CAFE: a Collaborative Analysis Framework for distributed Environmental data

Hao Xu
On behalf of the CAFE Team

Dec.7th 2016

Outline

01 CAFE Introduction

02 System Design

- ✓ Logical design
- ✓ Modularity
- ✓ Extensibility

03 Prototype and Use Case

04 Conclusions

Introduction-about CAFE

CAFE is a dedicated software package for **collaborative analysis** of large volumes of **distributed environmental data.**

Key features:

1) Computing near the data;
2) data is logically grouped, while physically distributed;
3) Analytic tasks are divided as subtasks, and then fulfilled on corresponding nodes;
4) Easy way to enrich the built-in analytic functions;
5) Open source projects on github

Introduction-why CAFE?

Typical workflow

❖ *Download data from multiple nodes*

❖ *Data subsetting*

❖ *Data regridding*

❖ *Data averaging*

❖ *Writing analytic scripts*

❖ *Executing codes locally*

❖ *Result visulization*

❖ *Result analysis*

❖ ……

CAFE Features

❖ *Web-based UI provided*

❖ *REST APIs availbale*

❖ *One-stop service for data discovery, visualization, and analysis*

❖ *User-transparent*

❖ *Task management*

❖ *Multi-node collaboration*

❖ *Support for data intercomparison*

❖ *Multiple built-in analytic functions*

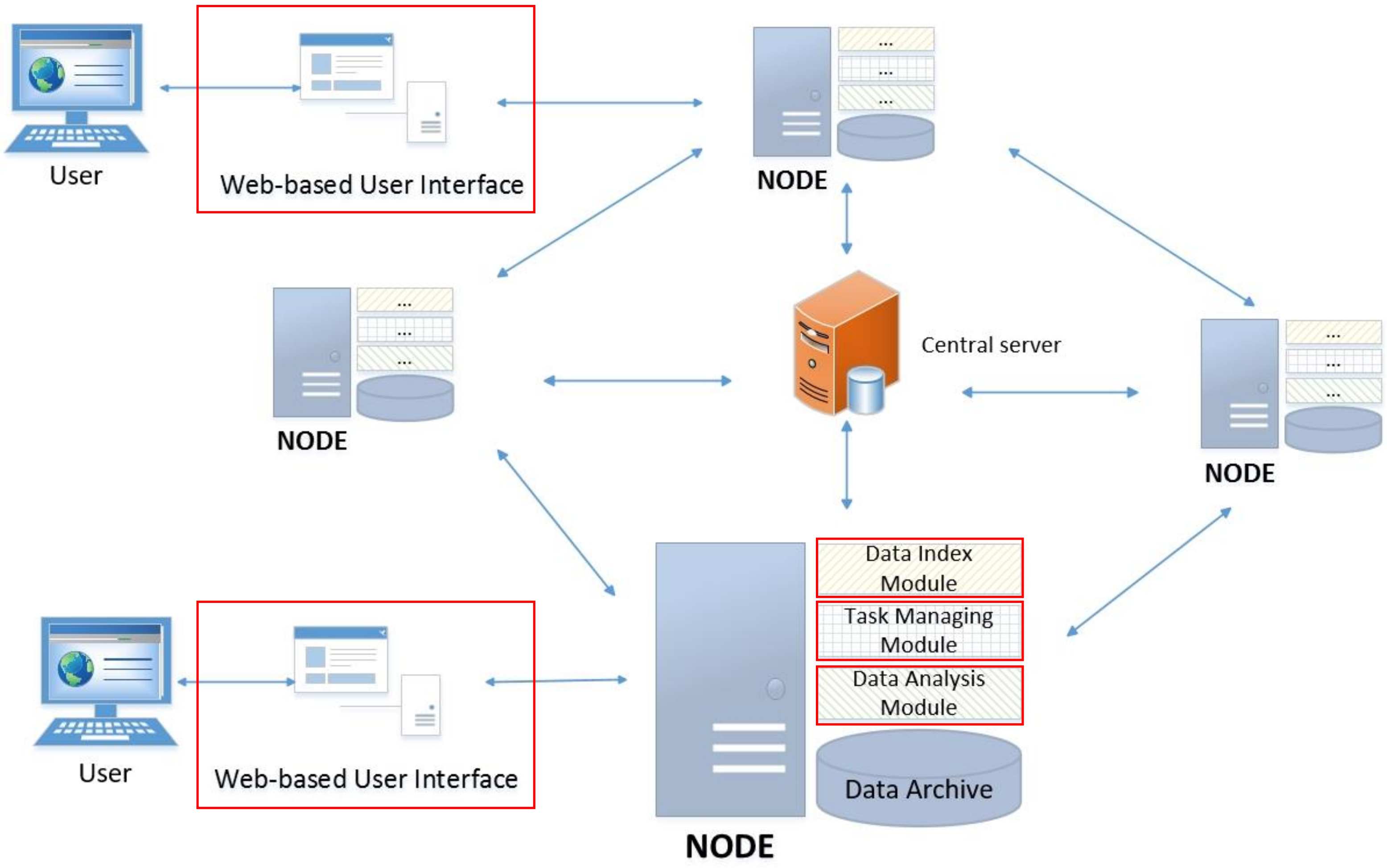
❖ *Easy way for extensions*

❖ *Batch analysis*

……

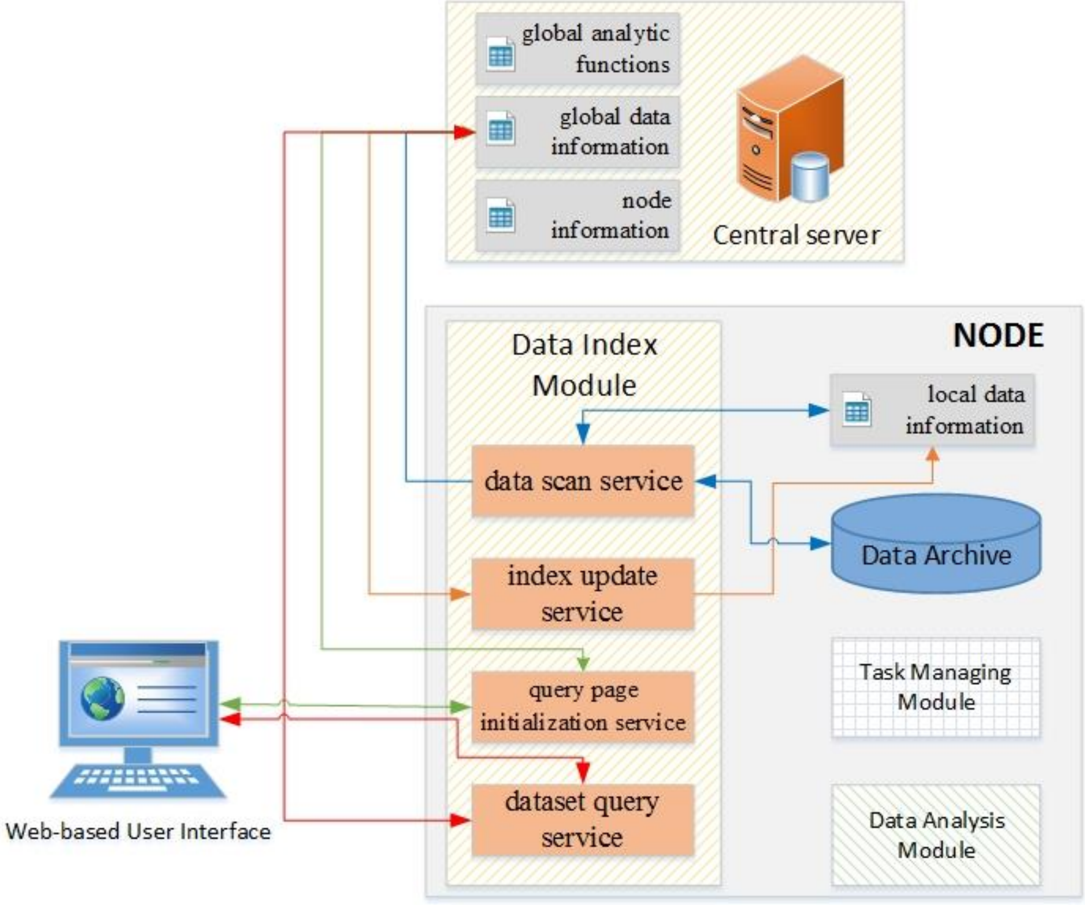
System design-logical design

A p2p architecture

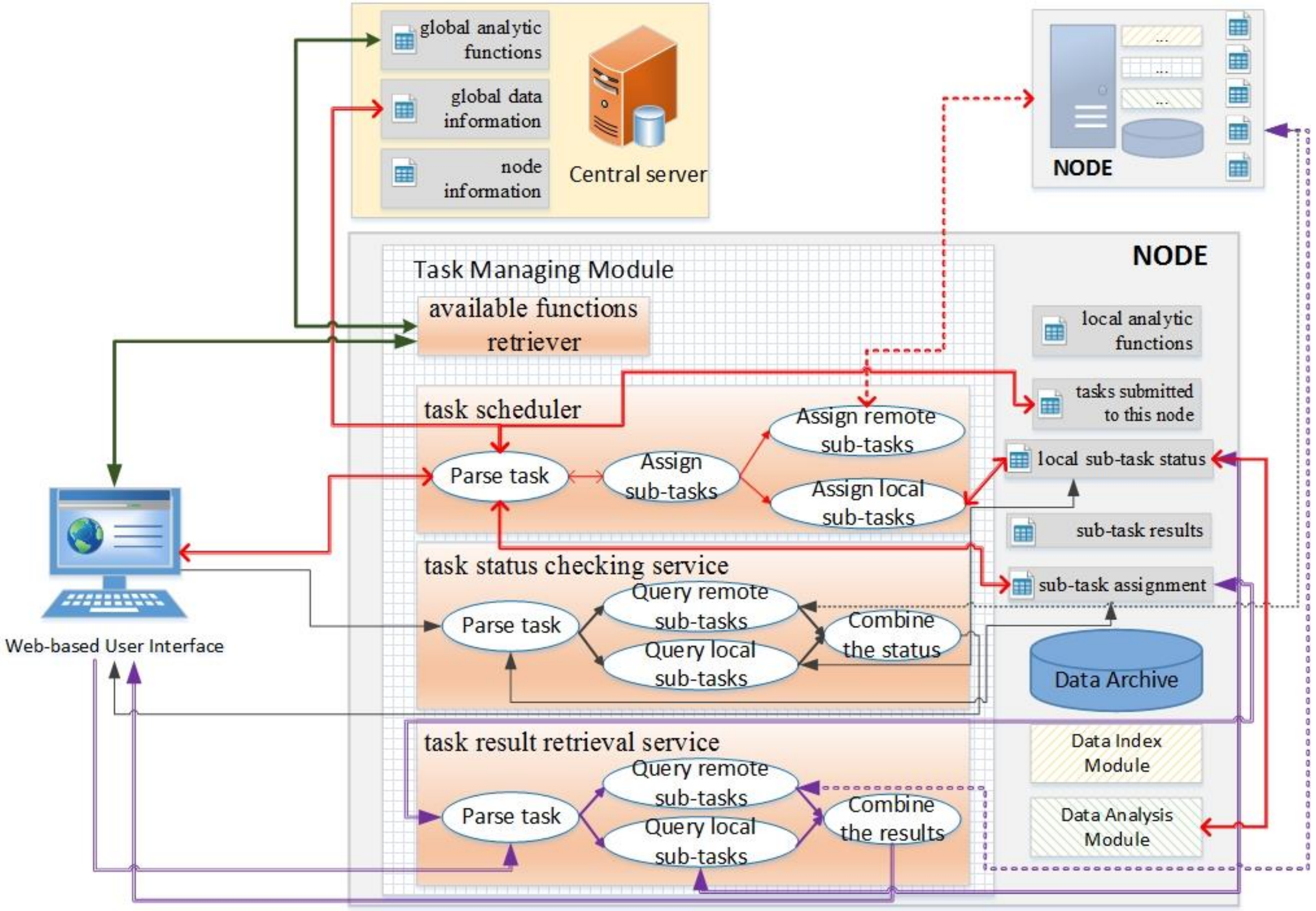


System design-modularity

Data Index Module

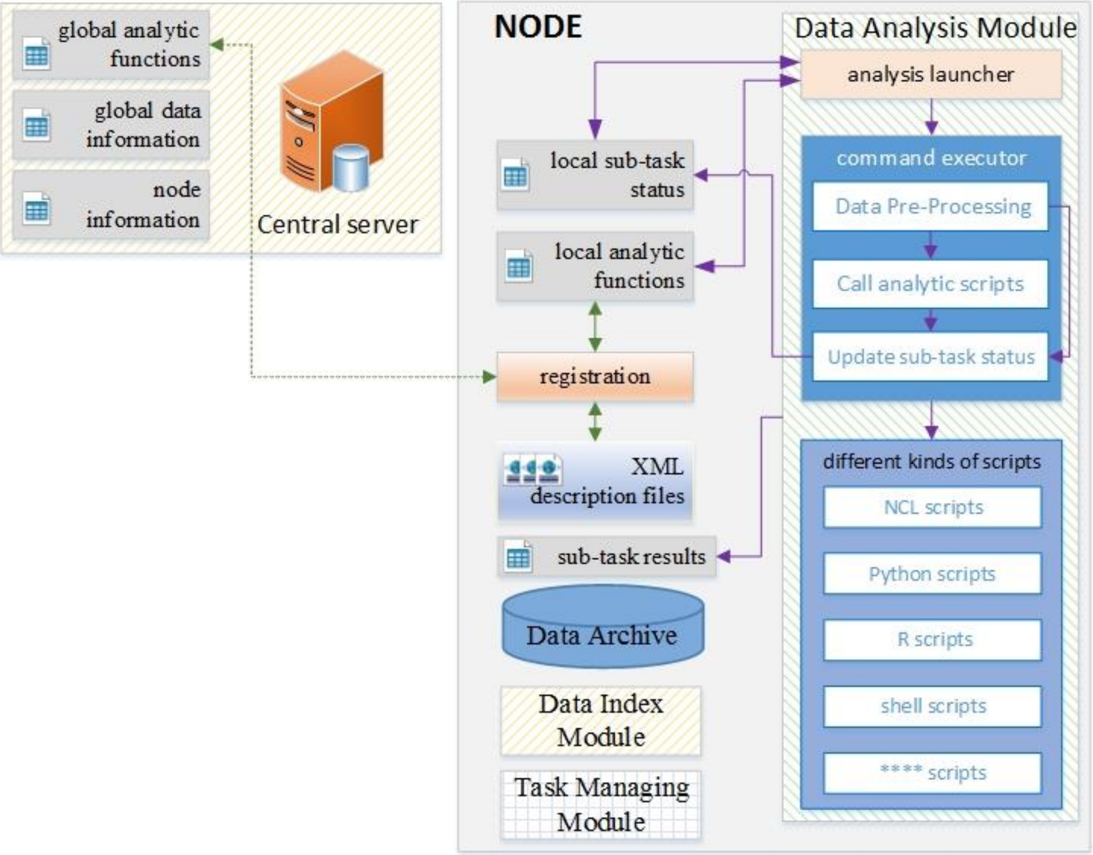


Task Managing Module

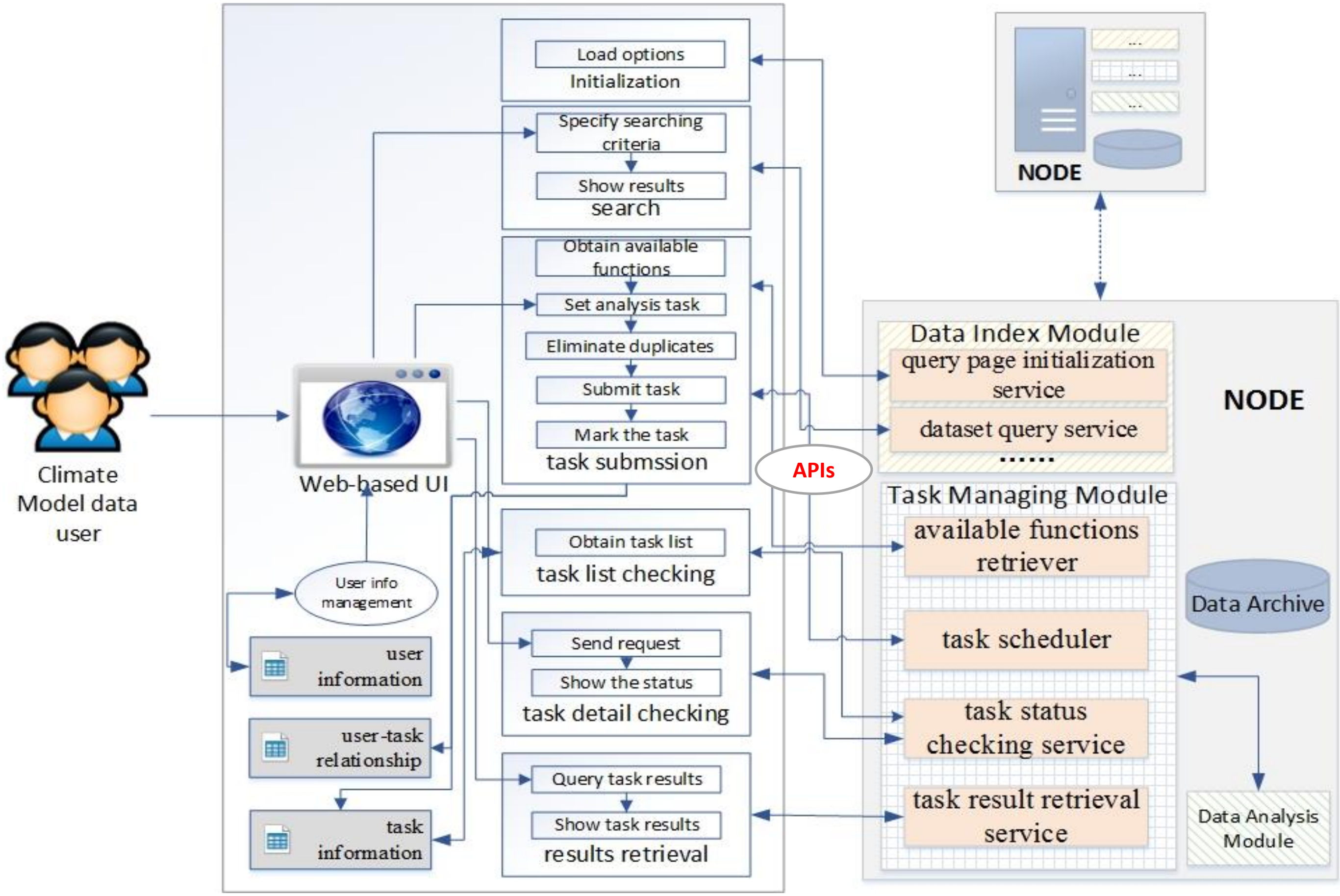


System design-modularity

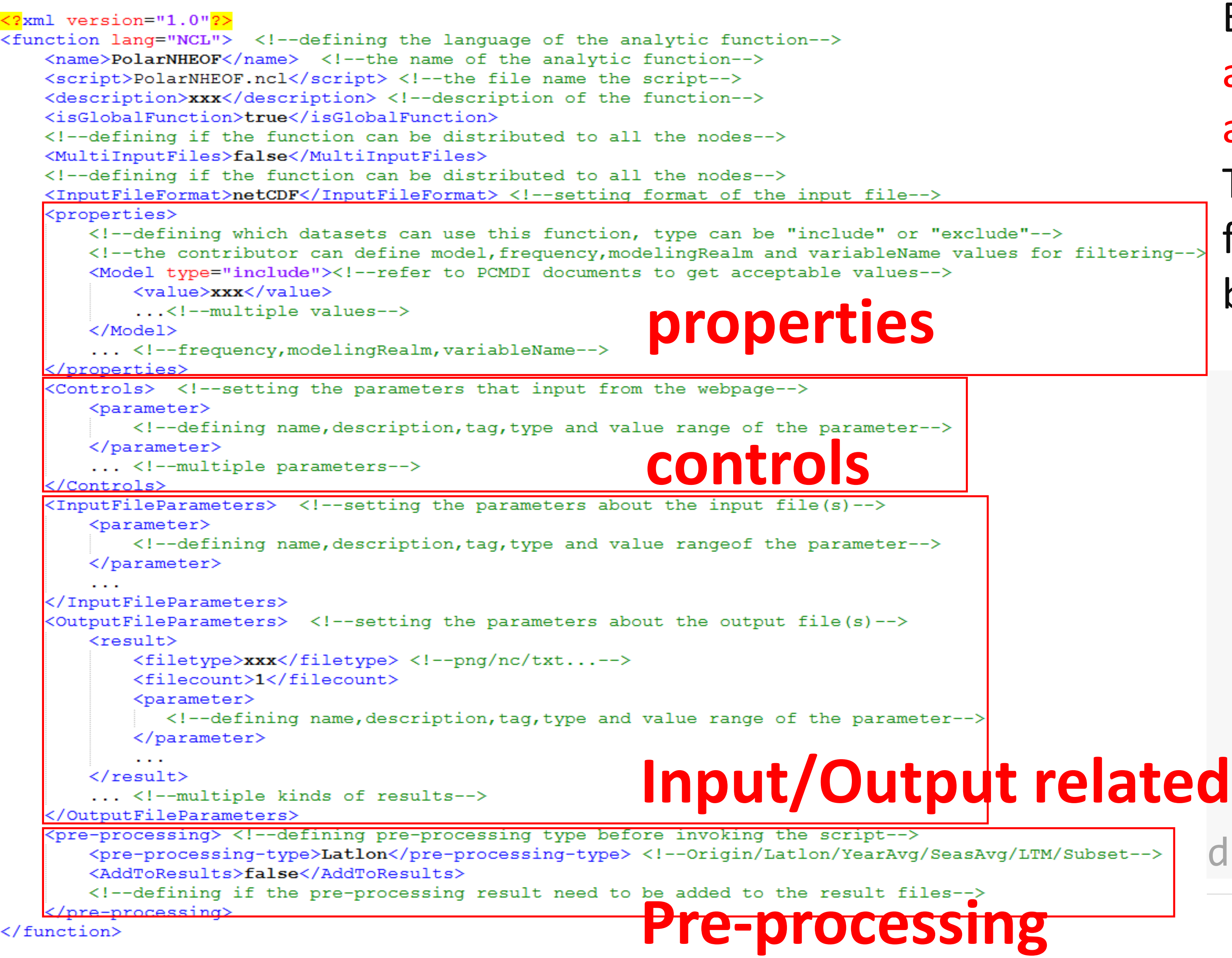
Data Analysis Module



Web-based User Interface



System design-extensibility



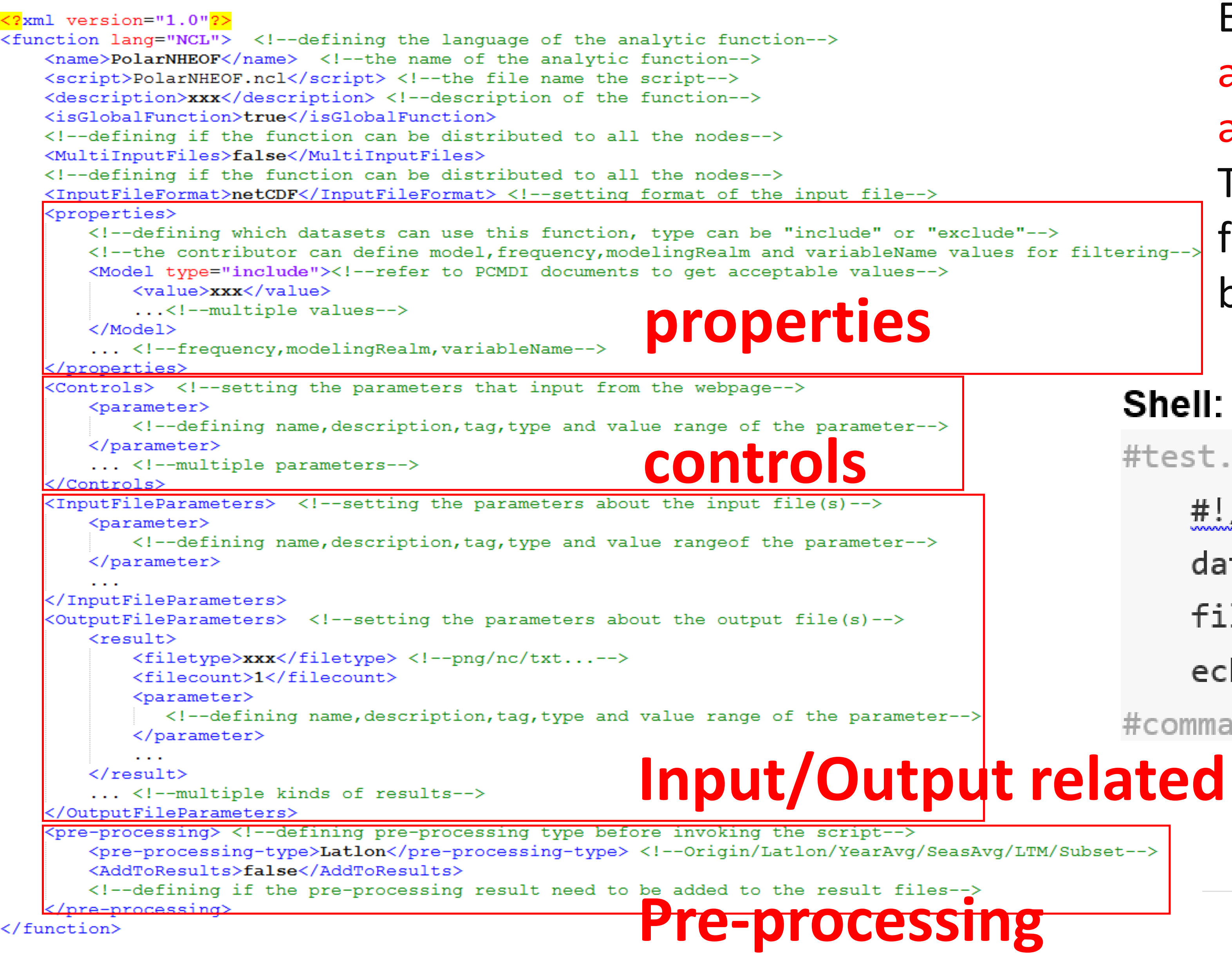
```
<?xml version="1.0"?>
<function lang="NCL">  <!--defining the language of the analytic function-->
    <name>PolarNHEOF</name>  <!--the name of the analytic function-->
    <script>PolarNHEOF.ncl</script> <!--the file name the script-->
    <description>xxx</description> <!--description of the function-->
    <isGlobalFunction>true</isGlobalFunction>
    <!--defining if the function can be distributed to all the nodes-->
    <MultiInputFiles>false</MultiInputFiles>
