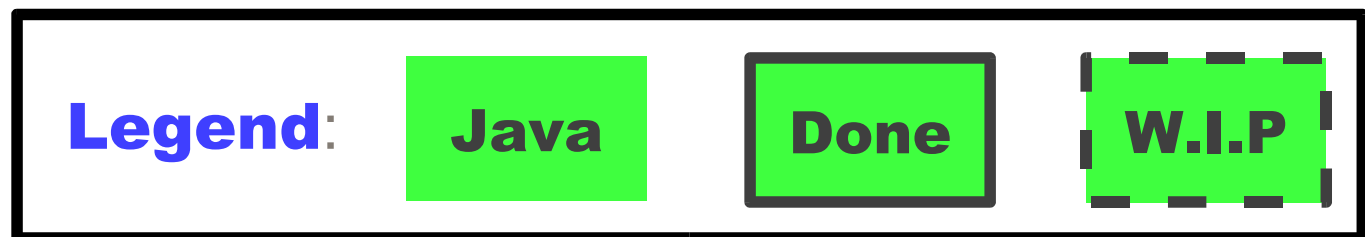
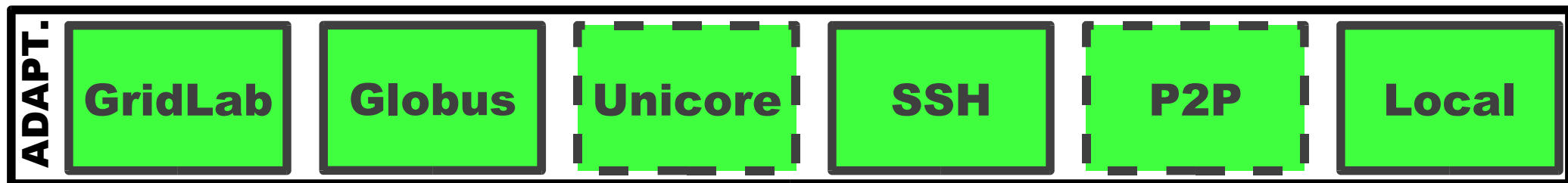
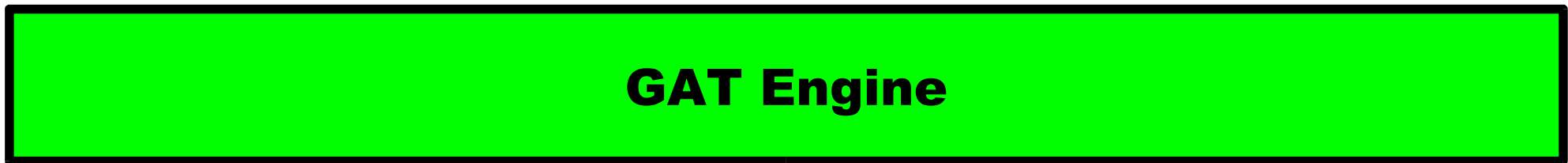
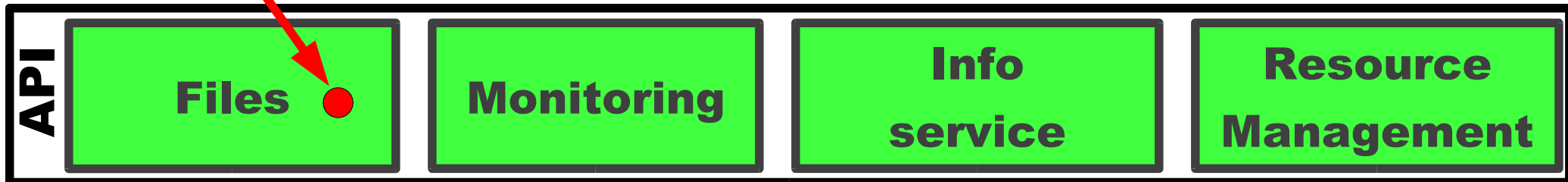
Java

Done

W.I.P

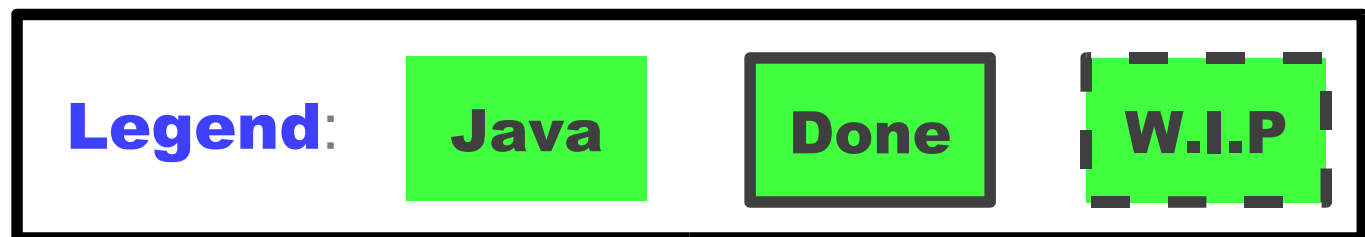
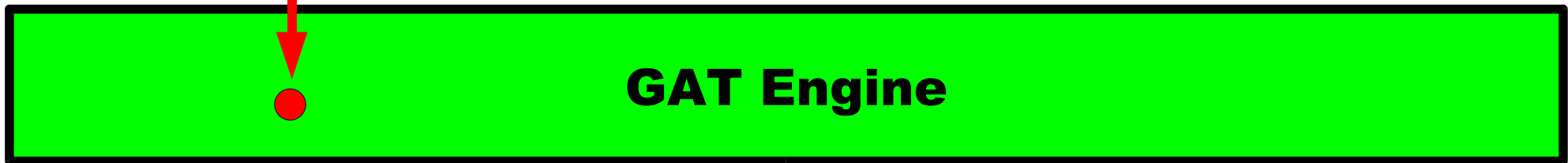
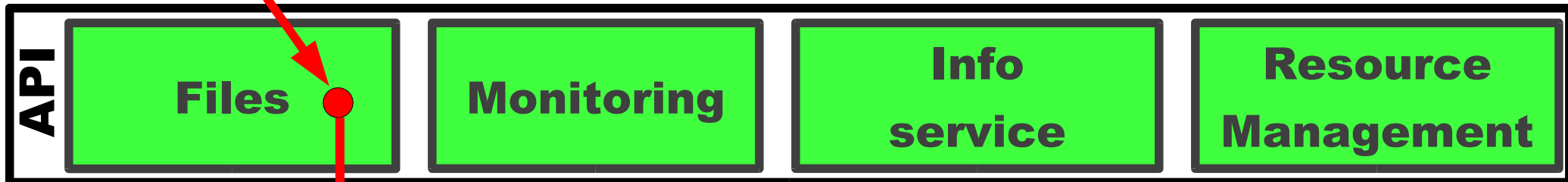


