UniGrids

Integration of GridFTP in UNICORE

First UNICORE Summit

Sophia Antipolis, 12 Oct 2005

Simone Lanzarini, CINECA
s.lanzarini@cineca.it

Outline

- Why do we need to integrate GridFTP in Unicore?
- What’s Unicore ARFT?
- ARFT architecture
- ARFT performance
- ARFT over “slow” connections
- What’s next

Unicore ARFT

- Scenario: multi-site job where a huge amount of data has to be transferred between sites
- The UNICORE internal file transfer mechanism (UPL) is not optimal for transfers of large amounts of data both between clients and Vsites and among Vsites.
- NJS has been extended to use Alternative File Transfer (AFT) mechanisms
- Globus 4.0 Reliable File Transfer (RFT) as an Alternative File Transfer for Unicore: **A**lternative **R**eliable **F**ile **T**ransfer (ARFT)

Why Globus RFT?

Why Globus RFT?

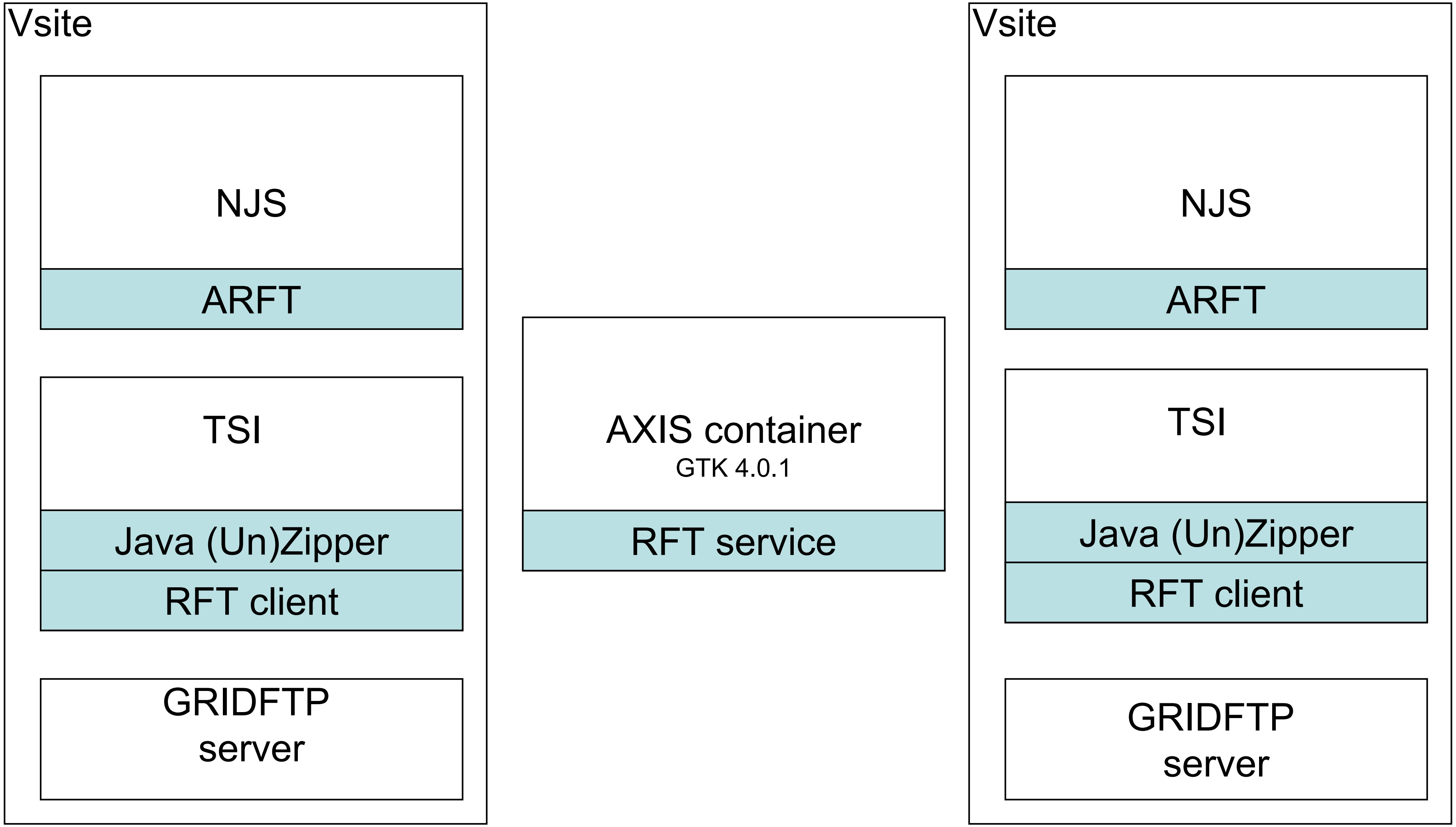
- GridFTP is the de-facto standard for file transfers in Grid environments

- Globus RFT inherits performance and features provided by GridFTP:
  - ♦ third-party control of data transfer
  - ♦ parallel / striped data transfer
  - ♦ partial file transfer
  - ♦ manual control of TCP buffer size

- Plus other features:
  - ♦ Transfer state is stored in a persistent manner:
    - in case of failure the transfer can be started from the last restart marker recorded
  - ♦ WSRF compliant

