

UNICORE

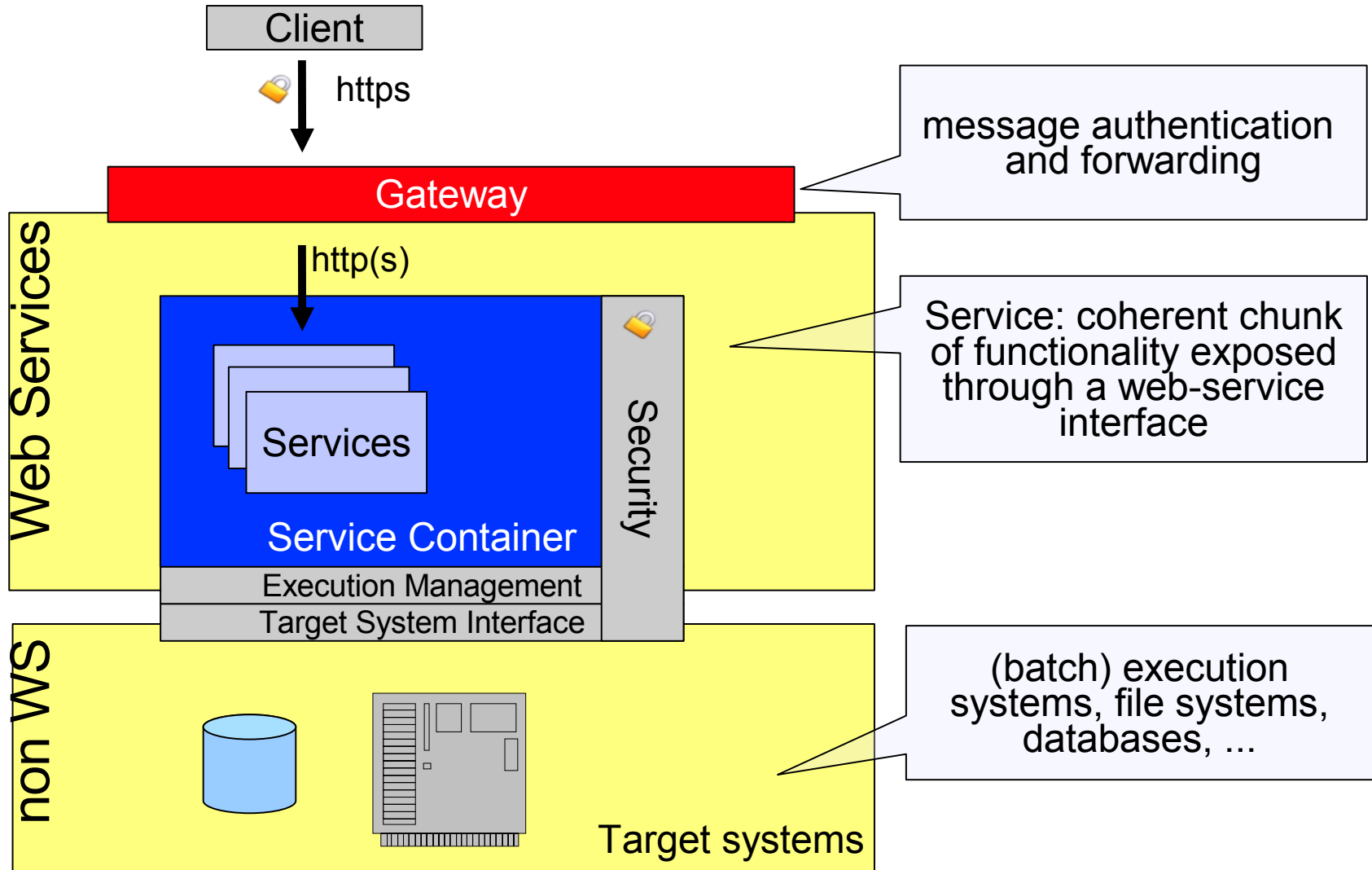
An in-depth view

Bernd Schuller, FZ Jülich

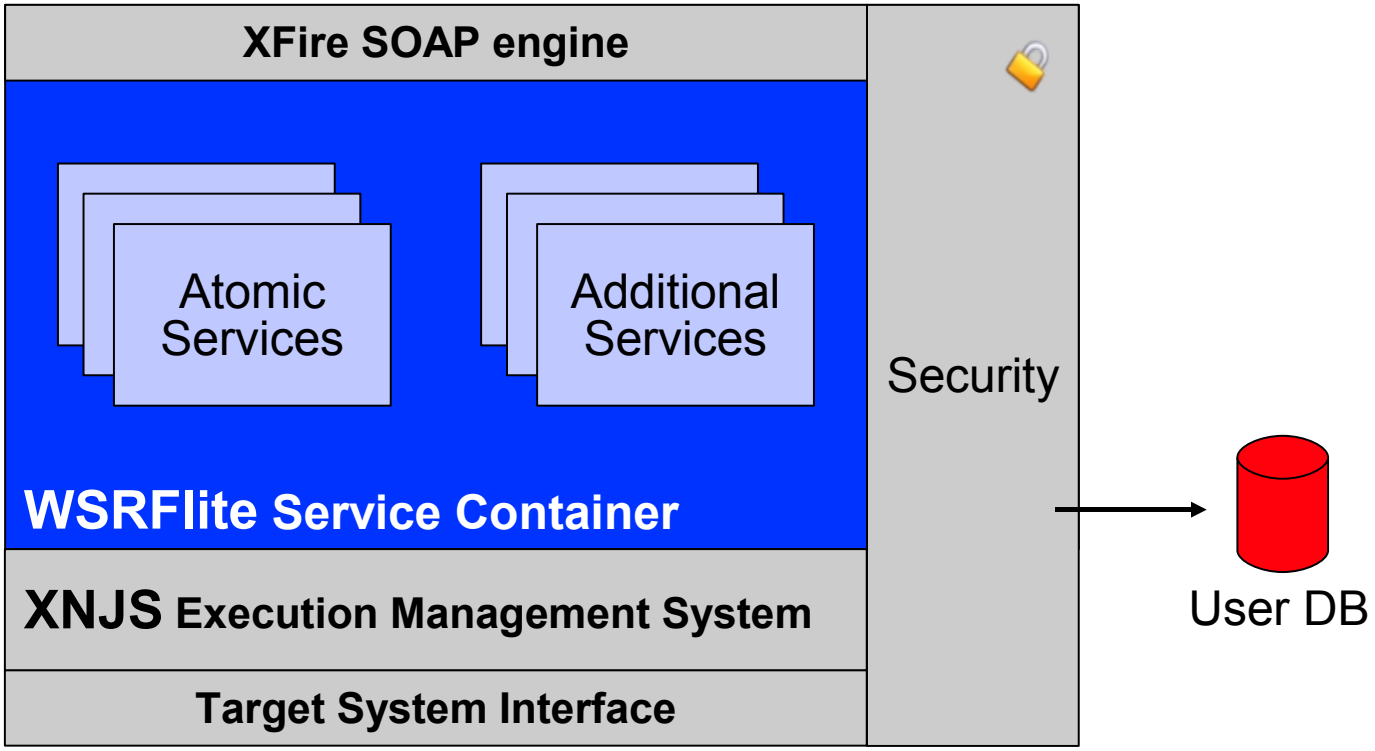
Outline

- Base services and their relationships
- XNJS Architecture
- Possible extensions

UNICORE 6 architecture

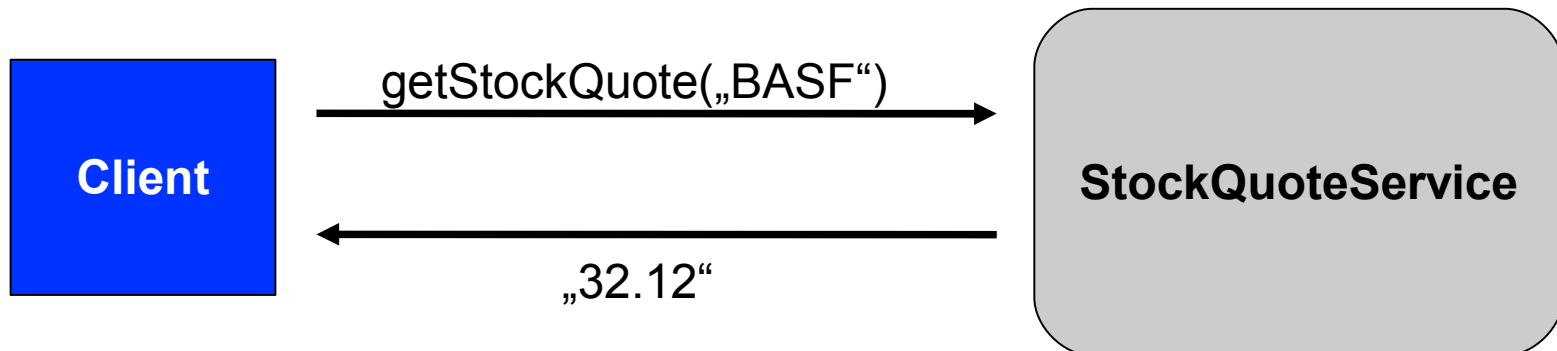


Detailed view: VSite



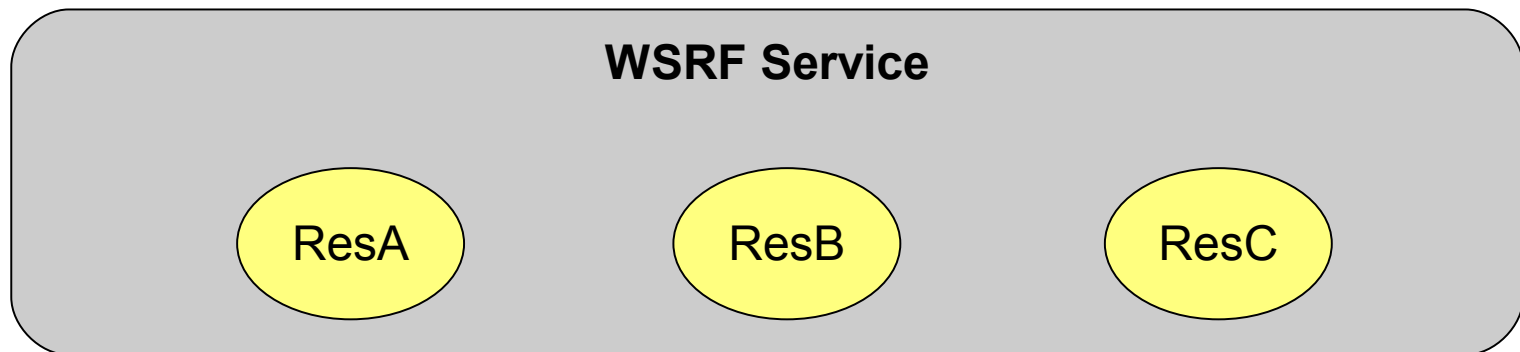
Web Services

- Web services provide basic communication paradigm:
 - exchange of XML messages (SOAP protocol)
 - contact remote service
 - e.g. <http://www.some.com/StockQuoteService>
 - provide parameters as an XML document
 - retrieve results in a XML document



WSRF

- Web service resource framework
- Allows building web services for accessing and managing multiple resources : „WS-Resources“
- WS-Resource have a state: resource properties document defined by an XML Schema