    <!--defining if the function can be distributed to all the nodes-->
    <InputFileFormat>netCDF</InputFileFormat> <!--setting format of the input file-->
    <properties>
        <!--defining which datasets can use this function, type can be "include" or "exclude"-->
        <!--the contributor can define model,frequency,modelingRealm and variableName values for filtering-->
        <Model type="include"><!--refer to PCMDI documents to get acceptable values-->
            <value>xxx</value>
            ...<!--multiple values-->
        </Model>
        ... <!--frequency,modelingRealm,variableName-->
    </properties>
    <Controls>  <!--setting the parameters that input from the webpage-->
        <parameter>
            <!--defining name,description,tag,type and value range of the parameter-->
        </parameter>
        ... <!--multiple parameters-->
    </Controls>
    <InputFileParameters>  <!--setting the parameters about the input file(s)-->
        <parameter>
            <!--defining name,description,tag,type and value rangeof the parameter-->
        </parameter>
        ...
    </InputFileParameters>
    <OutputFileParameters>  <!--setting the parameters about the output file(s)-->
        <result>
            <filetype>xxx</filetype> <!--png/nc/txt...-->
            <filecount>1</filecount>
            <parameter>
               <!--defining name,description,tag,type and value range of the parameter-->
            </parameter>
            ...
        </result>
        ... <!--multiple kinds of results-->
    </OutputFileParameters>
    <pre-processing> <!--defining pre-processing type before invoking the script-->
        <pre-processing-type>Latlon</pre-processing-type> <!--Origin/Latlon/YearAvg/SeasAvg/LTM/Subset-->
        <AddToResults>false</AddToResults>
        <!--defining if the pre-processing result need to be added to the result files-->
    </pre-processing>
</function>
```

Easy way for extensions. Only the analytic script and its XML description are needed.

The script should have an I/O interface for command line and can be invoked by Java.

```
#Example: for NCL
if(.not.isvar("date")) then
   date=19000101
end if
if(.not.isvar("filename")) then
    filename="test.nc"
end if   print(date)
print(filename)
#command invoking: ncl test.ncl
date=19000120'filename="test1.nc"'
```