Java GAT Structure



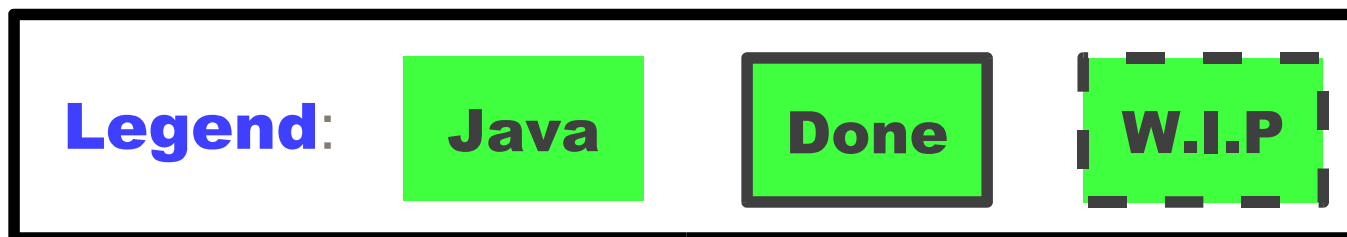
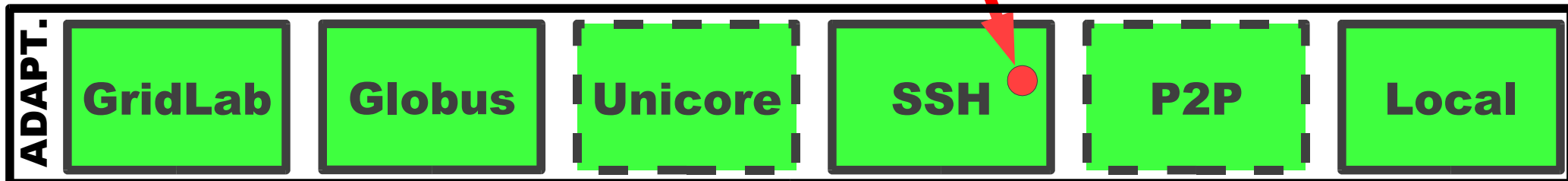
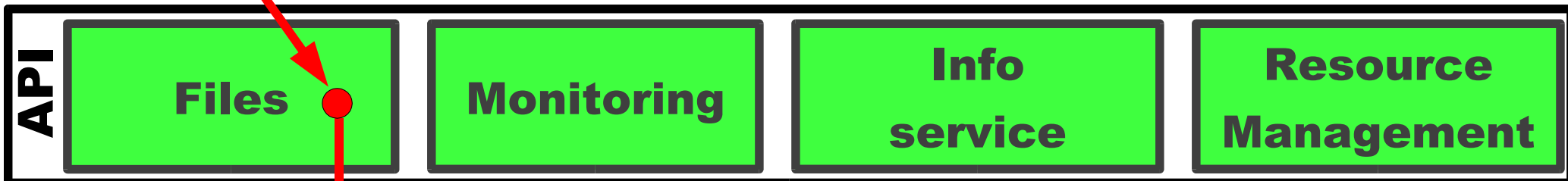


Java GAT Structure



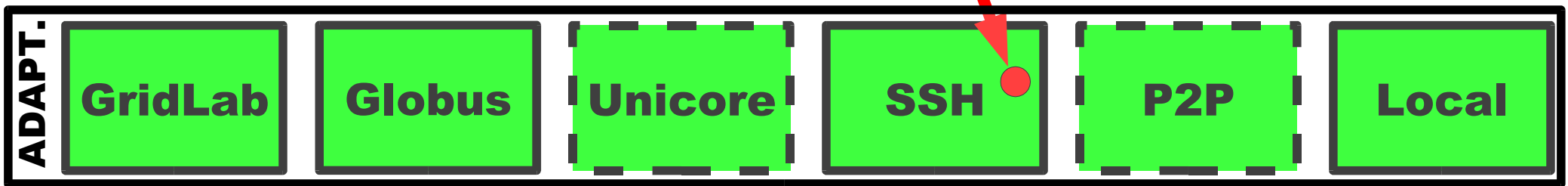
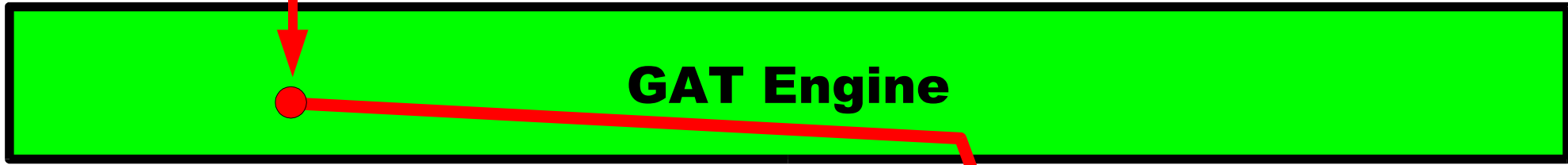
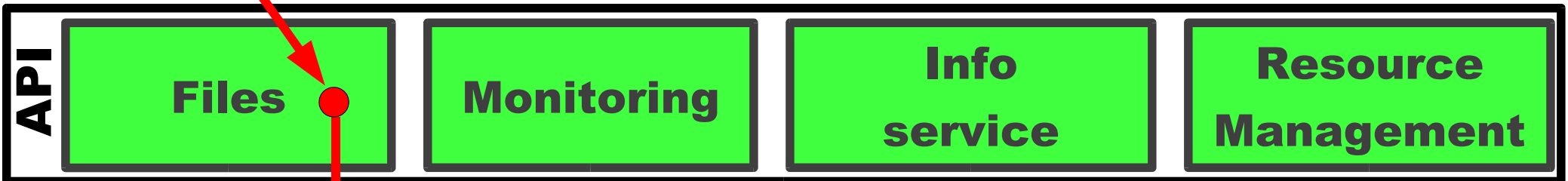