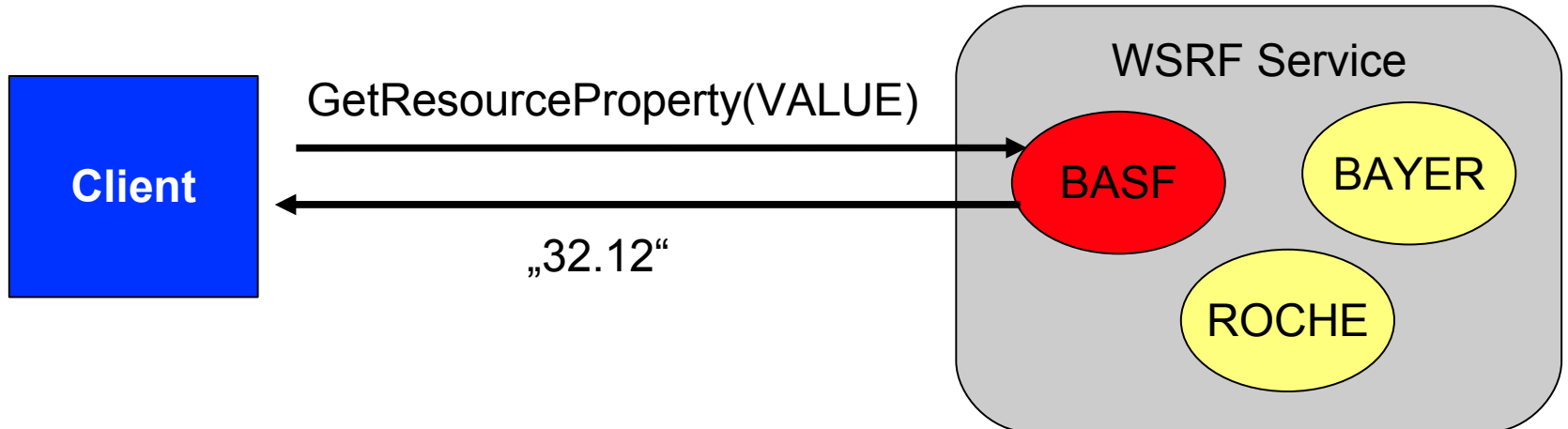


Example: Job resource properties

```
<jms:JobProperties xmlns:jms="http://unigrids.org/2006/04/services/jms"
                  xmlns:rl="http://docs.oasis-open.org/wsrp/rl-2">
  <jms:SubmissionTime>2007-07-25T15:23:29.784+02:00</jms:SubmissionTime>
  <jms:OriginalJSDL...</jms:OriginalJSDL>
  <jms:ExecutionJSDL>...</jms:ExecutionJSDL>
  <jms:Log>...</jms:Log>
  <jms:TargetSystemReference>...</jms:TargetSystemReference>
  <jms:WorkingDirectoryReference>...</jms:WorkingDirectoryReference>
  <jms:StdOut>stdout</jms:StdOut>
  <jms:StdErr>stderr</jms:StdErr>
  <typ:StatusInfo xmlns:typ="http://unigrids.org/2006/04/types">
    <typ:Status>SUCCESSFUL</typ:Status>
  </typ:StatusInfo>
  ...
  <rl:CurrentTime>2007-07-25T21:11:55.300+02:00</rl:CurrentTime>
  <rl:TerminationTime>2007-07-26T15:23:29.771+02:00</rl:TerminationTime>
</jms:JobProperties>
```

WSRF II

- Stock quote example
- Client addresses specific resource:
 - e.g. <http://some.com/StockQuoteService?resID=BASF>
 - → WS-Addressing



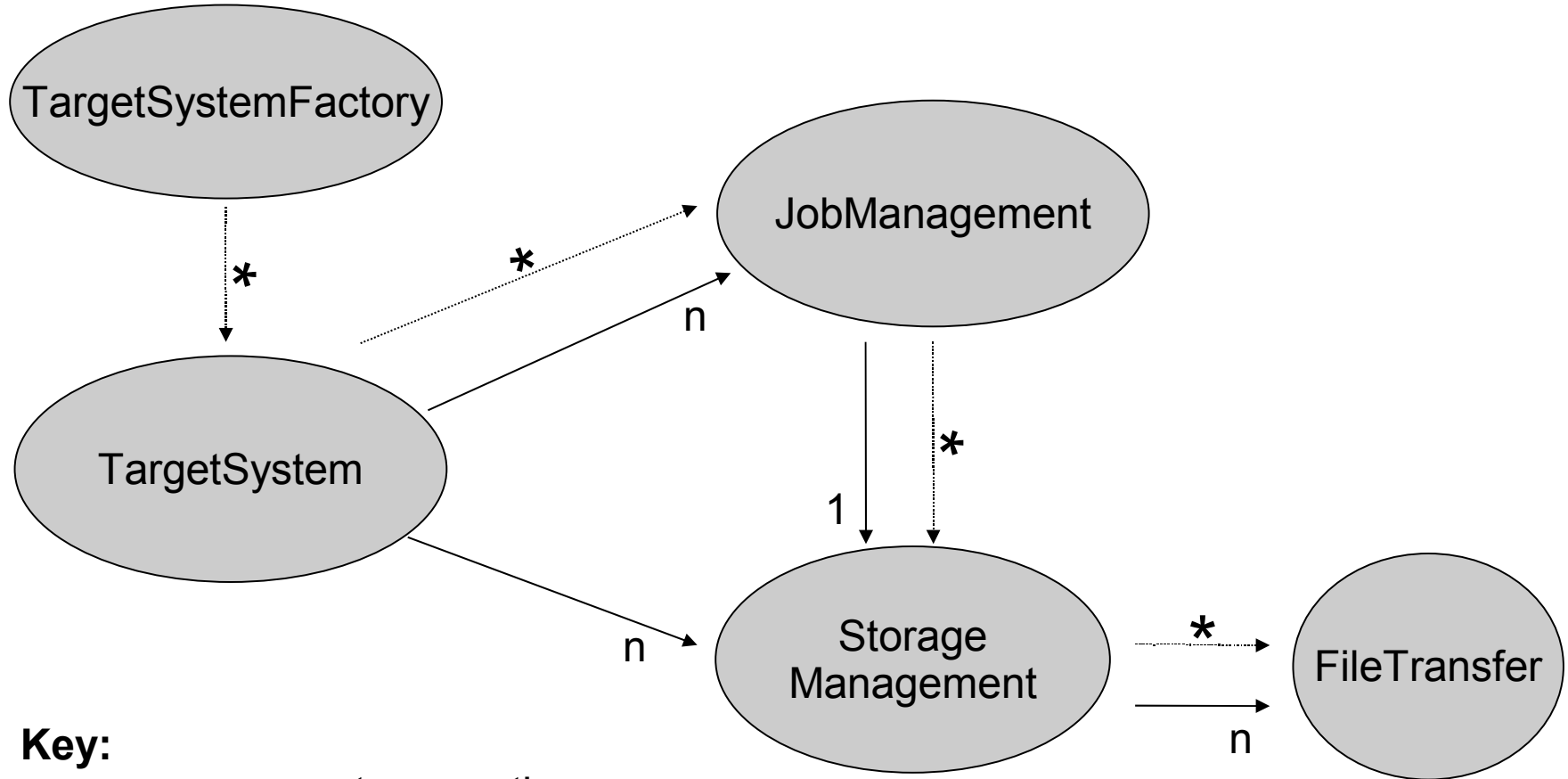
Base Services

- UNICORE Atomic Services
 - TargetSystemFactory
 - TargetSystem
 - JobManagement
 - StorageManagement
 - FileTransfer
- Registry
 - LocalRegistry
 - Registry (the shared one)