System design-extensibility

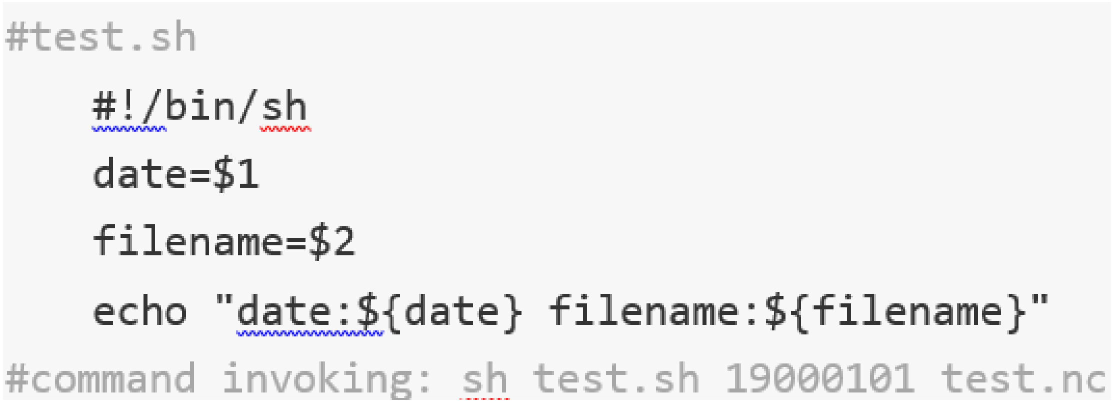


```
<?xml version="1.0"?>
<function lang="NCL">  <!--defining the language of the analytic function-->
    <name>PolarNHEOF</name>  <!--the name of the analytic function-->
    <script>PolarNHEOF.ncl</script> <!--the file name the script-->
    <description>xxx</description> <!--description of the function-->
    <isGlobalFunction>true</isGlobalFunction>
    <!--defining if the function can be distributed to all the nodes-->
    <MultiInputFiles>false</MultiInputFiles>
    <!--defining if the function can be distributed to all the nodes-->
    <InputFileFormat>netCDF</InputFileFormat> <!--setting format of the input file-->
    <properties>
        <!--defining which datasets can use this function, type can be "include" or "exclude"-->
        <!--the contributor can define model,frequency,modelingRealm and variableName values for filtering-->
        <Model type="include"><!--refer to PCMDI documents to get acceptable values-->
            <value>xxx</value>
            ...<!--multiple values-->
        </Model>
        ... <!--frequency,modelingRealm,variableName-->
    </properties>
    <Controls>  <!--setting the parameters that input from the webpage-->
        <parameter>
            <!--defining name,description,tag,type and value range of the parameter-->
        </parameter>
        ... <!--multiple parameters-->
    </Controls>
    <InputFileParameters>  <!--setting the parameters about the input file(s)-->
        <parameter>
            <!--defining name,description,tag,type and value rangeof the parameter-->
        </parameter>
        ...
    </InputFileParameters>
    <OutputFileParameters>  <!--setting the parameters about the output file(s)-->
        <result>
            <filetype>xxx</filetype> <!--png/nc/txt...-->
            <filecount>1</filecount>
            <parameter>
               <!--defining name,description,tag,type and value range of the parameter-->
            </parameter>
            ...
        </result>
        ... <!--multiple kinds of results-->
    </OutputFileParameters>
    <pre-processing> <!--defining pre-processing type before invoking the script-->
        <pre-processing-type>Latlon</pre-processing-type> <!--Origin/Latlon/YearAvg/SeasAvg/LTM/Subset-->
        <AddToResults>false</AddToResults>
        <!--defining if the pre-processing result need to be added to the result files-->
    </pre-processing>
</function>
```

Easy way for extensions. Only the analytic script and its XML description are needed.

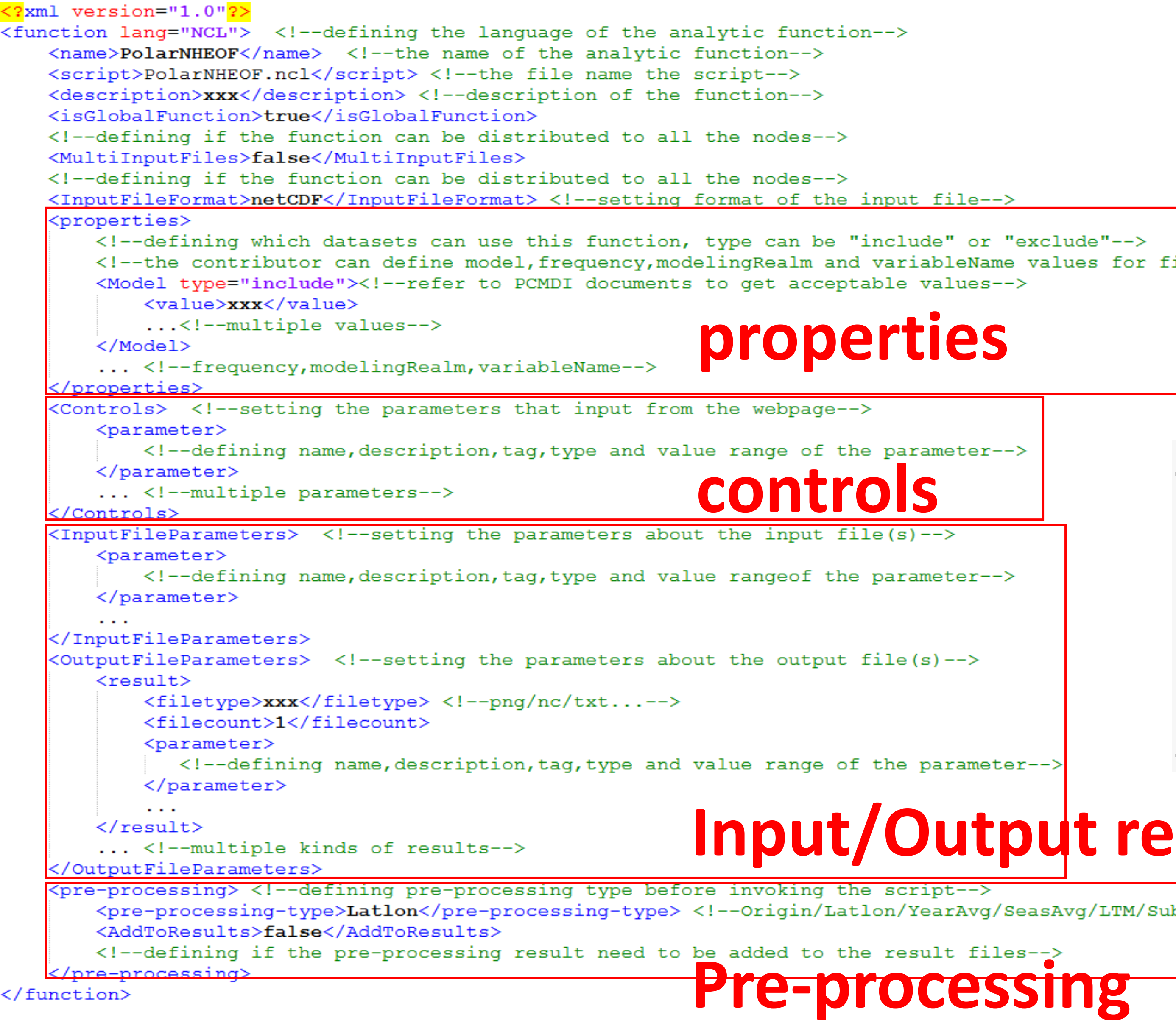
The script should have an I/O interface for command line and can be invoked by Java.

Shell:



```
#test.sh
    #!/bin/sh
    date=$1
    filename=$2
    echo "date:${date} filename:${filename}"
#command invoking: sh test.sh 19000101 test.nc
```

System design-extensibility



```
<?xml version="1.0"?>
<function lang="NCL">  <!--defining the language of the analytic function-->
    <name>PolarNHEOF</name>  <!--the name of the analytic function-->
    <script>PolarNHEOF.ncl</script> <!--the file name the script-->
    <description>xxx</description> <!--description of the function-->
    <isGlobalFunction>true</isGlobalFunction>
    <!--defining if the function can be distributed to all the nodes-->
    <MultiInputFiles>false</MultiInputFiles>
    <!--defining if the function can be distributed to all the nodes-->
    <InputFileFormat>netCDF</InputFileFormat> <!--setting format of the input file-->
    <properties>
        <!--defining which datasets can use this function, type can be "include" or "exclude"-->
        <!--the contributor can define model,frequency,modelingRealm and variableName values for filtering-->
        <Model type="include"><!--refer to PCMDI documents to get acceptable values-->
            <value>xxx</value>
            ...<!--multiple values-->
        </Model>
        ... <!--frequency,modelingRealm,variableName-->
    </properties>
    <Controls>  <!--setting the parameters that input from the webpage-->
        <parameter>
            <!--defining name,description,tag,type and value range of the parameter-->
        </parameter>
        ... <!--multiple parameters-->
    </Controls>
    <InputFileParameters>  <!--setting the parameters about the input file(s)-->
        <parameter>
            <!--defining name,description,tag,type and value rangeof the parameter-->
        </parameter>
        ...
    </InputFileParameters>
    <OutputFileParameters>  <!--setting the parameters about the output file(s)-->
        <result>
            <filetype>xxx</filetype> <!--png/nc/txt...-->
            <filecount>1</filecount>
            <parameter>
               <!--defining name,description,tag,type and value range of the parameter-->
            </parameter>
            ...
        </result>
        ... <!--multiple kinds of results-->
    </OutputFileParameters>
    <pre-processing> <!--defining pre-processing type before invoking the script-->
        <pre-processing-type>Latlon</pre-processing-type> <!--Origin/Latlon/YearAvg/SeasAvg/LTM/Subset-->
        <AddToResults>false</AddToResults>
        <!--defining if the pre-processing result need to be added to the result files-->
    </pre-processing>
</function>
```