ARFT architecture

ARFT architecture



Vsite

NJS

ARFT

TSI

Java (Un)Zipper

RFT client

GRIDFTP server

AXIS container

GTK 4.0.1

RFT service

Vsite

NJS

ARFT

TSI

Java (Un)Zipper

RFT client

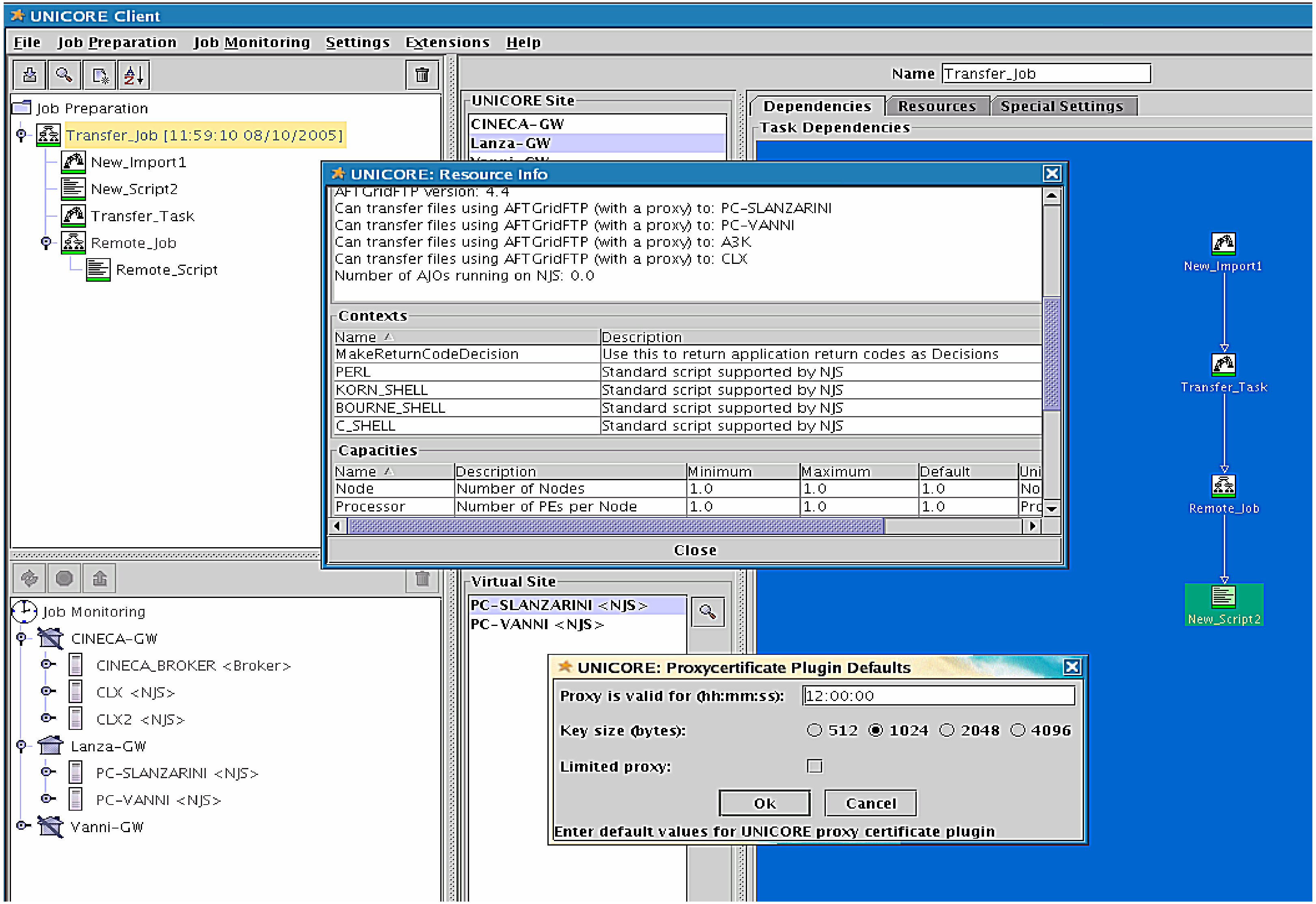
GRIDFTP server

ARFT overview

- ARFT is transparently used for transfers
  - The job must contain a transfer task between two Vsites
  - Both Vsites have to be properly set-up in order to support GridFTP transfers.
  - The Unicore Client uses the GlobusProxy plugin to automatically generate the proxy certificate needed to GSI

UniGrids

ARFT overview



Name: Transfer_Job

UNICORE: Resource Info

```
AFTGridFTP version: 4.4
Can transfer files using AFTGridFTP (with a proxy) to: PC-SLANZARINI
Can transfer files using AFTGridFTP (with a proxy) to: PC-VANNI
Can transfer files using AFTGridFTP (with a proxy) to: A3K
Can transfer files using AFTGridFTP (with a proxy) to: CLX
Number of AJOs running on NJS: 0.0
```

Contexts

Name	Description
MakeReturnCodeDecision	Use this to return application return codes as Decisions
PERL	Standard script supported by NJS
KORN_SHELL	Standard script supported by NJS
BOURNE_SHELL	Standard script supported by NJS
C_SHELL	Standard script supported by NJS

Capacities

Name	Description	Minimum	Maximum	Default	Uni
Node	Number of Nodes	1.0	1.0	1.0	No
Processor	Number of PEs per Node	1.0	1.0	1.0	Pro

UNICORE: Proxycertificate Plugin Defaults

Proxy is valid for (hh:mm:ss): 12:00:00

Key size (bytes): ○ 512 ● 1024 ○ 2048 ○ 4096

Limited proxy: □

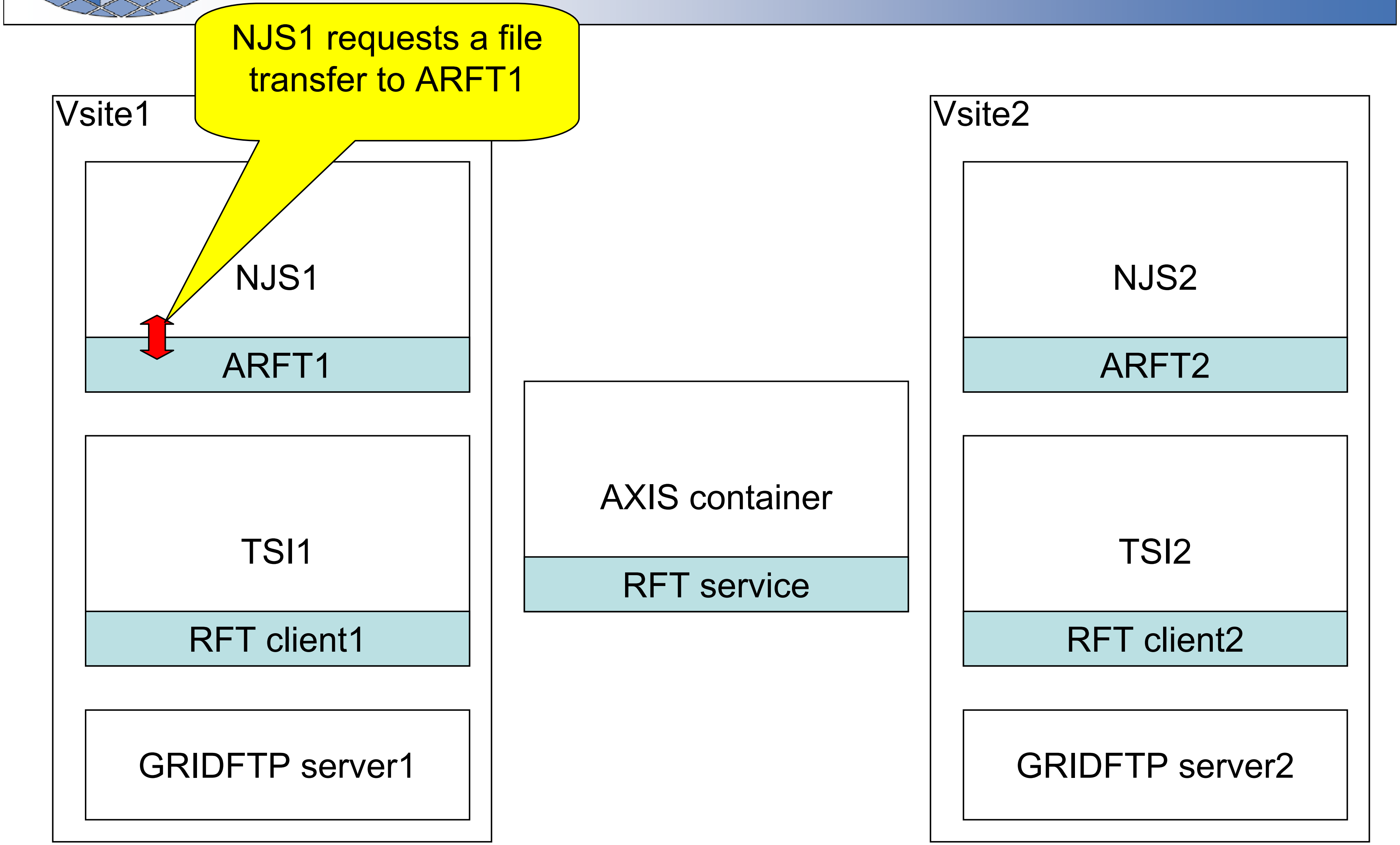
Enter default values for UNICORE proxy certificate plugin