GPE Services

- Workflow
 - WorkflowTargetSystem
 - WorkflowJobManagement
- GridBeanService

Service Relationships

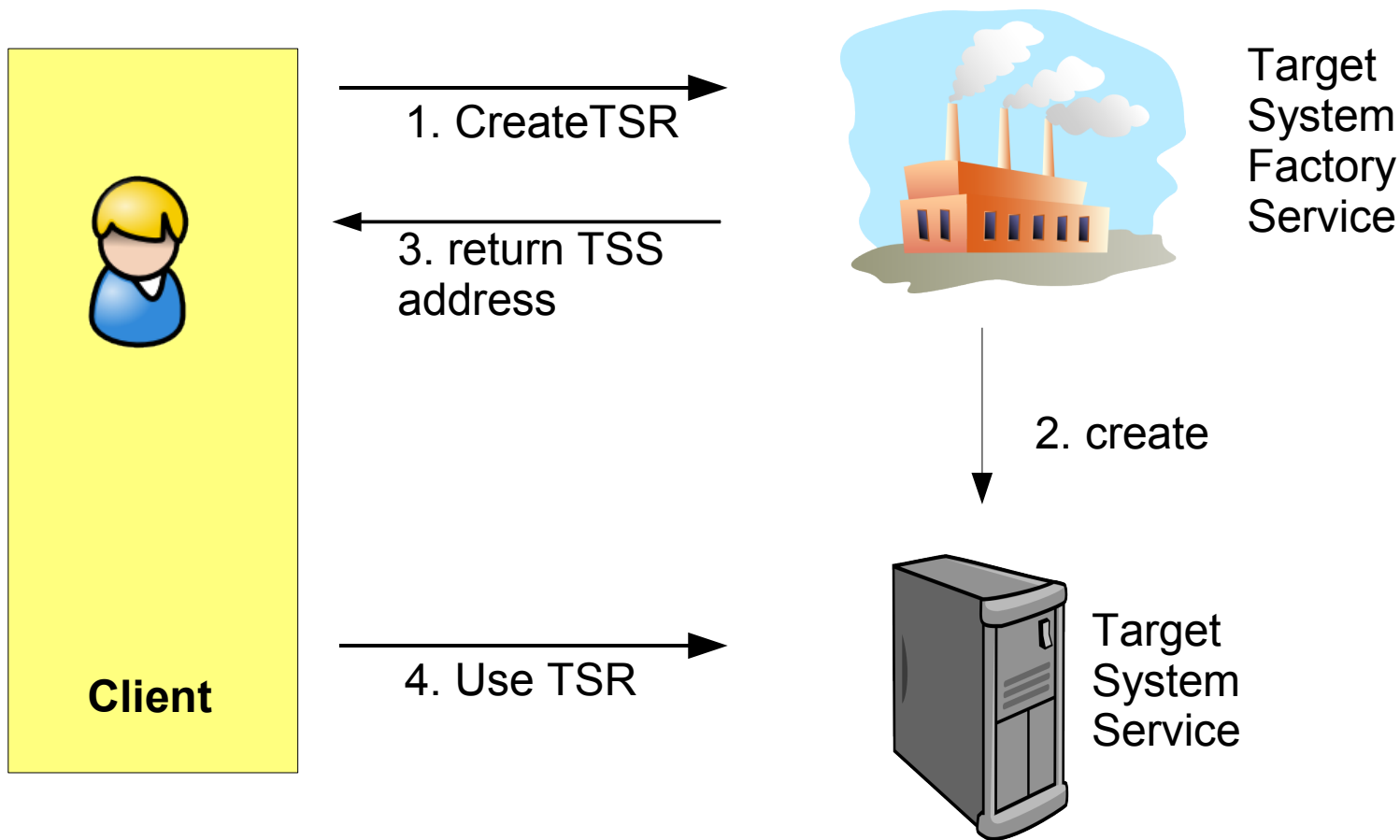


Key:

create operation

association („has“)
with multiplicity

Target system factory



TargetSystemFactory properties

```
<tsf:TargetSystemFactoryProperties
  xmlns:tsf="http://unigrids.org/2006/04/services/tsf"
  xmlns:tss="http://unigrids.org/2006/04/services/tss"
  xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl">

  <tsf:TargetSystemReference>...</tsf:TargetSystemReference>
  <tsf:TargetSystemReference>...</tsf:TargetSystemReference>
  <tsf:TargetSystemReference>...</tsf:TargetSystemReference>

  <tss:ApplicationResource>
    <jsdl:ApplicationName>Date</jsdl:ApplicationName>
    <jsdl:ApplicationVersion>1.0</jsdl:ApplicationVersion>
  </tss:ApplicationResource>
  <tss:ApplicationResource >...</tss:ApplicationResource>
  <tss:ApplicationResource >...</tss:ApplicationResource>

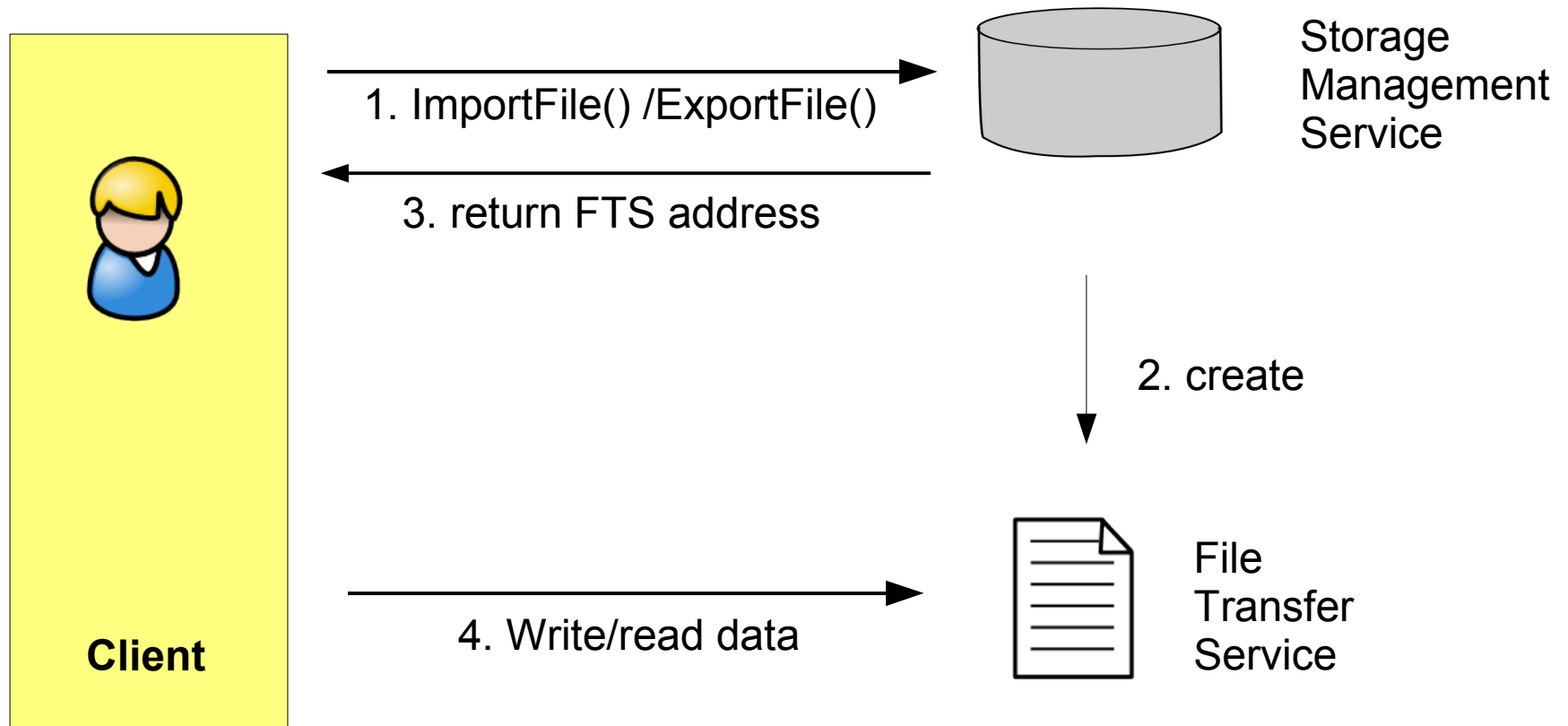
  ...
</tsf:TargetSystemFactoryProperties>
```

TargetSystem properties

```
<tss:TargetSystemProperties
  xmlns:tss="http://unigrids.org/2006/04/services/tss"
  xmlns:typ="http://unigrids.org/2006/04/types">
  <tss:Name>DEMO-SITE</tss:Name>
  <tss:TotalNumberOfJobs>4</tss:TotalNumberOfJobs>
  <tss:JobReference>...</tss:JobReference>
  <jSDL:IndividualPhysicalMemory...</jSDL:IndividualPhysicalMemory>
  <tss:ApplicationResource>...</tss:ApplicationResource>
  <typ:Processor >...</typ:Processor>
  <typ:StorageReference xmlns:typ="http://unigrids.org/2006/04/types">
    <typ:Type>Home</typ:Type>
    <typ:StorageEndpointReference>... </typ:StorageEndpointReference>
  </typ:StorageReference>

  ...
  <jSDL:IndividualCPUTime ... </jSDL:IndividualCPUTime>
  <jSDL:TotalResourceCount ... </jSDL:TotalResourceCount>
</tss:TargetSystemProperties>
```