Java GAT Structure



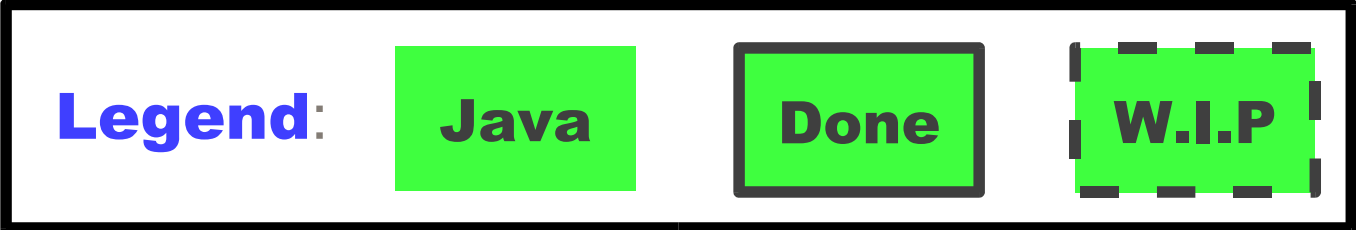
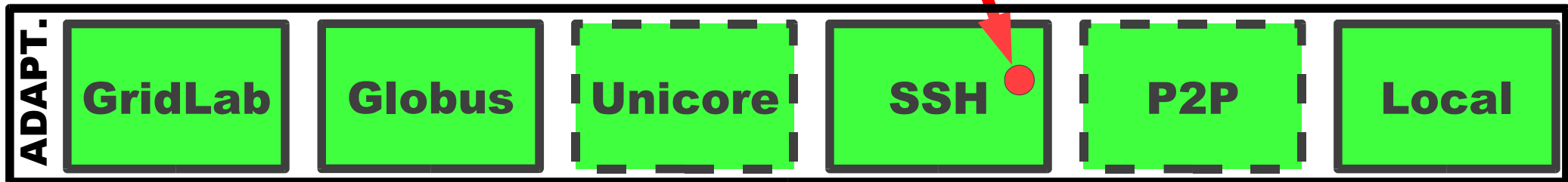
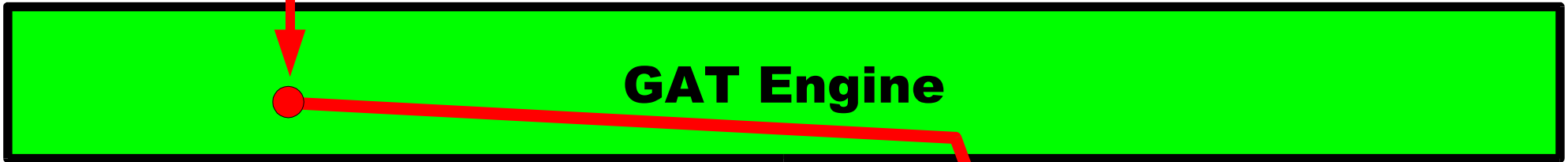