CINECA Consorzio Interuniversitario

ARFT communications

UniGrids

Communications / 1



NJS1 requests a file transfer to ARFT1

Vsite1

NJS1

ARFT1

TSI1

RFT client1

GRIDFTP server1

AXIS container

RFT service

Vsite2

NJS2

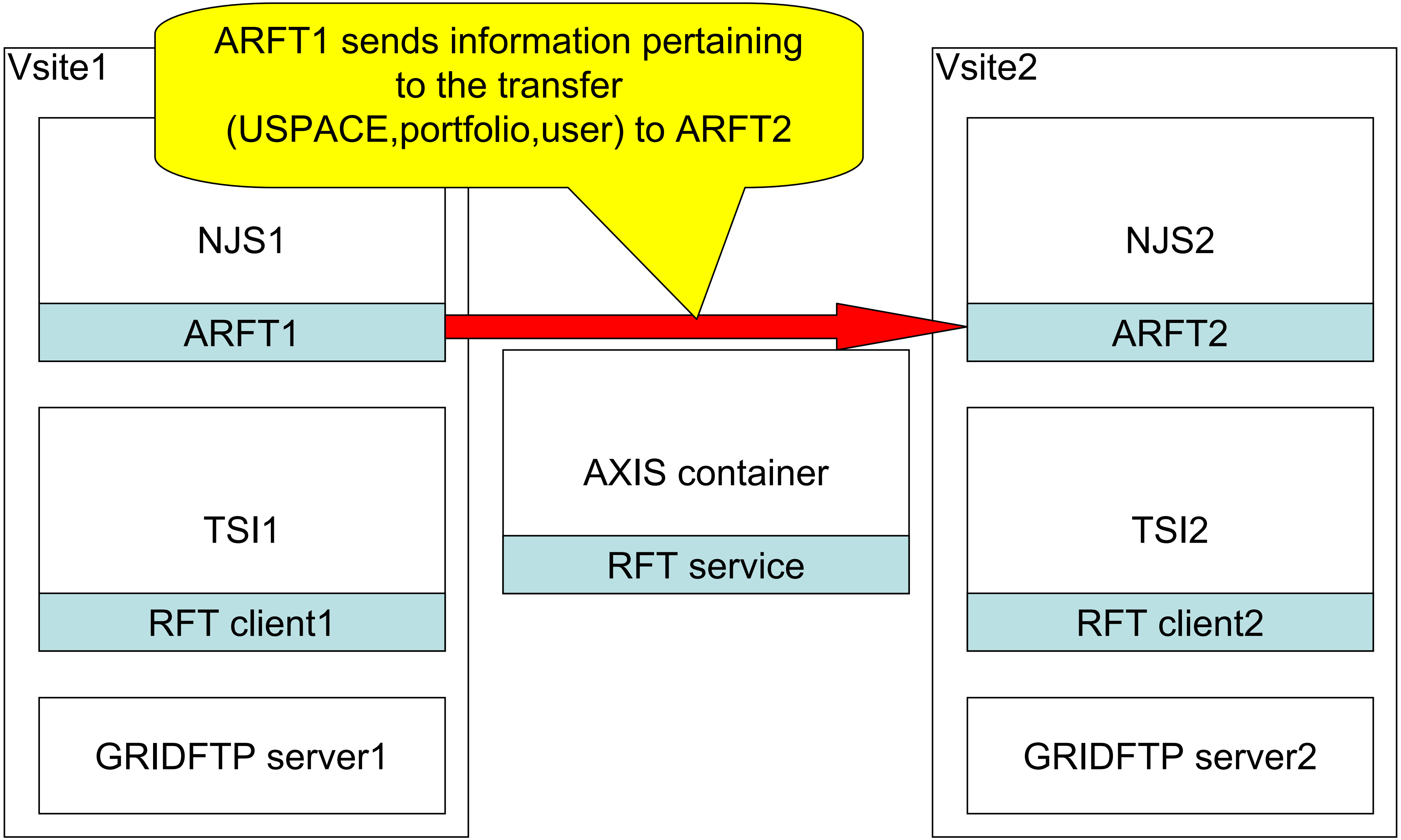
ARFT2

TSI2

RFT client2

GRIDFTP server2

Communications / 2



ARFT1 sends information pertaining to the transfer (USPACE,portfolio,user) to ARFT2