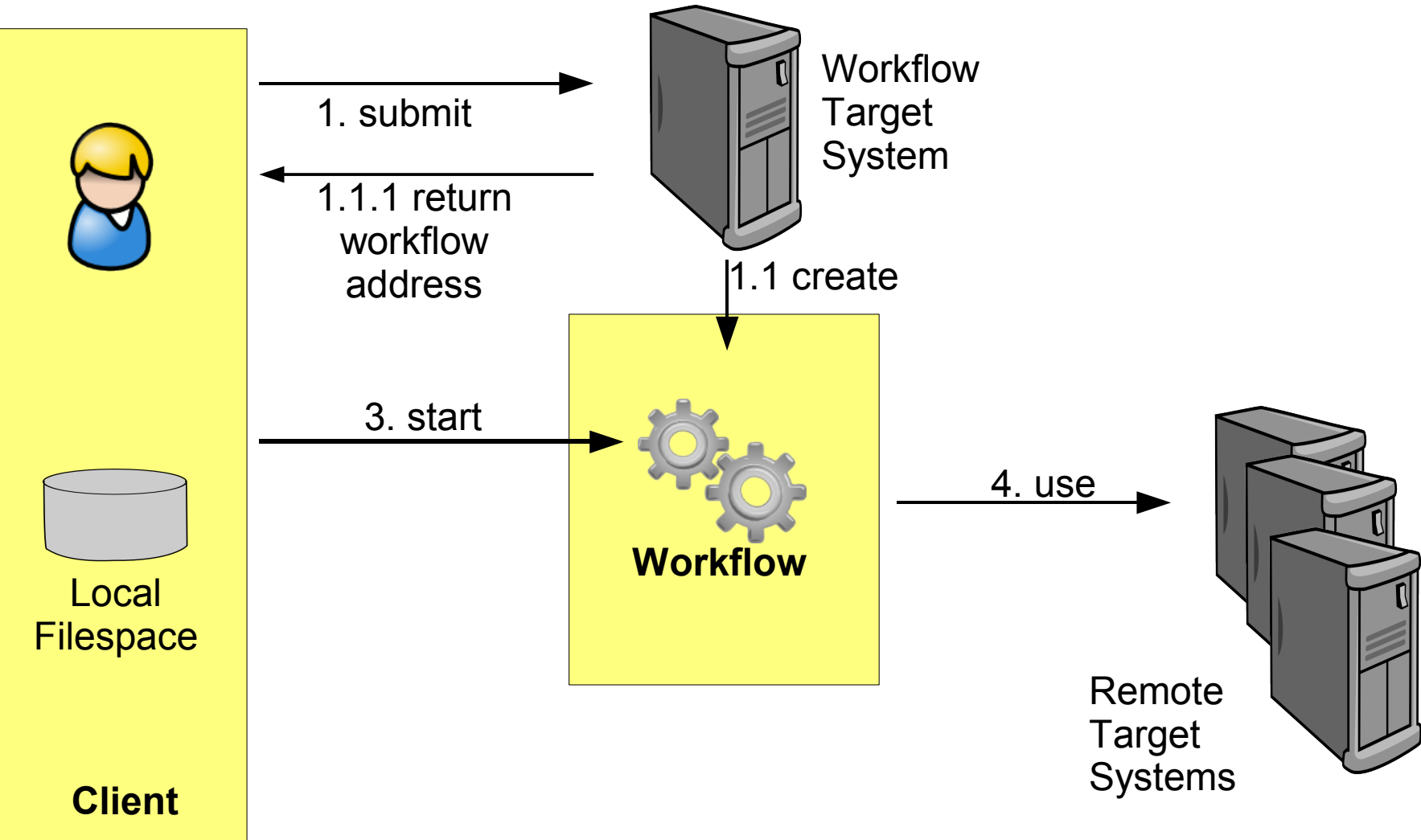
Storage and File Transfer



Intel GPE Workflow service

- Basic idea: Client creates „low-level“ BPEL description from „high-level“ graph
- „low-level“: BPEL contains detailed specification of job execution
 - submit job, wait until ready, start, ...
- BPEL is submitted to server and executed by a (homemade) BPEL engine

BPEL workflows



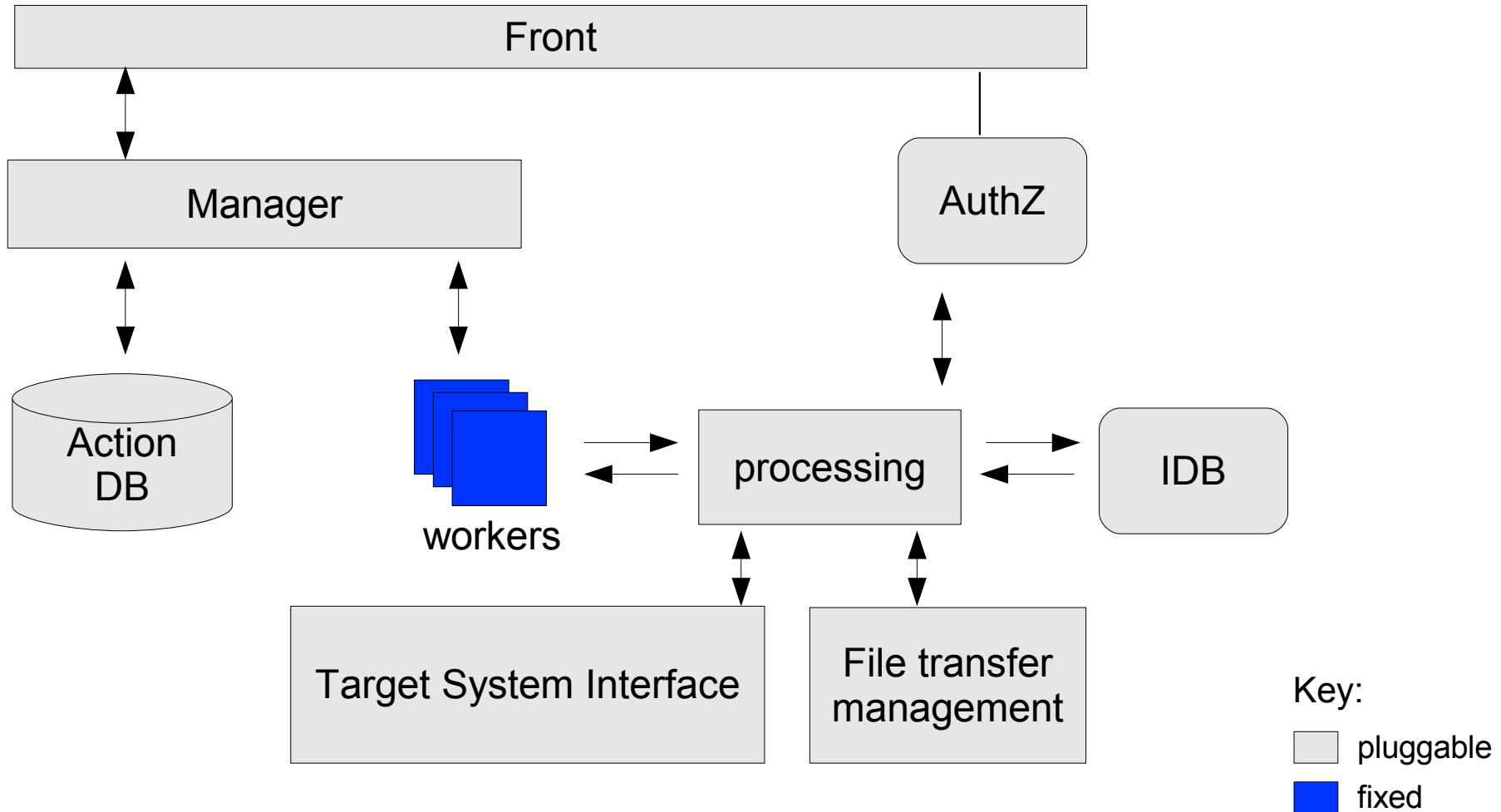
UNICORE 6 Extension points

- Add new services
- Extend existing services with new capabilities
 - storage, filetransfer
- Replace existing service implementations by custom implementations (→wsrflite.xml config file)
- Security
 - SOAP handler chain
 - Modify policy decision
 - XUADB lookup
- Reconfigure / extend the XNJS