Java GAT Structure



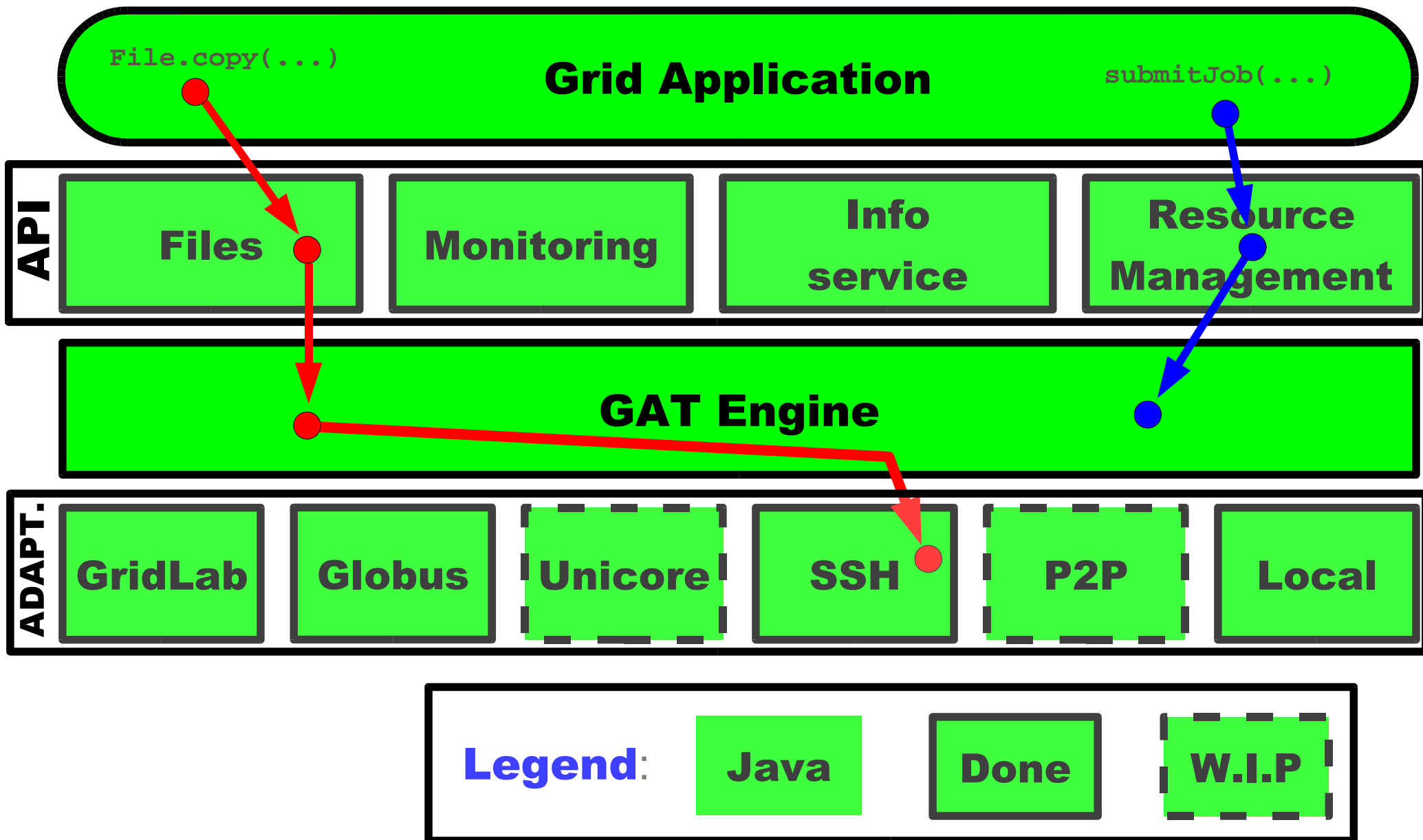


Java GAT Structure



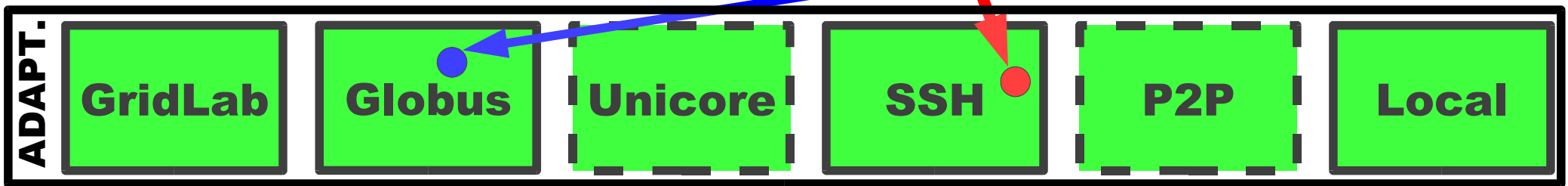
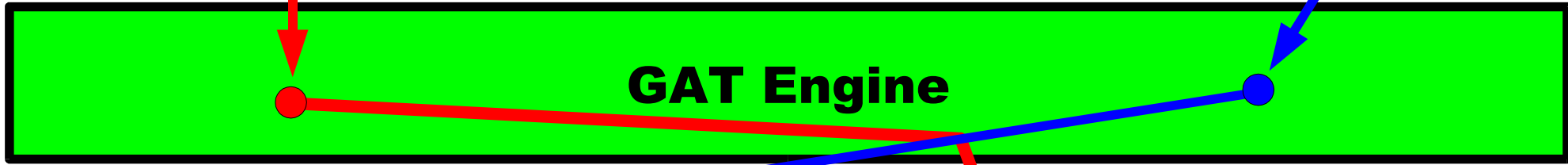
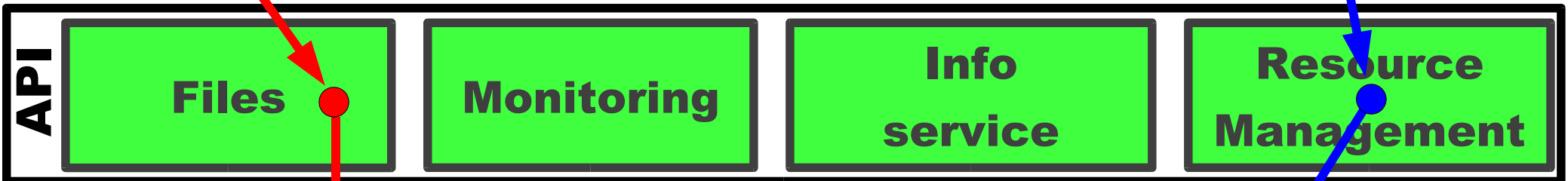


Java GAT Structure



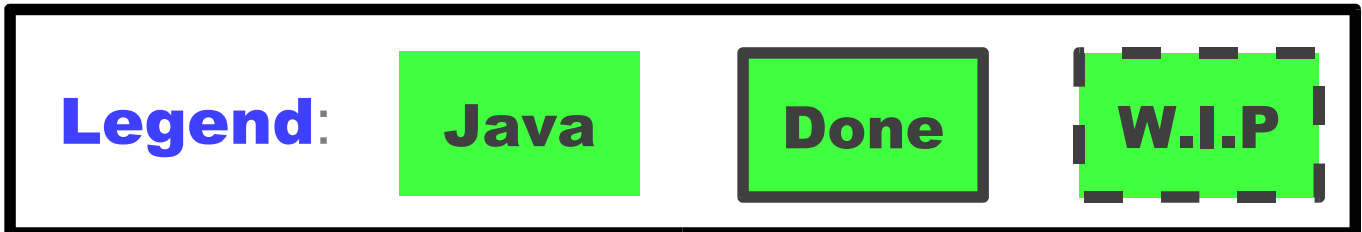
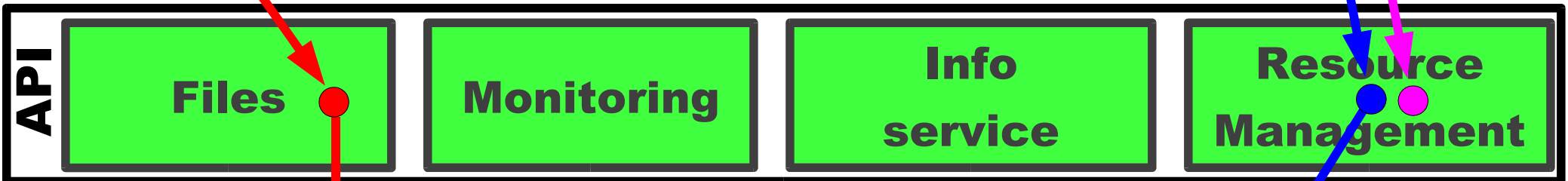


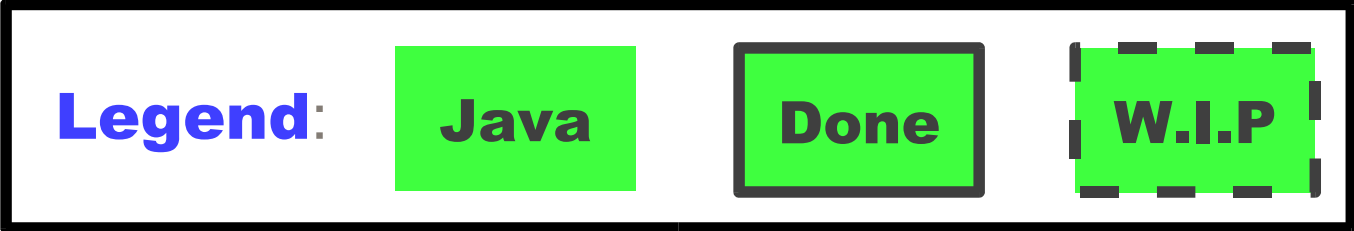
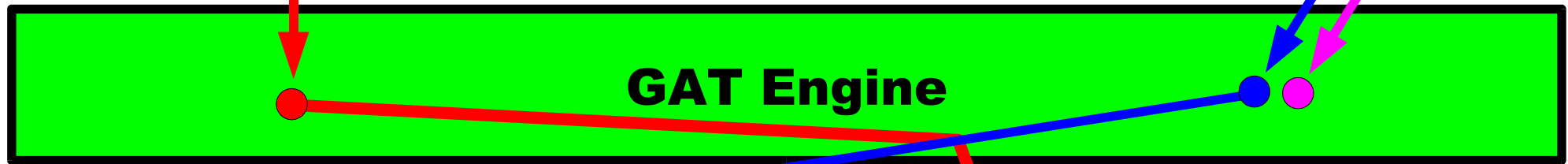
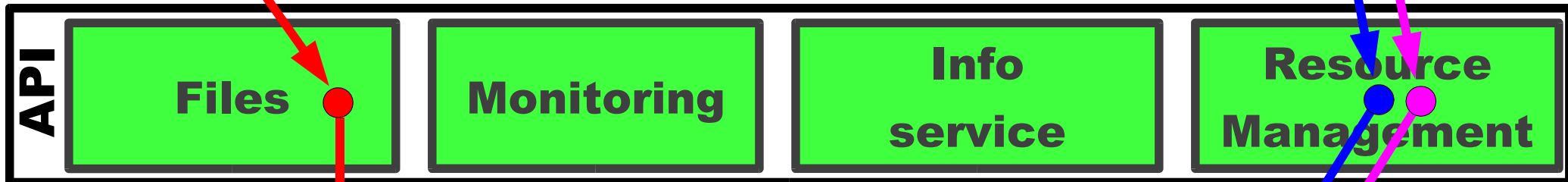
Java GAT Structure