Vsite1

NJS1

ARFT1

TSI1

RFT client1

GRIDFTP server1

AXIS container

RFT service

Vsite2

NJS2

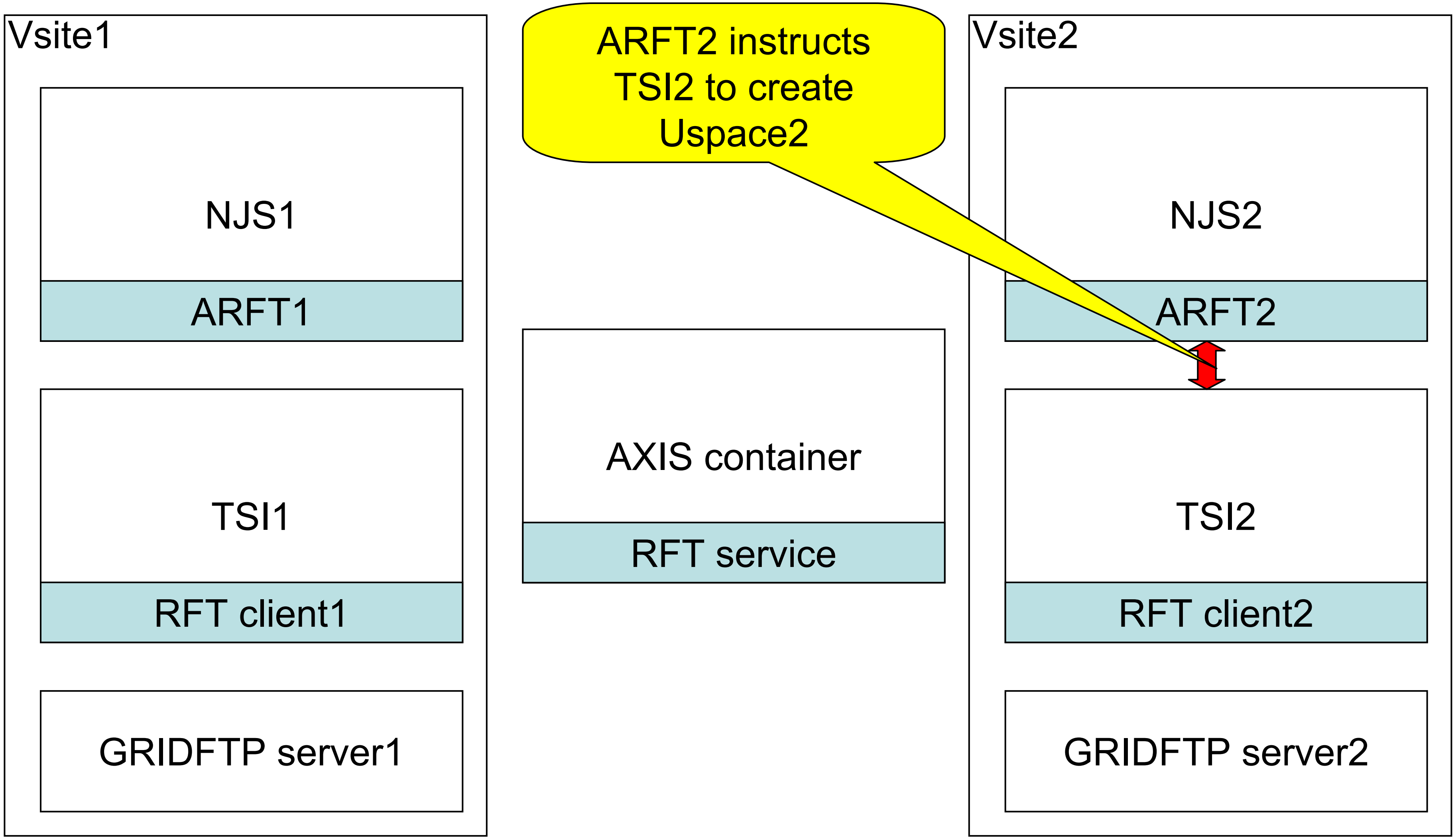
ARFT2

TSI2

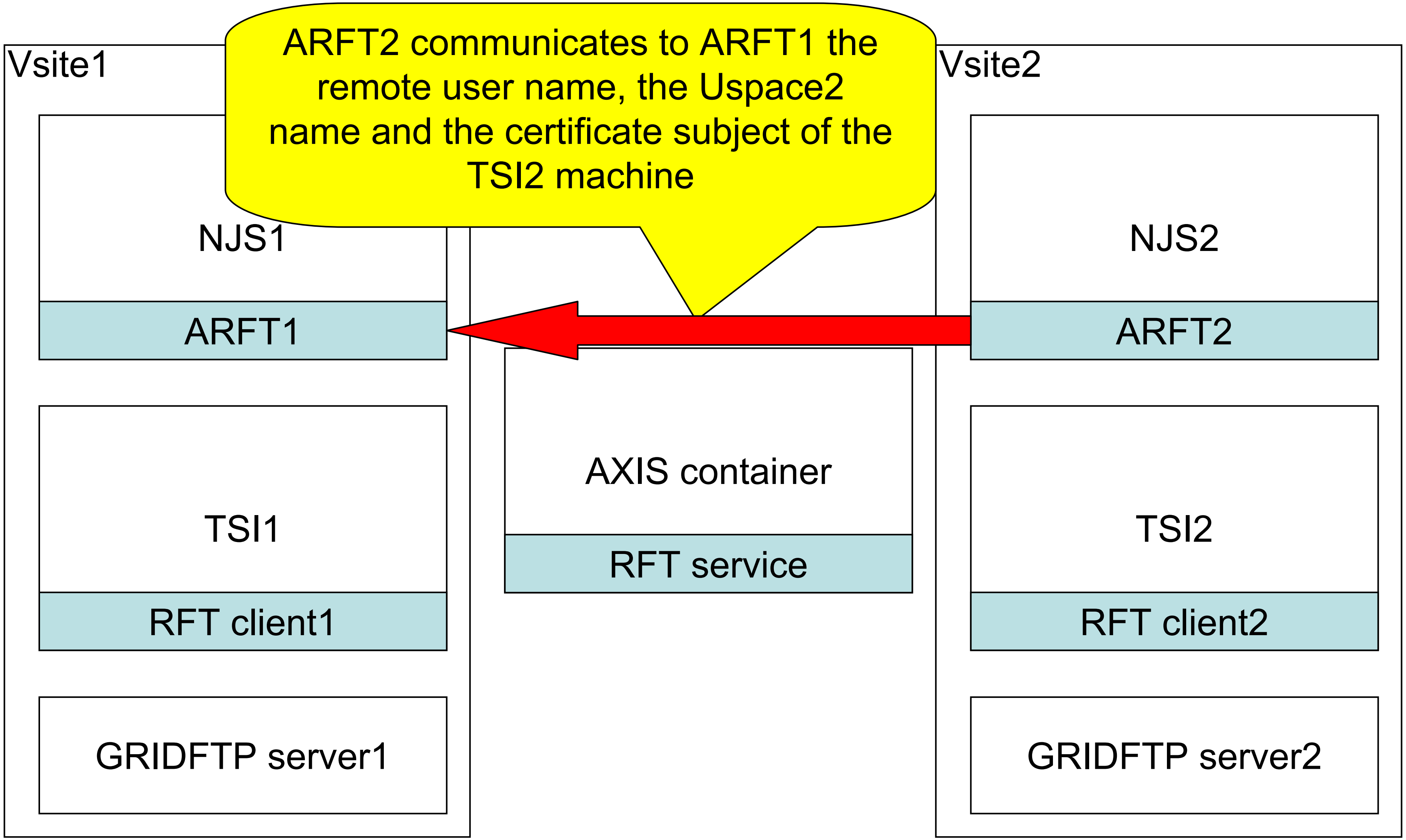
RFT client2

GRIDFTP server2