XNJS

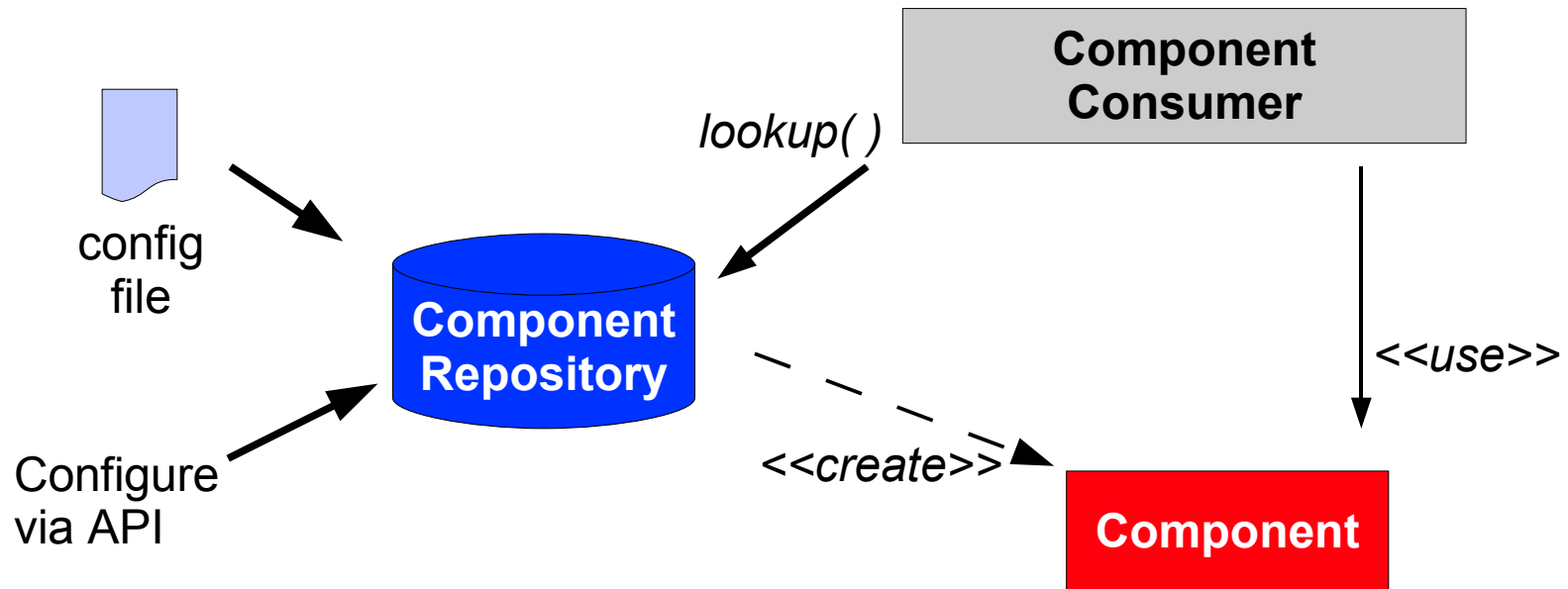
- Execution management system
- Deals with jobs and file access
- Bridge between WS world and target system

Overall architecture



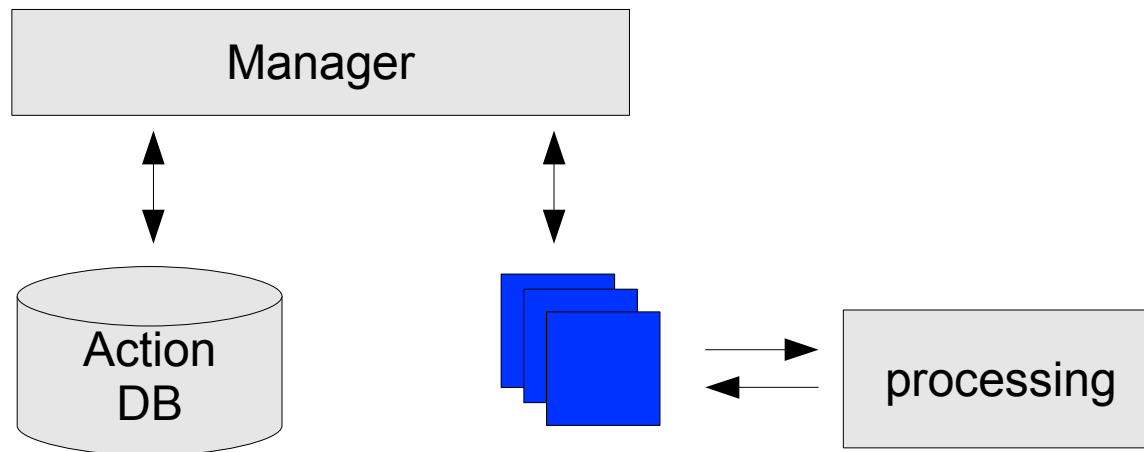
Modularisation

- Use a component repository
 - Components lookup other components *by interface*
- Concrete system configuration defined in a config file (→xnjs.xml)



Actions

- Actions are the things that the XNJS processes
 - Description (any XML)
 - Status
 - Unique ID
 - Client (user and security information)