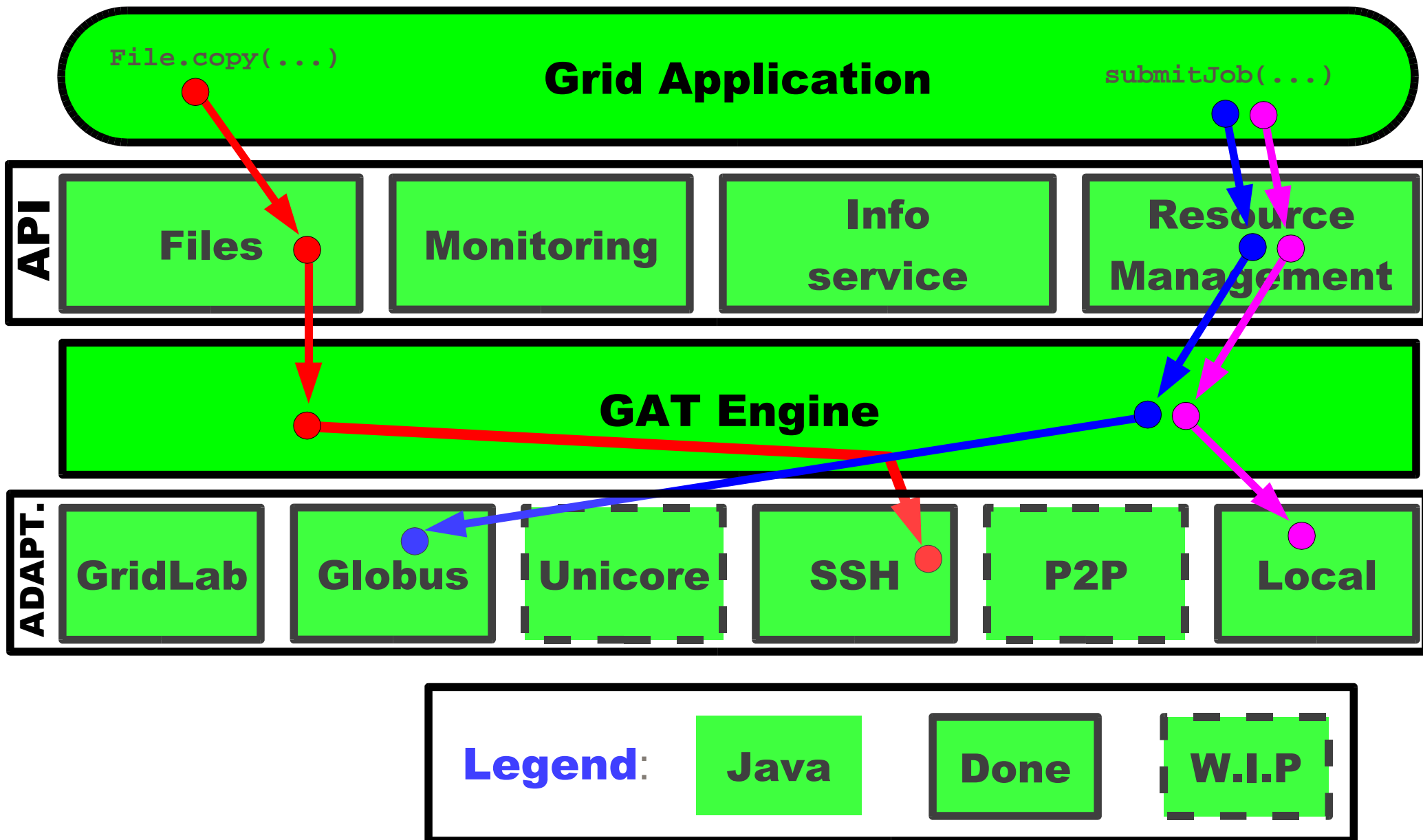
Java GAT Structure







Java GAT Structure





Implementation



- Adaptors are Java JAR files, dynamically loaded into the application.
- Late binding (specification allows this):
 - The GAT engine selects the best adaptor for each method.
 - Example: Create file object.
 - Engine selects and instantiates all adaptors than can implement this file object.
 - File.copy from site A to site B and C.
 - A -> B copy with GridFTP.
 - A -> C copy with GridLab Data movement.
 - Provides flexibility and fault tolerance.



Spec vs JavaGAT (1)



- Memory management is done for you (GC).
- GAT utility objects are not needed (list, table).
- No status objects, but exceptions
 - Can be nested.
- Security contexts
 - Avoid giving away credentials / passwords.
 - User can attach meta information to context.
 - Context can only be used by adaptor X.
 - Context can only be used with host Y.
 - Especially useful for passwords (FTP).



Spec vs JavaGAT (2)



- File and Streaming API:
 - JavaGAT subclasses existing “java.io” classes.
 - Streams and random access are separate
 - The GAT specification defines seeks on streams.
 - Directories are “first class citizens” and can be moved, copied, etc.
 - Directories can be used with any adopter
 - Stage in and out with resource broker.
 - Any Java code that uses files can now transparently use remote files.
 - File operations (getSize, isReadable).
 - Random File access.
 - Streaming.



Available Adaptors (1)



- Complete set of local adaptors.
- Generic:
 - Service locator (Igrid); used by GridLab adaptors.
 - Stage in / out on top of GAT file; used by broker adaptors.
 - Makes writing new adaptors easier
- File, streams and random file access:
 - GridLab data service
 - FTP, HTTP, HTTPS
 - GridFTP
 - SSH, SFTP
- Logical file (replication):
 - Generic adaptor on top of GAT File
 - Logical file adaptor for GridLab replica service.