Communications / 3

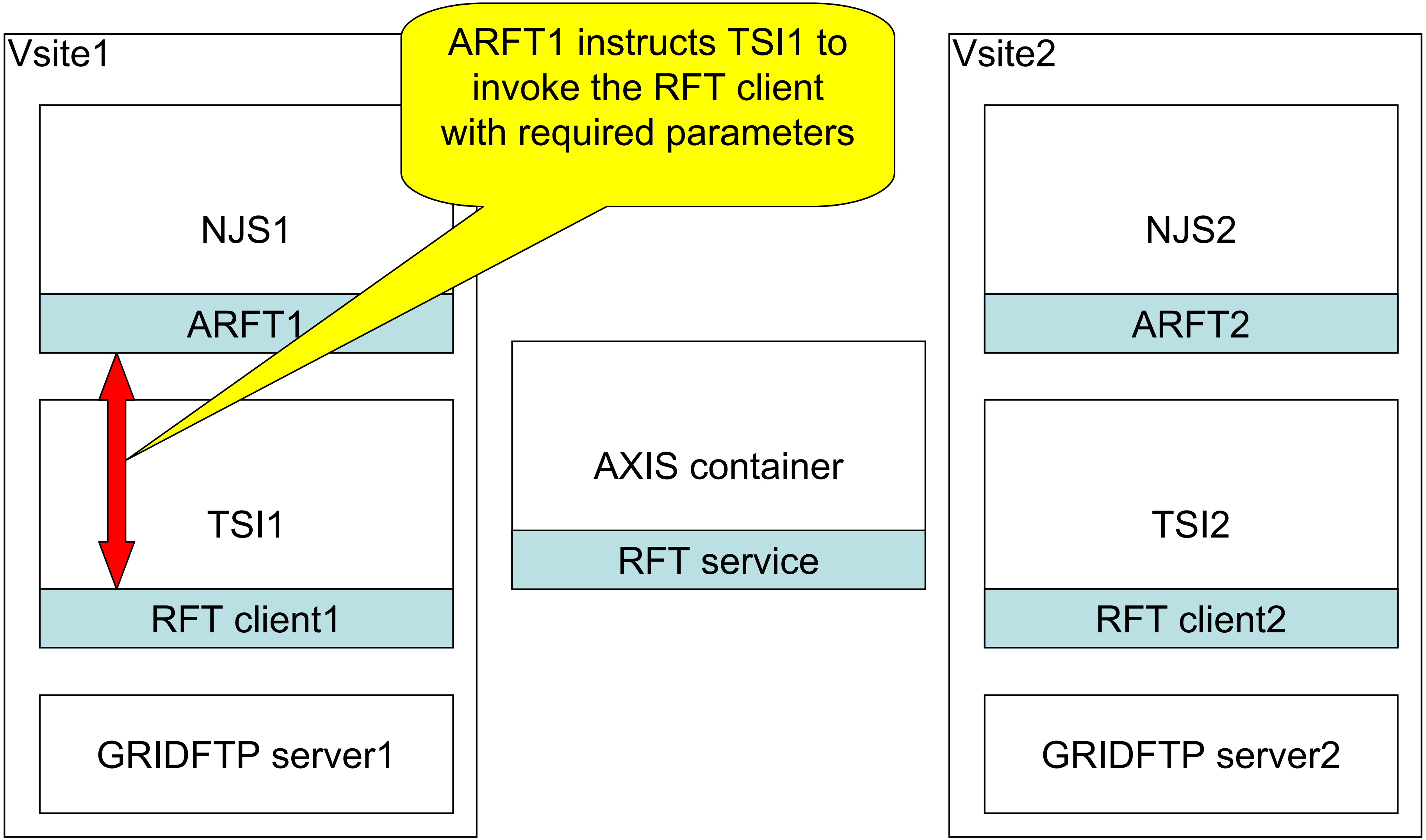


Communications / 4



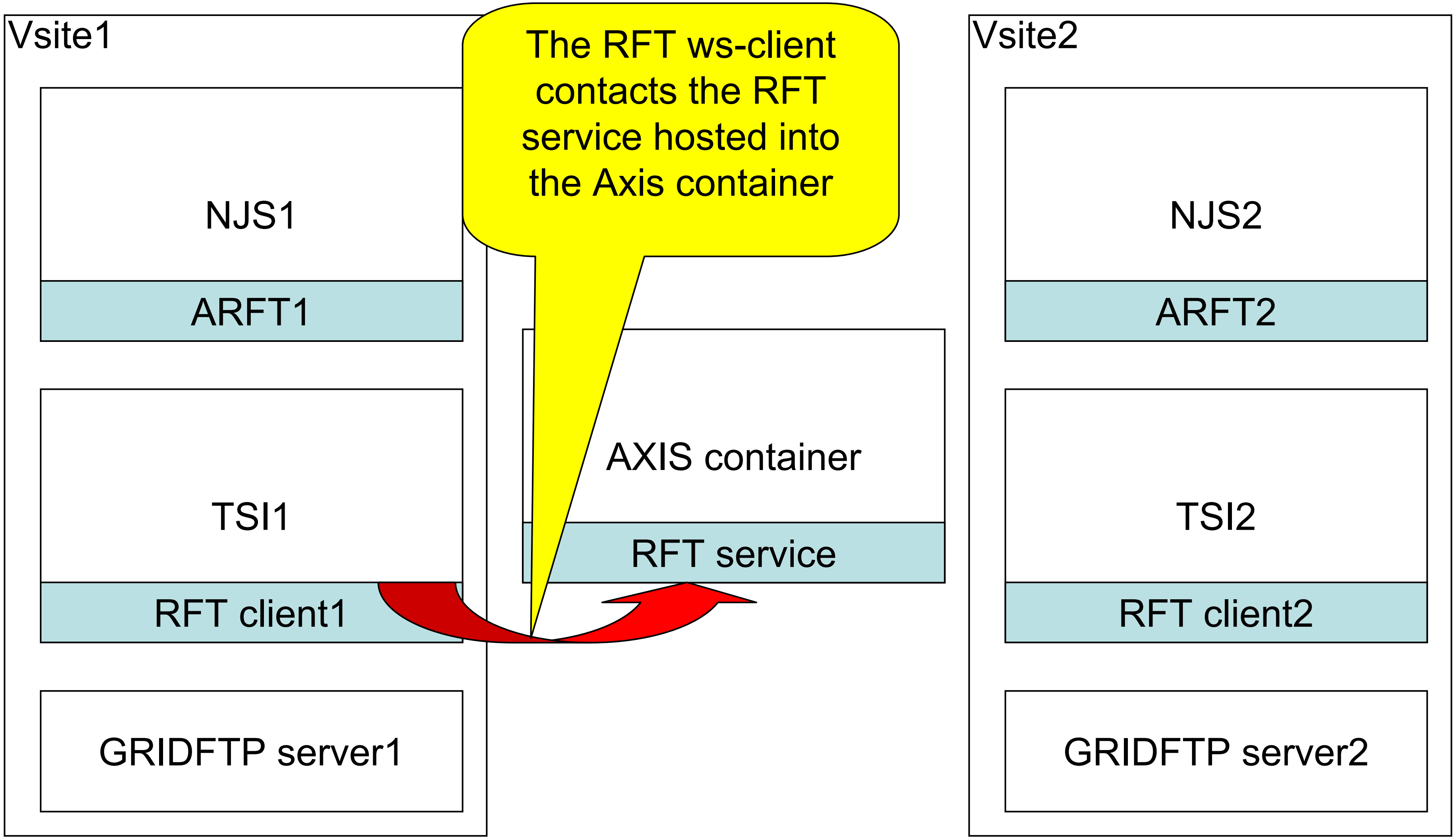
ARFT2 communicates to ARFT1 the remote user name, the Uspace2 name and the certificate subject of the TSI2 machine