properties

controls

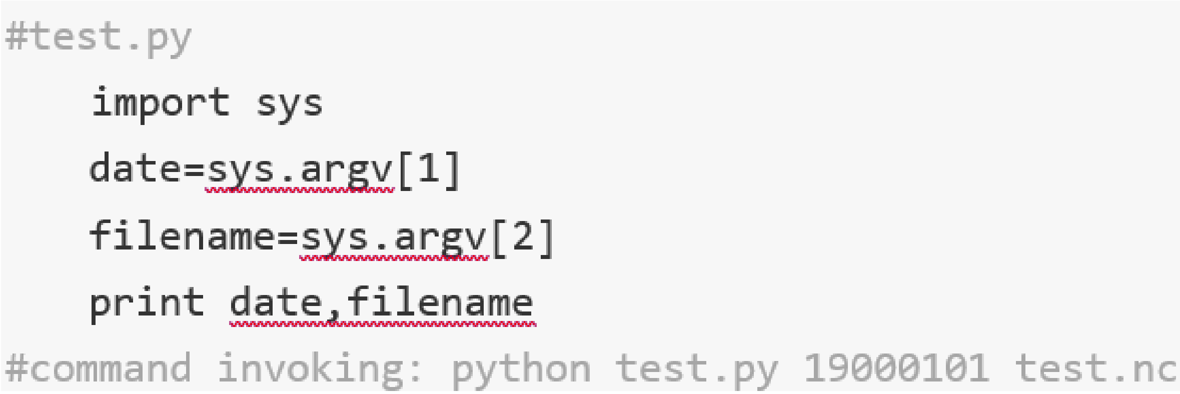
Input/Output related

Pre-processing

Easy way for extensions. Only the analytic script and its XML description are needed.

The script should have an I/O interface for command line and can be invoked by Java.

Python:

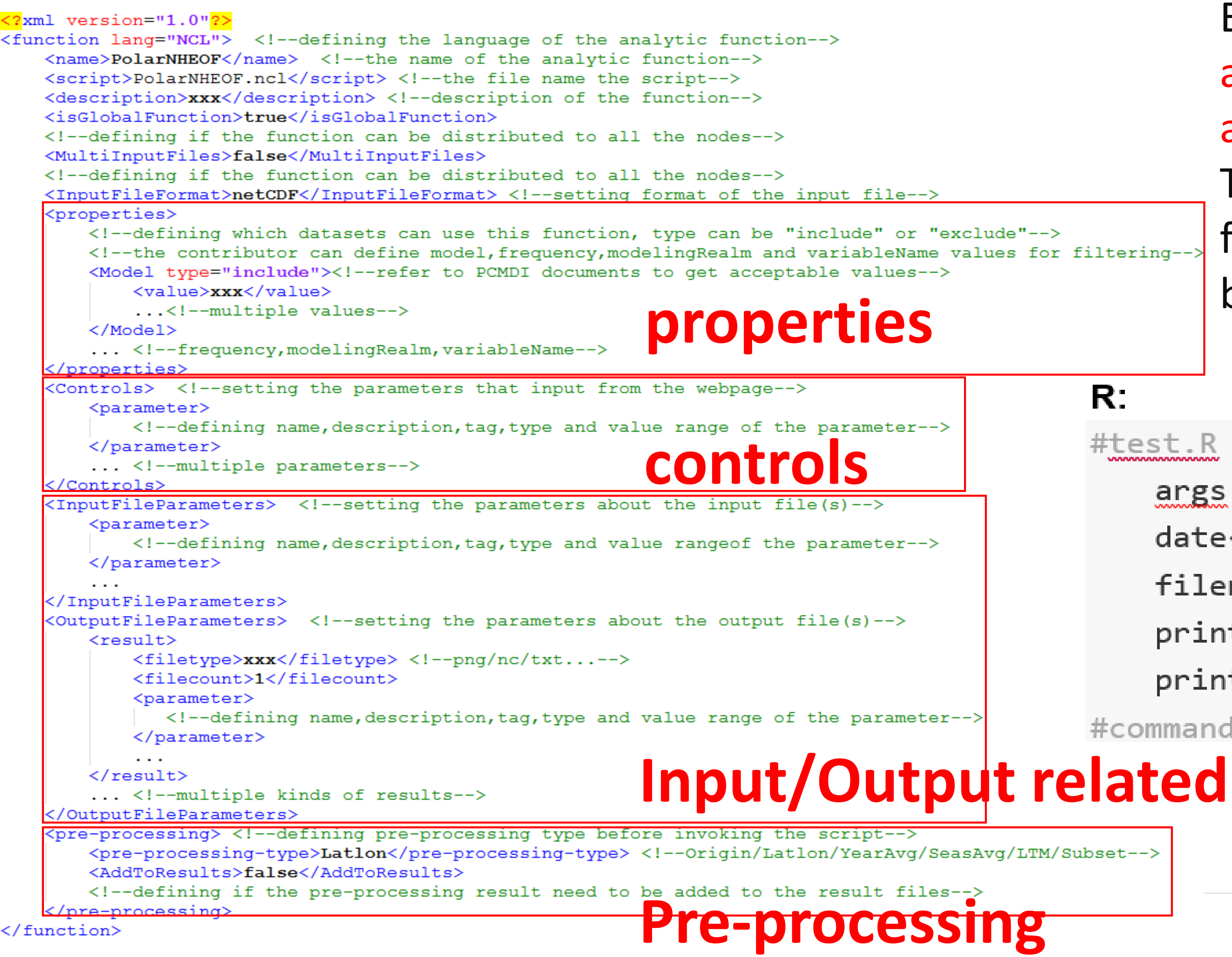


```
#test.py
    import sys
    date=sys.argv[1]
    filename=sys.argv[2]
    print date,filename
#command invoking: python test.py 19000101 test.nc
```

System design-extensibility

Easy way for extensions. Only the analytic script and its XML description are needed.

The script should have an I/O interface for command line and can be invoked by Java.



```
<?xml version="1.0"?>
<function lang="NCL">  <!--defining the language of the analytic function-->
    <name>PolarNHEOF</name>  <!--the name of the analytic function-->
    <script>PolarNHEOF.ncl</script> <!--the file name the script-->
    <description>xxx</description> <!--description of the function-->
    <isGlobalFunction>true</isGlobalFunction>
    <!--defining if the function can be distributed to all the nodes-->
    <MultiInputFiles>false</MultiInputFiles>
    <!--defining if the function can be distributed to all the nodes-->
    <InputFileFormat>netCDF</InputFileFormat> <!--setting format of the input file-->
    <properties>
        <!--defining which datasets can use this function, type can be "include" or "exclude"-->
        <!--the contributor can define model,frequency,modelingRealm and variableName values for filtering-->
        <Model type="include"><!--refer to PCMDI documents to get acceptable values-->
            <value>xxx</value>
            ...<!--multiple values-->
        </Model>
        ... <!--frequency,modelingRealm,variableName-->
    </properties>
    <Controls>  <!--setting the parameters that input from the webpage-->
        <parameter>
            <!--defining name,description,tag,type and value range of the parameter-->
        </parameter>
        ... <!--multiple parameters-->
    </Controls>
    <InputFileParameters>  <!--setting the parameters about the input file(s)-->
        <parameter>
            <!--defining name,description,tag,type and value rangeof the parameter-->
        </parameter>
        ...
    </InputFileParameters>
    <OutputFileParameters>  <!--setting the parameters about the output file(s)-->
        <result>
            <filetype>xxx</filetype> <!--png/nc/txt...-->
            <filecount>1</filecount>
            <parameter>
               <!--defining name,description,tag,type and value range of the parameter-->
            </parameter>
            ...
        </result>
        ... <!--multiple kinds of results-->
    </OutputFileParameters>
    <pre-processing> <!--defining pre-processing type before invoking the script-->
        <pre-processing-type>Latlon</pre-processing-type> <!--Origin/Latlon/YearAvg/SeasAvg/LTM/Subset-->
        <AddToResults>false</AddToResults>
        <!--defining if the pre-processing result need to be added to the result files-->
    </pre-processing>
</function>
```