Available Adaptors (2)



- Pipe:
 - Sockets
- Information service:
 - GridLab Storagebox
- Resource broker:
 - GridLab GRMS
 - Globus
 - SSH
 - ProActive
- Monitoring / Steering
 - (GridLab) Mercury
 - Adaptor-specific (job management)
- Work-in-progress:
 - Resource broker adaptor for Unicore (Peggy Lindner).



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- **Security**
- Grid I/O
- Resource Management
- Application Information Management
- Monitoring

- Hands-on session



Some Basic GAT Objects



- Preferences
 - Key value pairs
 - adaptor-specific instructions
 - Global or local, local overrides global
 - Example: ("File.adaptor.name", "globus")
- GATContext
 - Contains security information
 - Contains global preferences
 - There can be more than one context
 - Needed to create GAT objects
- GAT
 - Factory for all GAT objects
- GAT Exceptions
 - Nested, helps debugging (Needed because of late binding)



Some Basic GAT Objects



- URI
 - Slightly different semantics compared to java.net.URI
 - Use the right number of /'s in the URI's
 - Full URI is easy....
 - protocol://machine/<path>file
 -but some fields may be blank
 - file:///output (local file in current directory)
 - file:////output (local file in root (/) directory)
 - file://///tmp/output (local file in /tmp directory)
 - ftp://10.0.0.1/output (remote file in default ftp dir)
- Use the right scheme (protocol) in the URI:
 - JavaGAT can choose (late binding):
 - any://
 - Force a specific adaptor (early binding):
 - ftp://, gsiftp://, http://, file://,



Example



```
GATContext context = new GATContext();
Preferences prefs = new Preferences();
prefs.put("File.adaptor.name", "globus");
context.addPreferences(prefs);
```

```
src = new URI("hello");
file = GAT.createFile(context, src);
```

OR use local preferences to override globals:

```
file = GAT.createFile(context, morePrefs, src);
```



GAT Security



- SecurityContext
 - A container for security Information.
- Abstract, use subclasses
 - PasswordSecurityContext
 - CertificateSecurityContext
 - MyProxyServerCredentialSecurityContext
- Notes restrict the access to the context
 - Avoid broadcasting of passwords / credentials
 - Restrict access to a set of hosts or adaptors
 - One or more notes -> restricted to those adaptors/hosts
 - No notes -> any adaptor can use context for any host
- Typically not needed if default credentials / private keys are used



GAT Security example



```
GATContext context = new GATContext();
SecurityContext pwd =
    new PasswordSecurityContext(username, password);

// restrict access
pwd.addNote("hosts", "hostname1:port1,hostname2:port2");
pwd.addNote("adaptors", "ftp,ssh");

SecurityContext cert =
    new CertificateSecurityContext(keyfile, username, passphrase);

// add them to the GAT context
context.addSecurityContext(pwd);
context.addSecurityContext(cert);
```



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- Security
- **Grid I/O**
- Resource Management
- Application Information Management
- Monitoring

- Hands-on session



Grid I/O Classes



Information Society
Technologies

- File
 - FileInputStream / FileOutputStream
 - RandomAccessFile
- } Extend java.io classes
- ↓
- Grid-enable your code, just by replacing one "new" statement
- LogicalFile
 - Replicated file support
 - Basic Inter-process communication
 - Endpoint
 - Pipe
 - PileListener



GAT File



Information Society
Technologies

- Represents both files and directories (like java.io)
 - canRead, canWrite
 - delete
 - mkdir
 - list
 - copy
 - move
 - ...



GAT File example



```
package tutorial;

import org.gridlab.gat.*;
import org.gridlab.gat.io.File;

public class RemoteCopy {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();

        URI src = new URI(args[0]);
        URI dest = new URI(args[1]);
        File file = GAT.createFile(context, src);           // create file object

        file.copy(dest);                                   // and copy it
        GAT.end();
    }
}
```



GAT File Streaming Example



```
package tutorial;
import java.io.*;
import org.gridlab.gat.*;
import org.gridlab.gat.io.FileInputStream;
```

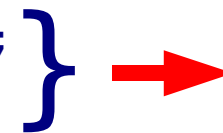
```
class RemoteCat {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();
        URI loc = new URI(args[0]);
```

```
        FileInputStream in = GAT.createFileInputStream(context, loc);
        InputStreamReader reader = new InputStreamReader(in);
        BufferedReader buf = new BufferedReader(reader);
```

```
        String result = buf.readLine();
        System.err.println("result = " + result);
        in.close();
```

```
    }
```

```
}
```



Standard
java.io
classes



GAT Remote Random Access Files



- Random access to remote files
 - read
 - write
 - seek
 - length
 - ...



GAT Logical File (replica management)



- LogicalFile class
 - An abstract representation of a set of identical physical files
 - addFile / addURI
 - removeFile / removeURI
 - **replicate(URI destination)**
- Replicate a logical file to a new location.
 - Copy one of the files in the set to the new location
 - Choose the “best” one
 - Closest in terms of bandwidth
 - Cheapest
 - ...
 - Depends on adaptor
- Typically used for staging in files for jobs
 - Resource broker (or GAT) chooses one of the files in the set



GAT Inter-Process Communication (1)



- Basic IPC primitives
 - Not intended for parallel programming
 - API looks like socket
 - Point to point only
 - Protocol independent
- Endpoint
 - Create them
 - **listen** (bind/accept in “socket speak”)
 - **connect** (setup connection to another Endpoint)
 - **listen** and **connect** both return a Pipe
- Pipe
 - A connection between to Endpoints (like a socket)
 - Provides InputStream and OutputStream



GAT Inter-Process Communication (2)



- Endpoints are shared using the GAT application information service
- Asynchronous connection setup
- PipeListener
 - `processPipe(Pipe pipe);`
- `Endpoint.listen(PipeListener p)`
 - Asynchronous (returns immediately)
 - When a remote site connects, GAT calls `processPipe` on listener



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- Security
- Grid I/O
- **Resource Management**
- **Application Information Management**
- **Monitoring**

- **Hands-on session**



GAT Resource Management Classes (1)



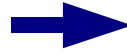
- ResourceDescription
 - SoftwareResourceDescription
 - Describes a software component: library or Operating System, etc
 - Key/value pairs
 - “os.name”, “linux”
 - “os.release”, “2.6”
 - HardwareResourceDescription
 - Describes a hardware component: machine to run a job on
 - Key/value pairs
 - “memory.size”, “1GB”
 - “machine.type”, “i686”
 - Some keys and values defined in the spec, others may be adaptor specific
 - Both have a list of additional ResourceDescriptions
 - Build trees of ResourceDescriptions