Communications / 5



ARFT1 instructs TSI1 to invoke the RFT client with required parameters

Communications / 6



The RFT ws-client contacts the RFT service hosted into the Axis container

Vsite1

NJS1

ARFT1

TSI1

RFT client1

GRIDFTP server1

AXIS container

RFT service

Vsite2

NJS2

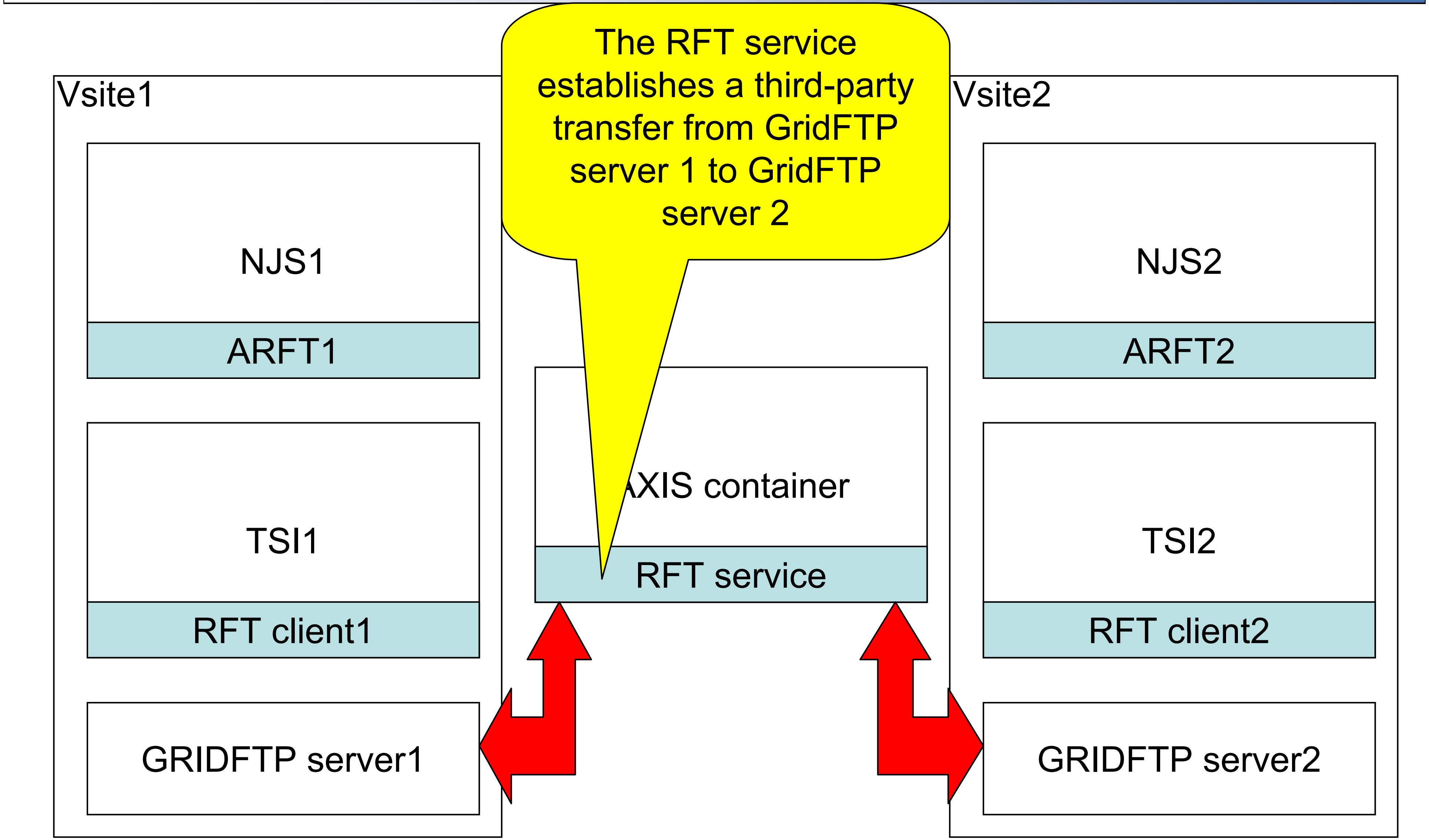
ARFT2

TSI2

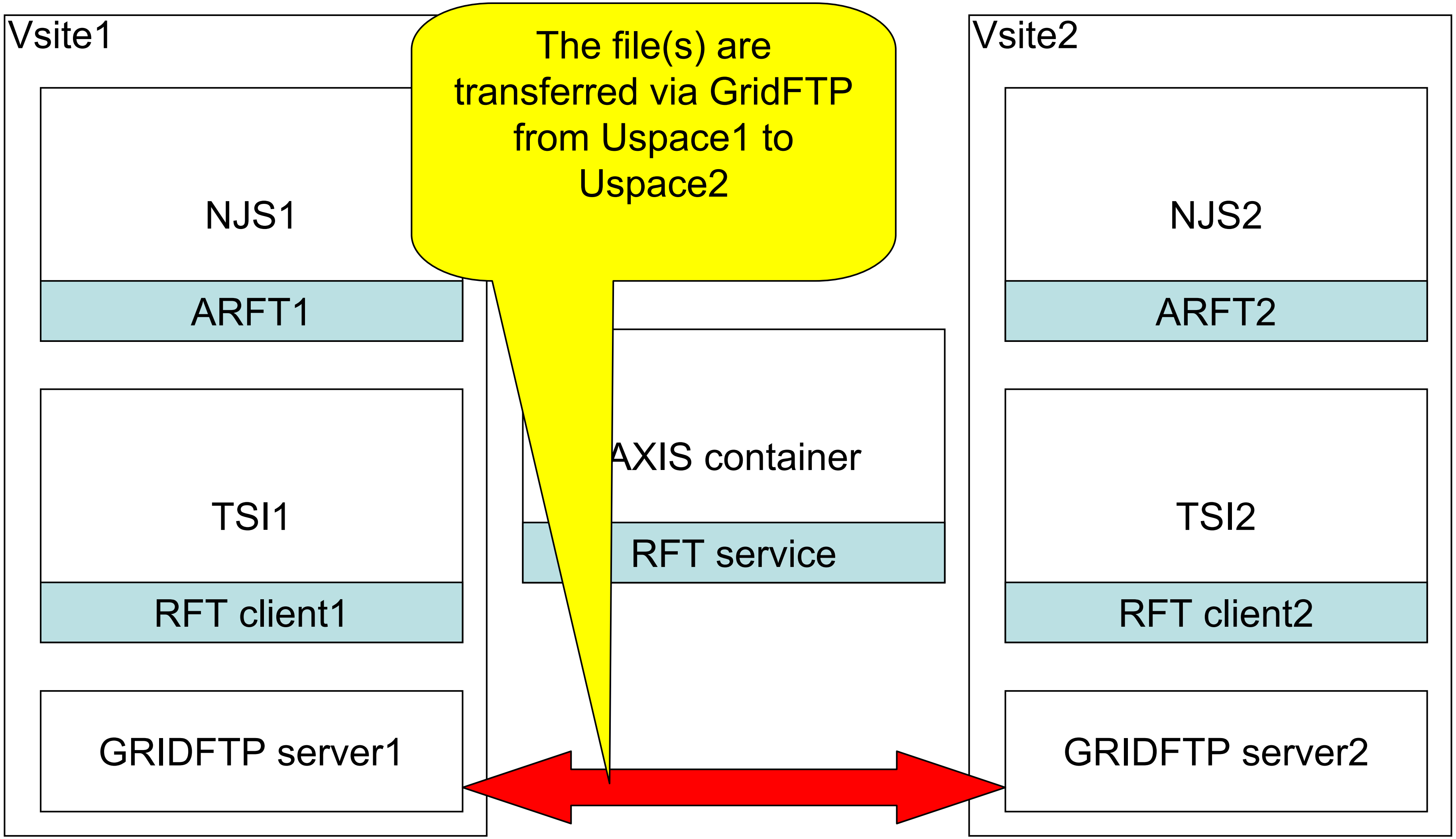
RFT client2

GRIDFTP server2

Communications / 7


The RFT service establishes a third-party transfer from GridFTP server 1 to GridFTP server 2

Communications / 8



The file(s) are transferred via GridFTP from Uspace1 to Uspace2

ARFT performance

UPL vs GridFTP

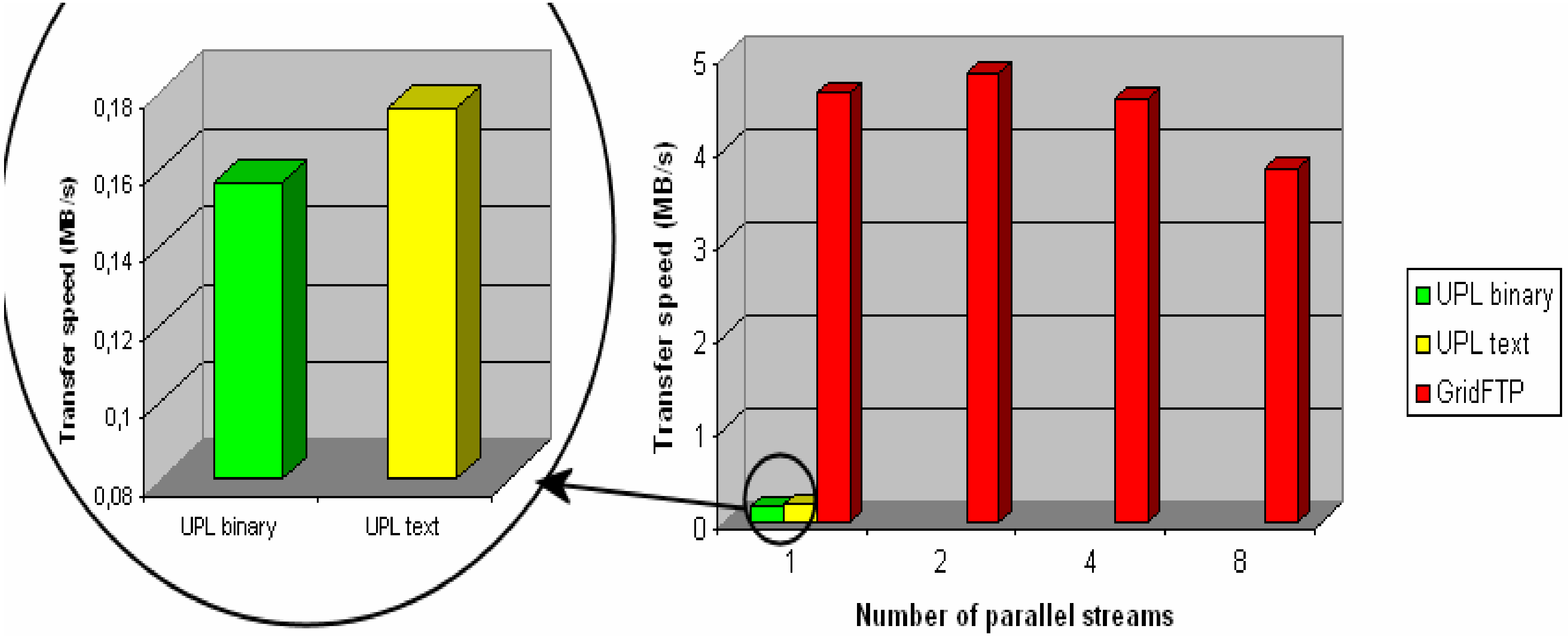
- In order to compare the performance of a file transfer via GridFTP with the performance of a file transfer via UPL, the transfer speed has been measured.
- Since UPL uses a Java ZipStream to compress data, two different file transfers for each file size have been performed with UPL: the former with a binary file with a low compression ratio (approx. 1%), the latter with a text file with periodic char sequences, with a very high compression ratio (approx. 99%).