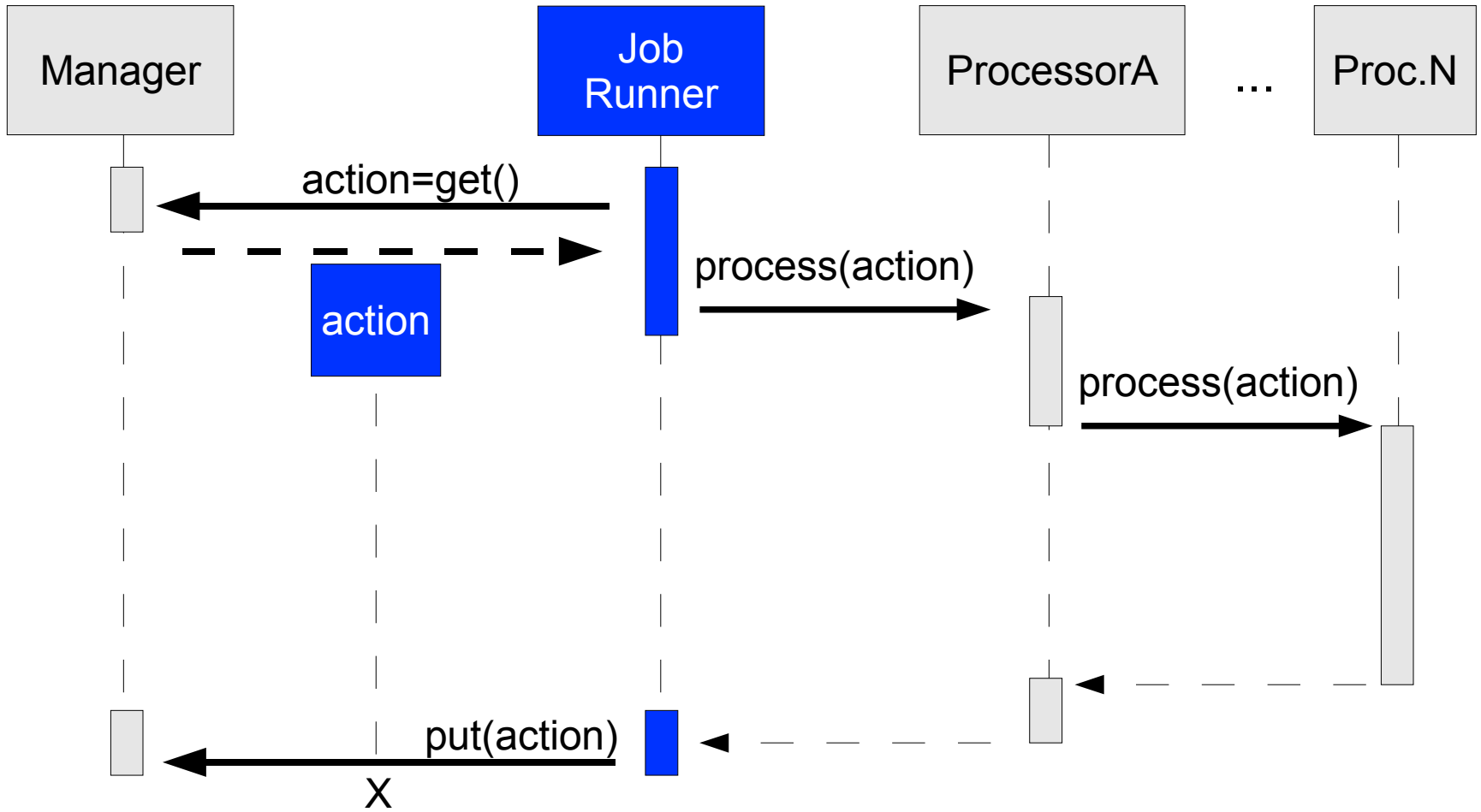


Action types

- In UNICORE 6:
 - JSDL, Data staging
- New action types can be added easily:
 - Add code to process the action
 - Re-configure the XNJS

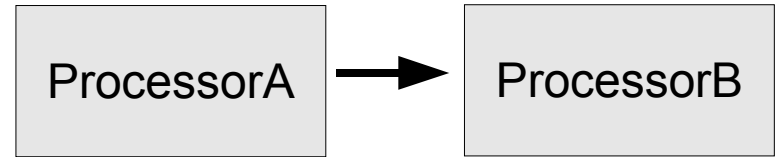
```
<!-- processing -->
<eng:ProcessingChain actionType="JSDL"
jobDescriptionType="{http://schemas.ggf.org/jSDL/2005/11/jSDL}JobDefinition">
  <eng:Processor>de.fzj.unicore.xnjs.jSDL.JSDLProcessor</eng:Processor>
</eng:ProcessingChain>
<eng:ProcessingChain actionType="JSDL_STAGEIN">
  <eng:Processor>de.fzj.unicore.uas.xnjs.Unicore6FileTransferProcessor</eng:Processor>
</eng:ProcessingChain>
<eng:ProcessingChain actionType="JSDL_STAGEOUT">
  <eng:Processor>de.fzj.unicore.uas.xnjs.Unicore6FileTransferProcessor</eng:Processor>
</eng:ProcessingChain>
```

Processing



Extensions of XNJS

- Processing chains are configurable per action type



- New action types can be added without changing the XNJS core
 - Need to add new Processor implementations
 - Edit config file
 - In principle even at runtime

Example: extending the processing chain

- Goal: notification when a job has finished
- Simple solution: on XNJS level

```
<!-- processing -->
<eng:ProcessingChain actionType="JSDL"
  jobDescriptionType="{http://schemas.ggf.org/jsdl/2005/11/jsdl}JobDefinition">
  <eng:Processor>de.fzj.unicore.xnjs.jsdl.JSDLProcessor</eng:Processor>
  <eng:Processor>
    de.fzj.unicore.xnjs.ems.processors.NotifyUserWhenDoneProcessor
  </eng:Processor>
</eng:ProcessingChain>
```

Code example

```
package de.fzj.unicore.xnjs.ems.processors;

import javax.swing.JOptionPane;
import de.fzj.unicore.xnjs.Configuration;
import de.fzj.unicore.xnjs.ems.ActionStatus;

public class NotifyUserWhenDoneProcessor extends DefaultProcessor {

    public NotifyUserWhenDoneProcessor(Configuration config){
        super(config);
    }

    /*
    * if done, show a Swing dialog
    */
    protected void done() {
        if(action.getStatus()==ActionStatus.DONE){
            try{
                String msg="Action "+action.getUUID()+" has finished";
                JOptionPane.showMessageDialog(null, msg);
            }catch(Exception ex){}
        }
    }
}
```