Two requirements



SoftwareResourceDescription
os.name=linux



HardwareResourceDescription
machine.type=i686

```
SoftwareResourceDescription srd =  
    new SoftwareResourceDescription();  
srd.addResourceAttribute("os.name", "linux");
```

```
HardwareResourceDescription hrd =  
    new HardwareResourceDescription();  
hrd.addResourceAttribute("machine.type", "i686");
```

```
srd.addResourceDescription(hrd); // require srd and hrd
```



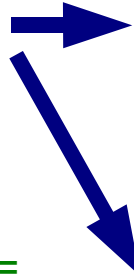
More Complex Descriptions



SoftwareResourceDescription
os.name=linux

HardwareResourceDescription
machine.type=i686

HardwareResourceDescription
machine.type=power4



```
SoftwareResourceDescription srd =  
    new SoftwareResourceDescription();  
srd.addResourceAttribute("os.name", "linux");
```

```
HardwareResourceDescription hrd1 = new HardwareResourceDescription();  
hrd1.addResourceAttribute("machine.type", "i686");
```

```
HardwareResourceDescription hrd2 = new HardwareResourceDescription();  
hrd2.addResourceAttribute("machine.type", "power4");
```

```
srd.addResourceDescription(hrd1); // We must have Linux  
srd.addResourceDescription(hrd2); // on either i686 or power4
```



GAT Resource Management Classes (2)



- Resource
 - SoftwareResource
 - HardwareResource
- Reservation
- SoftwareDescription
 - Executable
 - Arguments
 - stdin, stdout, stderr
 - Pre-staged, post-staged files
- JobDescription
 - Must have SoftwareDescription
 - Has ResourceDescription or Resource

Describes the software you want to run on the grid

Describes your job:
Software and Requirements (for brokering)
or
Software and a specific resource to run on



GAT Resource Management Classes (3)



- ResourceBroker
 - Submit job description
 - Find resources
 - Make reservations
- Job
 - Result of job submission
 - Query state
 - Cancel
 - Checkpoint
 - migrate



Resource Management Example (1)



```
public class SubmitLocalJob {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();

        SoftwareDescription sd = new SoftwareDescription();
        sd.setLocation("file:///bin/hostname");
        File stdout = GAT.createFile(context, "hostname.txt");
        sd.setStdout(stdout);

        JobDescription jd = new JobDescription(sd);
        ResourceBroker broker = GAT.createResourceBroker(context);
        Job job = broker.submitJob(jd);
        while (job.getState() != Job.STOPPED &&
            job.getState() != Job.SUBMISSION_ERROR) Thread.sleep(1000);
    }
}
```



Resource Management Example (2)



```
public class SubmitRemoteJob {
    public static void main(String[] args) throws Exception {
        // ... above code is the same as before
        ResourceDescription rd = new HardwareResourceDescription();
        rd.addAttribute("machine.node", args[0]);

        JobDescription jd = new JobDescription(sd, rd);
        ResourceBroker broker = GAT.createResourceBroker(context);
        Job job = broker.submitJob(jd);

        while (job.getState() != Job.STOPPED &&
            job.getState() != Job.SUBMISSION_ERROR) {
            System.err.println("job state = " + job.getInfo());
            Thread.sleep(1000);
        }
    }
}
```



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- Security
- Grid I/O
- Resource Management
- **Application Information Management**
- **Monitoring**

- **Hands-on session**



GAT Advert Service



Information Society
Technologies

- Advert service is a directory with an hierarchical name space
- The advert service can publish GAT objects
 - File
 - LogicalFile
 - Endpoint
 - Resource (software, hardware)
 - Job
- User attaches meta data to GAT objects
- Application can query, based on meta data
- Advert service is persistent
- Advert service is language independent



Advert Service Use Cases



Information Society
Technologies

- Publish your (replicated) files, so other applications can open them
- Publish Endpoints to establish inter process communication
- Publish the (hardware) resources you found
- Spawn jobs, publish their Job object, let other applications (like a portal) monitor them



Advert Service Classes



- Advertisable
 - Marker interface (like java.io.Serializable)
 - GAT objects that implement this interface can be advertised
- MetaData
 - Table of (key,value) pairs (Strings)
 - Attached to Advertisable
- AdvertService
 - Publish adverts at some path in the hierarchical name space
 - Path uniquely identifies advert (one advert at any given path)
 - Supports current working directory (relative paths)
 - `add(advert, metaData, path)`
 - `getAdvertisable(path)`
 - `find(metaData)` metaData can contain regular expressions
 - ...



Example: Publishing GAT Objects



```
public class AdvertServicePublish {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();
        AdvertService advert = GAT.createAdvertService(context);

        File file = GAT.createFile(context, "foo");
        MetaData meta = new MetaData();
        meta.put("name", "myTestFile");
        meta.put("purpose", "tutorial");
        advert.add(file, meta, "/home/rob/tutorialFile");

        Endpoint e = GAT.createEndpoint(context);
        meta = new MetaData();
        meta.put("name", "myTestEndpoint");
        meta.put("purpose", "tutorial");
        advert.add(e, meta, "/home/rob/tutorialEndpoint");
    }
}
```



Example: retrieving GAT Objects



```
public class AdvertServicePrinter {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();

        AdvertService advert = GAT.createAdvertService(context);

        Advertisable result = advert.getAdvertisable(args[0]);
        if(result == null) {
            System.err.println("object not found");
            return;
        }

        System.err.println("got object back: " + result);
    }
}
```



Querying the Advert Service



- Create MetaData query description
 - All fields in query must be matched
 - Results may have additional fields
 - MetaData value fields can be regular expressions

```
/home/rob/tutorialFile  
"name", "myTestFile"  
"purpose", "tutorial"
```

```
/home/rob/tutorialEndpoint  
"name", "myTestEndpoint"  
"purpose", "tutorial"
```

- {("purpose", "tutorial")} matches both
- {("purpose", "tutorial"), ("location", "any")} matches none
- {("name", ".*File")} matches blue
- {("name", "my.*")} matches both



Example: Query the Advert Service



```
public class AdvertServiceQuery {
    public static void main(String[] args) throws Exception {
        GATContext context = new GATContext();
        AdvertService advert = GAT.createAdvertService(context);

        Metadata meta = new Metadata();
        meta.put(args[0], args[1]);

        String[] paths = advert.find(meta);
        if (paths == null) { System.err.println("no objects found");return; }

        System.err.println(paths.length + " adverts found");
        for (int i = 0; i < paths.length; i++) {
            Advertisable result = advert.getAdvertisable(paths[i]);
            System.err.println("advert " + i + " is " + result);
        }
    }
}
```



Overview



- What is GAT and why do we need it?
- JavaGAT structure and overview
- Security
- Grid I/O
- Resource Management
- Application Information Management
- **Monitoring**
- Hands-on session



GAT Monitoring and Steering System use cases



- Monitor and steer applications
 - Get application info (current iteration number)
 - Tell application to checkpoint
 - Change application variable to steer simulation
 - ...
- Monitor grid resources
 - Host load
 - Free disc space
 - Current available network bandwidth
 - ...
- Monitor and steer GAT objects
 - Monitor progress of GAT file copy
 - Monitor job status (is it in the queue or already scheduled)
 - ...