UPL vs GridFTP

- The file sizes were 5, 15, 50, 100, 500 and 1000 MB.
- To evaluate the influence of multiple streams, each GridFTP file transfer was performed with 1,2,4 and 8 streams. Only for the 500 and 1000 MB file transfer the 16 and 32 streams configurations were tested.
- The tests have been performed over a 100 Mb LAN.

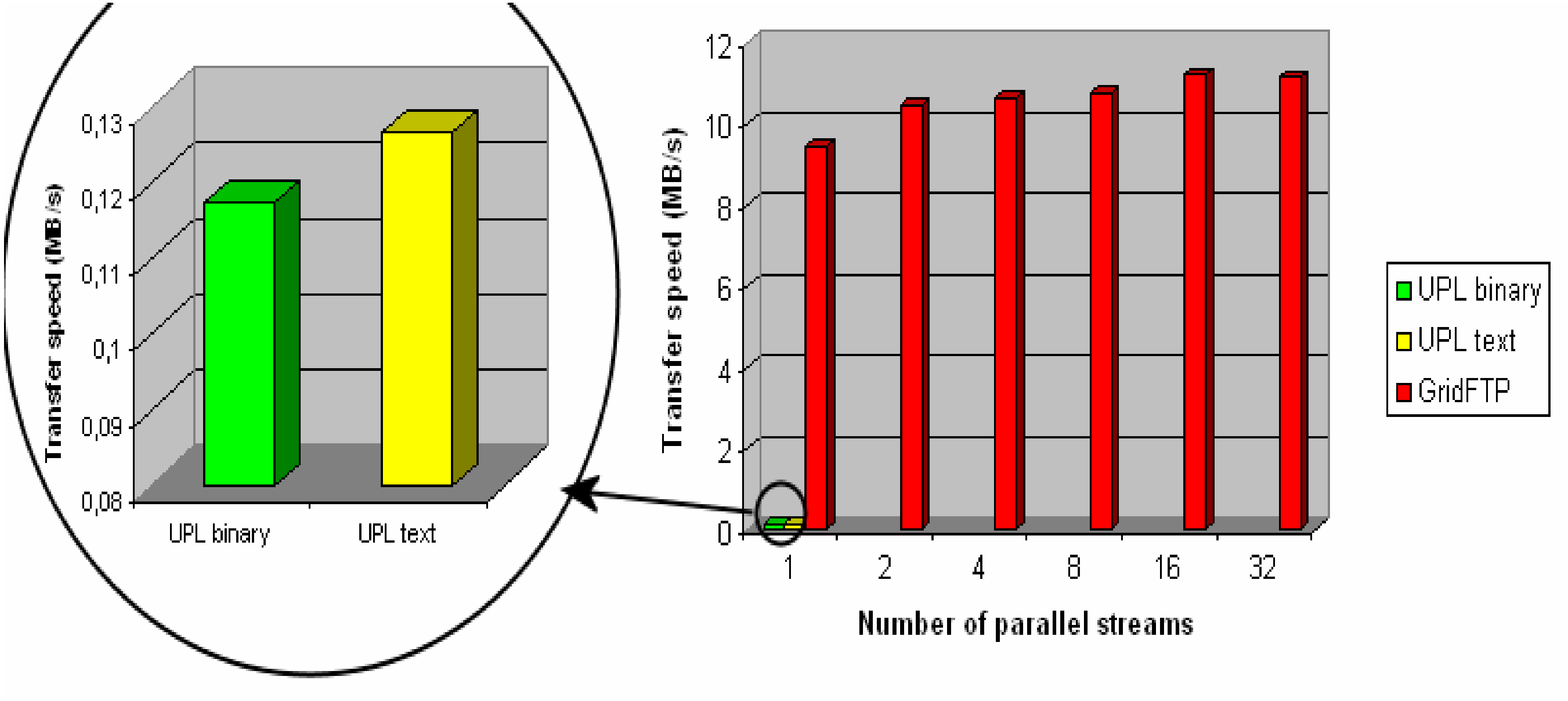
UPL vs GridFTP

- Average speed transferring 5 MB:



UPL vs GridFTP

- Average speed transferring 1 GB:



UPL vs GridFTP

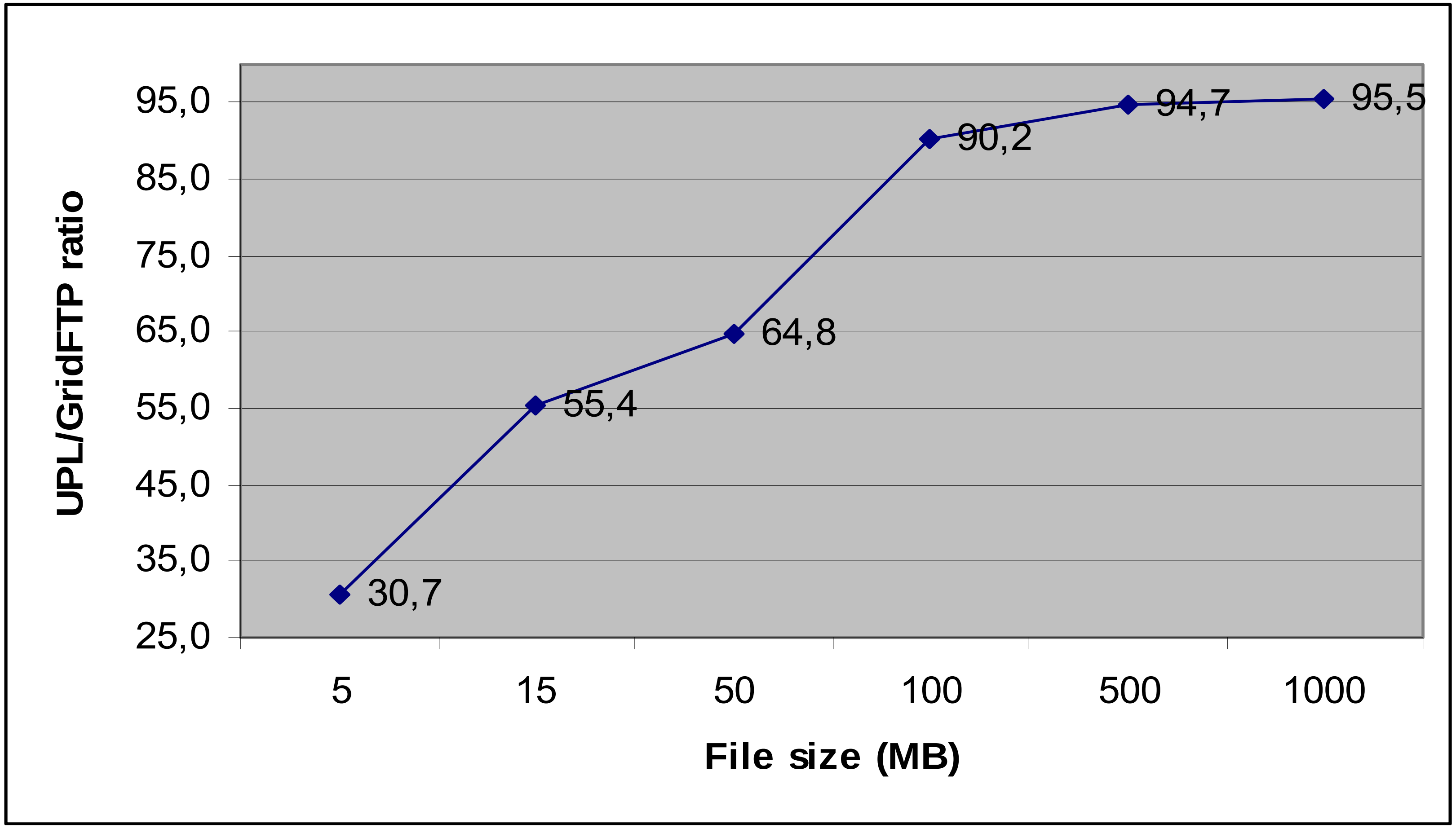
- The trend that emerges from the previous pictures is that the usage of GridFTP as an alternative to UPL is convenient even for smaller files (about 5 MB), and this convenience grows with the file size.