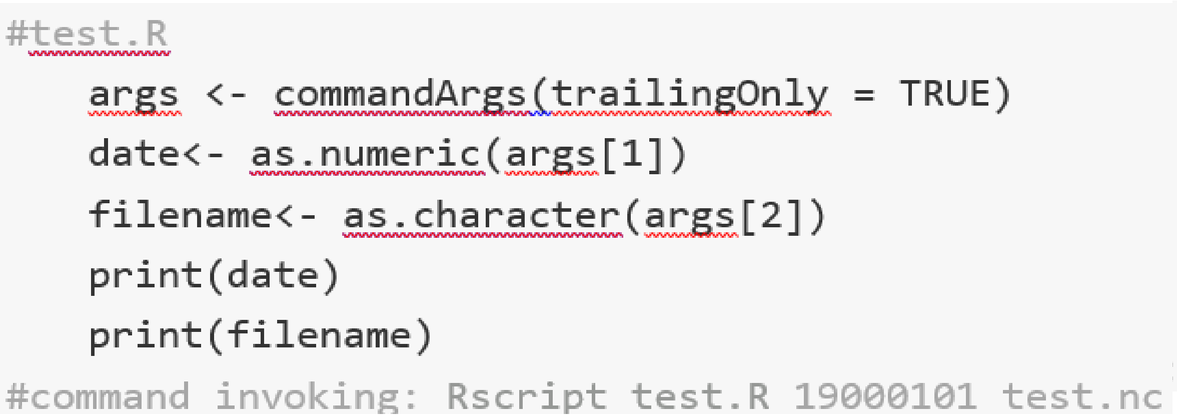
properties

controls

Input/Output related

Pre-processing

R:



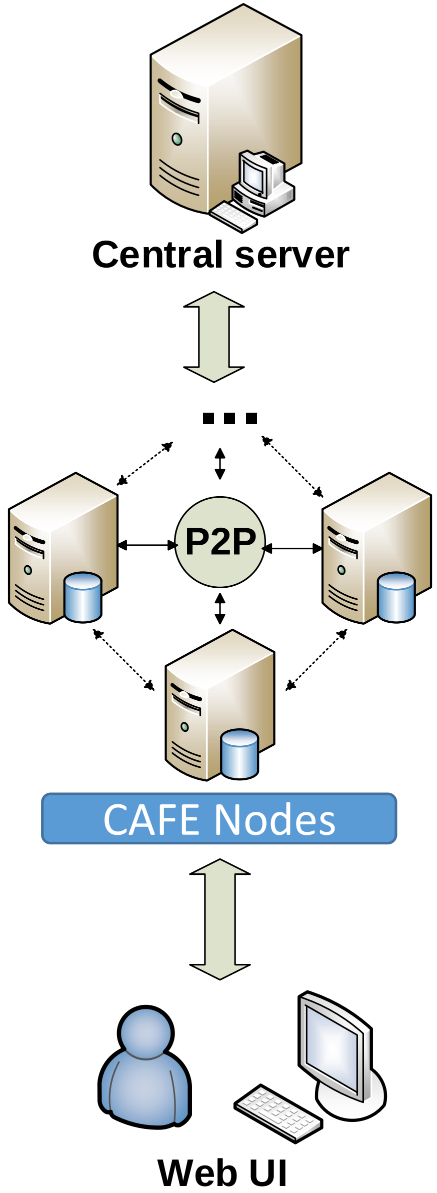
```
#test.R
    args <- commandArgs(trailingOnly = TRUE)
    date<- as.numeric(args[1])
    filename<- as.character(args[2])
    print(date)
    print(filename)
#command invoking: Rscript test.R 19000101 test.nc
```

Prototype and Use Case-prototype system

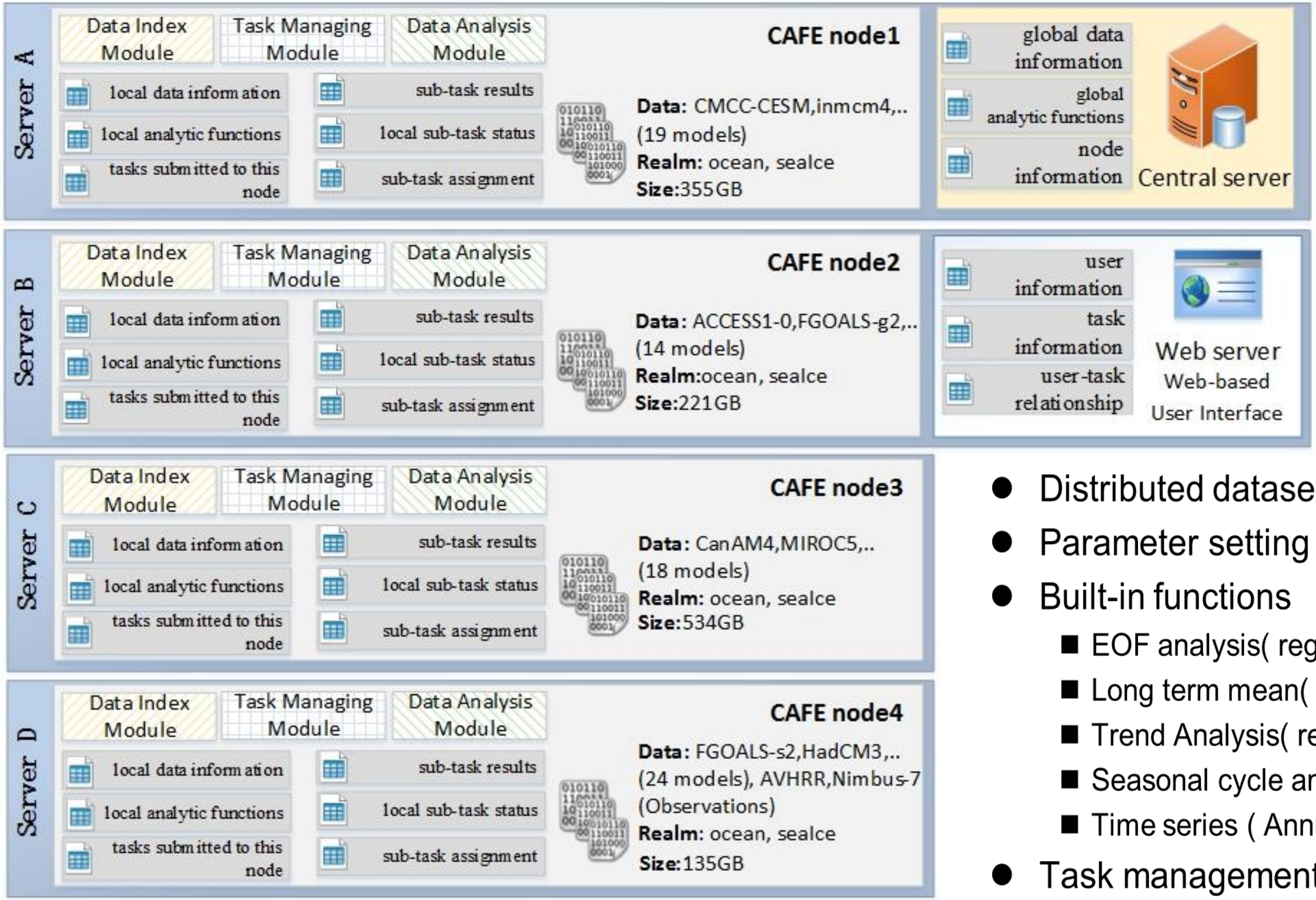
- **CAFE nodes:**
  - Data layer:mybatis3.2.3+mysql
  - Service layer: Spring4.0
  - Interaction: Spring MVC4.0+REST API