Monitoring System Classes (1)



- Monitorable
 - A marker interface indicating which objects can be monitored
 - Interface to external monitoring system
- Monitorable GAT objects
 - File
 - LogicalFile
 - RandomAccessFile
 - FileInputStream
 - FileOutputStream
 - Endpoint
 - Pipe
 - AdvertService
 - Resource
 - Job



Monitoring System Classes (2)



- **MetricDefinition**
 - A description of a measurable quantity within a monitoring system (free disk space)
 - Defines:
 - Measurement type: continuous / discrete
 - Data type and unit for parameters and return value
 - Create metrics by supplying parameters
 - Host name
 - Device name: /dev/hda
 - Monitoring system says:
 - I can measure free disk space.
 - I need two String parameters (device name, host name)
 - I will return a long value which is the free disk space in megabytes.
- **Get list of MetricDefinitions from Monitorable**



Monitoring System Classes (3)



- Metric
 - Use a metric definition to define the metric you are interested in
 - MetricDefinition is: I can measure disk space
 - Metric is an **instance** of this: disk space on localhost, /dev/hda
 - Create by `MetricDefinition.createMetric("localhost", "/dev/hda");`
- MetricValue
 - A value measured by the monitoring system
 - Poll monitoring system with `Monitorable.getMeasurement(Metric)`
 - Contains time stamp, value
- MetricListener
 - Asynchronous monitoring handler, produces new MetricValues
 - Can be called whenever something changes
 - Can be called with a specific frequency (for continuous metrics)



GAT support for external Monitoring Systems



- External monitoring systems
 - Mercury
 - Monitors grid resources: hosts, queues, etc.
 - Monitors / steers applications
 - the GridLab project
 - the Datagrid project
 - the Hungarian Supergrid project
 - the P-GRADE Parallel Grid Runtime and Application Development Environment
 - Delphoi
 - Monitors network
 - The GridLab project



Monitoring Example (1)



```
public class ListMetricDefinitions {
    public static void main(String[] args) throws Exception {
        GATContext c = new GATContext();
        c.addPreference("monitoring.adaptor.name", "mercury");

        Monitorable m = GAT.createMonitorable(c);

        List definitions = m.getMetricDefinitions();

        System.err.println("found " + definitions.size() + " definitions");

        for (int i = 0; i < definitions.size(); i++) {
            System.err.println(definitions.get(i));
        }
    }
}
```



Monitoring Example (2)



```
public class GetMetric {
    public static void main(String[] args) throws Exception {
        GATContext c = new GATContext();
        c.addPreference("monitoring.adaptor.name", "mercury");

        Monitorable m = GAT.createMonitorable(c);
        MetricDefinition def = m.getMetricDefinitionByName("host.mem.free");

        HashMap params = new HashMap();
        params.put("host", args[0]);

        Metric metric = def.createMetric(params);
        MetricValue result = m.getMeasurement(metric);

        System.err.println("Free memory at " +
            new Date(result.getEventTime()) + " was: " + result.getValue());
    }
}
```



The Power of GAT (1)



- Submit job to remote resource
- Stage in a directory
- Get output back
- Monitor the Job

- Resource broker independent
- Transfer protocol independent

- **In 70 lines of code!**



The Power of GAT (2)



```
public class SubmitJobCallback implements MetricListener {
    boolean exit = false;

    public void start(String[] args) throws Exception {
        GATContext context = new GATContext();
        Job job = submitJob(context, args[0]);

        MetricDefinition md = job.getMetricDefinitionByName("job.status");
        Metric m = md.createMetric(); // no parameters needed for metric
        job.addMetricListener(this, m); // register callback for job.status

        // wait until job is done
        synchronized (this) {
            while (!exit) wait();
        }
    }
}

// .... continued on next slide ....
```



My `processMetricEvent` method will be called when `job.status` changes



The Power of GAT (3)



```
public synchronized void processMetricEvent(MetricValue val) {
    String state = (String) val.getValue();

    System.err.println("SubmitJobCallback: Processing metric: "
        + val.getMetric() + ", state is " + state);

    if (state.equals("STOPPED") || state.equals("SUBMISSION_ERROR")) {
        exit = true;
        notifyAll();
    }
}

// .... continued on next slide ....
```



The Power of GAT (4)



```
Job submitJob(GATContext context, String hostname) throws Exception {
    File outFile = GAT.createFile(context, "any:///hostname.txt");
    File stageInDir = GAT.createFile(context, "any:///home/rob/testDir");

    SoftwareDescription sd = new SoftwareDescription();
    sd.setLocation("file:///bin/hostname");
    sd.setStdout(outFile);
    sd.setPreStaged(stageInDir);

    Hashtable attributes = new Hashtable();
    attributes.put("machine.node", hostname);
    ResourceDescription rd = new HardwareResourceDescription(attributes);

    JobDescription jd = new JobDescription(sd, rd);
    ResourceBroker broker = GAT.createResourceBroker(context);
    return broker.submitJob(jd);
}
```



The future of GAT



- GAT will be supported for at least the next two years
- GAT is being standardized within GGF
 - This will take time
 - GAT 2.0 will be called SAGA (Simple API for Grid Applications)
 - SAGA is very close to GAT
 - Adds task model to GAT for asynchronous grid operations
- We will offer a migration path from GAT to SAGA



Conclusion



- The GAT provides a simple and stable API to various Grid environments
- But powerful!
- Independent of grid middleware
- Portable
- Downloads:
 - www.gridlab.org
 - Distributions
 - Anonymous CVS access
 - C Platforms: Linux, Windows, Mac OS X, SGI Irix, True64 UNIX
 - Java Platforms: any (Java 1.4 or higher)
 - Support via mailing list gat@gridlab.org



Hands-on Session



- Requirements:
 - Java 1.4 or newer
 - Download at java.sun.com
 - Ant 1.6 or newer
 - Java's equivalent of “make”
 - Download at ant.apache.org
- The Java GAT
 - www.gridlab.org
 - Click Grid Application Toolkit
 - Click GAT releases (on the left)
 - Download latest Java GAT version