- For example, the transfer of a 1 GB binary file with GridFTP has proved to be 95 times faster than the UPL transfer of the same file.

- Tests over a WAN haven’t been performed yet, but it’s reasonable to expect a decrease of the performance gap between GridFTP and UPL .

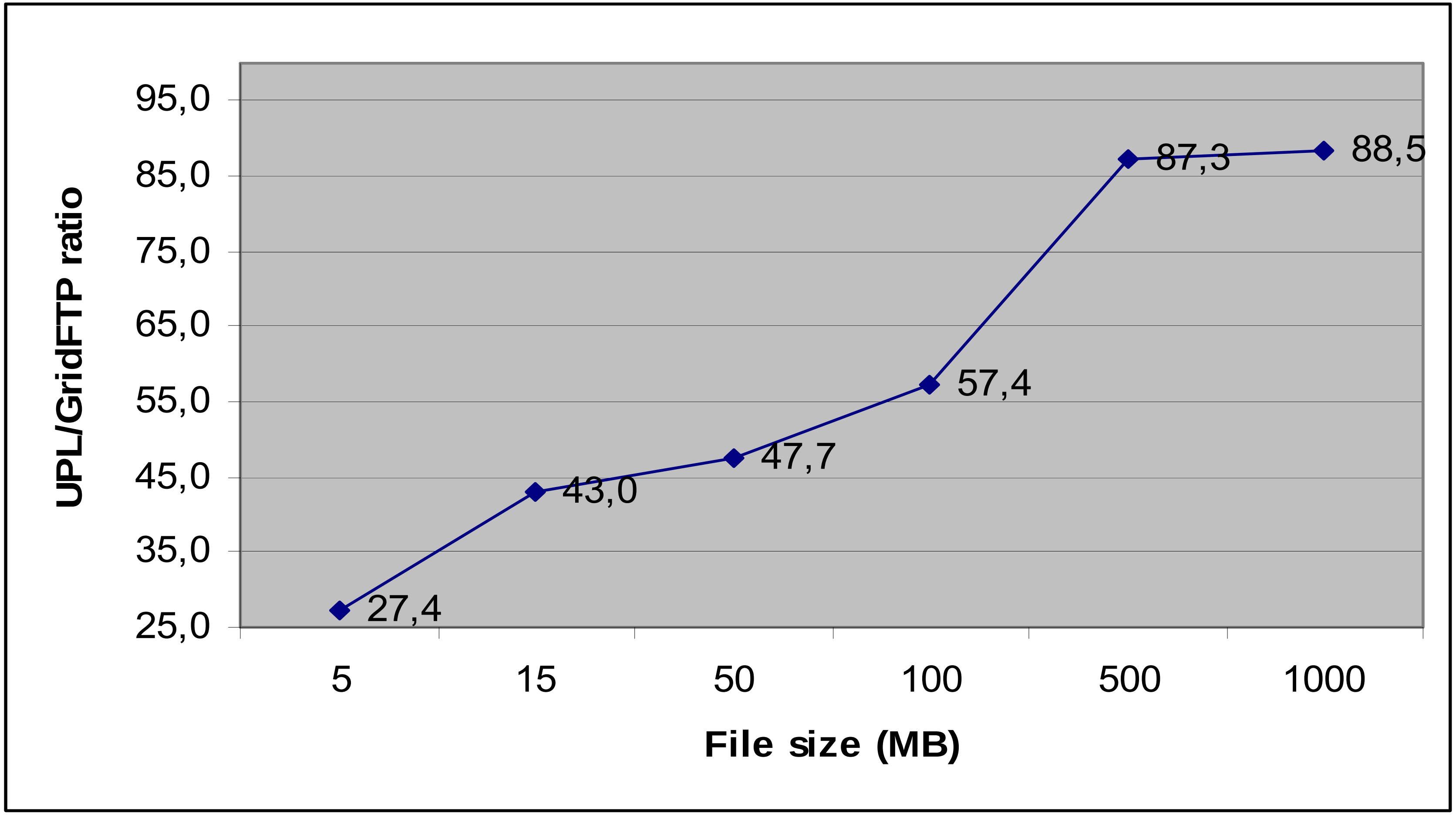
UPL vs GridFTP

- UPL/GridFTP performance ratio for binary transfers:



UPL vs GridFTP

- UPL/GridFTP performance ratio for ASCII transfers:



ARFT over “slow” connections

ARFT & compression

- Can data compression help to improve performance over "slow" connections?

- Several tests have been performed with big seismic data files (.segy files up to 20GB)

- Compression methods tested:
  - Zip
  - Gzip
  - Bzip2

- Compression levels tested: from 1 to 9

Compression tests results: ZIP &Gzip

- Zip and gzip (as expected) have similar performance
  - ♦ Average compression ratio (with segy files): 49%
  - ♦ Compression level 1 is four times faster than compression level 9, but the compression ratio is quite similar
  - ♦ The decompression time is not influenced by the compression level used before
  - ♦ Considering the overhead caused by compression and decompression, transferring a zip-compressed file with GridFTP is convenient only over networks with bandwidth <=5 Mbps

Compression tests results: bzip2

- Very good compression ratio: 60%
- But… very slow!
  - Up to 5 times slower than zip/gzip
- Transferring a bzip2-compressed file with GridFTP is convenient only over networks with bandwidth <=1 Mbps

ARFT & compression

- ARFT is able to compress and decompress files before and after the GridFTP transfer
- If enabled, compression (and decompression) is transparent to the user
- Compression method currently supported: ZIP
- Other compression mechanisms are very easy to plug

ARFT@sourceforge

- ARFT software and documentation can be downloaded here:

http://unicore.sourceforge.net

UNICORE & GridFTP:
Current and future work

What's next?

- Unicore is moving towards Web Services
- GridFTP will be integrated into UnicoreGS
- We are now working to integrate GridFTP capabilities into the UnicoreGS client
- Java COG-Kit libraries are used in order to allow to a client to move data to and from UnicoreGS TSS using GridFTP

Thank you!