- **Analytic scripts:**
  - NCL+NCO+CDO

- **Web-based UI:**
  - Service: pHp+yii
  - Database: mysql
  - Web Server: apache2



Prototype and Use Case-prototype system



Server A

Data Index Module | Task Managing Module | Data Analysis Module

local data information | sub-task results

local analytic functions | local sub-task status

tasks submitted to this node | sub-task assignment

CAFE node1

Data: CMCC-CESM,inmcm4,.. (19 models)
Realm: ocean, sealce
Size:355GB

global data information

global analytic functions

node information

Central server

Server B

Data Index Module | Task Managing Module | Data Analysis Module

local data information | sub-task results

local analytic functions | local sub-task status

tasks submitted to this node | sub-task assignment

CAFE node2

Data: ACCESS1-0,FGOALS-g2,.. (14 models)
Realm:ocean, sealce
Size:221GB

user information

task information

user-task relationship

Web server
Web-based
User Interface

Server C

Data Index Module | Task Managing Module | Data Analysis Module

local data information | sub-task results

local analytic functions | local sub-task status

tasks submitted to this node | sub-task assignment

CAFE node3

Data: CanAM4,MIROC5,.. (18 models)
Realm: ocean, sealce
Size:534GB

Server D

Data Index Module | Task Managing Module | Data Analysis Module

local data information | sub-task results

local analytic functions | local sub-task status

tasks submitted to this node | sub-task assignment

CAFE node4

Data: FGOALS-s2,HadCM3,.. (24 models), AVHRR,Nimbus-7 (Observations)
Realm: ocean, sealce
Size:135GB

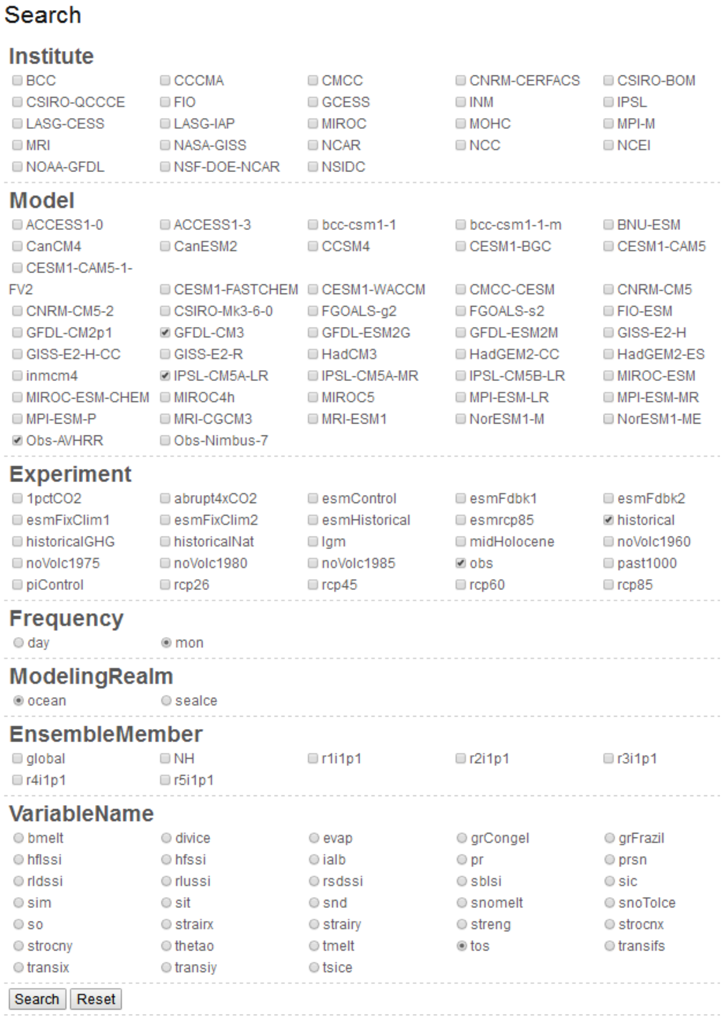
- Distributed dataset access and searching
- Parameter setting and task submitting
- Built-in functions
  - EOF analysis(region, NH, SH)
  - Long term mean(region, NH, SH)
  - Trend Analysis(region, NH, SH)
  - Seasonal cycle analysis(NH, SH)
  - Time series (Annual, Seasonal)
- Task management, result display and downloading

Prototype and Use Case-use case

- Compare and analyze sea surface temperature data(tos) among different model output and the observation data
- Model: GFDL-CM3 and IPSL-CM5B-LR
- Observation data: AVHRR
- Time range: 198501~200512
- Spatial range: 0° N~90° N , 110° E~280° E
- Analysis method: EOF Analysis (Specified regions)

Prototype and Use Case-use case

- At first, user can search for data by specifying querable attributes. Selected three datasets may be archived on several nodes.



Search

Institute

☐ BCC	☐ CCCMA	☐ CMCC	☐ CNRM-CERFACS	☐ CSIRO-BOM
☐ CSIRO-QCCCE	☐ FIO	☐ GCESS	☐ INM	☐ IPSL
☐ LASG-CESS	☐ LASG-IAP	☐ MIROC	☐ MOHC	☐ MPI-M
☐ MRI	☐ NASA-GISS	☐ NCAR	☐ NCC	☐ NCEI
☐ NOAA-GFDL	☐ NSF-DOE-NCAR	☐ NSIDC		

Model

☐ ACCESS1-0	☐ ACCESS1-3	☐ bcc-csm1-1	☐ bcc-csm1-1-m	☐ BNU-ESM
☐ CanCM4	☐ CanESM2	☐ CCSM4	☐ CESM1-BGC	☐ CESM1-CAM5
☐ CESM1-CAM5-1-FV2	☐ CESM1-FASTCHEM	☐ CESM1-WACCM	☐ CMCC-CESM	☐ CNRM-CM5
☐ CNRM-CM5-2	☐ CSIRO-Mk3-6-0	☐ FGOALS-g2	☐ FGOALS-s2	☐ FIO-ESM
☐ GFDL-CM2p1	☑ GFDL-CM3	☐ GFDL-ESM2G	☐ GFDL-ESM2M	☐ GISS-E2-H
☐ GISS-E2-H-CC	☐ GISS-E2-R	☐ HadCM3	☐ HadGEM2-CC	☐ HadGEM2-ES
☐ inmcm4	☑ IPSL-CM5A-LR	☐ IPSL-CM5A-MR	☐ IPSL-CM5B-LR	☐ MIROC-ESM
☐ MIROC-ESM-CHEM	☐ MIROC4h	☐ MIROC5	☐ MPI-ESM-LR	☐ MPI-ESM-MR
☐ MPI-ESM-P	☐ MRI-CGCM3	☐ MRI-ESM1	☐ NorESM1-M	☐ NorESM1-ME
☑ Obs-AVHRR	☐ Obs-Nimbus-7			

Experiment

☐ 1pctCO2	☐ abrupt4xCO2	☐ esmControl	☐ esmFdbk1	☐ esmFdbk2
☐ esmFixClim1	☐ esmFixClim2	☐ esmHistorical	☐ esmrcp85	☑ historical
☐ historicalGHG	☐ historicalNat	☐ lgm	☐ midHolocene	☐ noVolc1960
☐ noVolc1975	☐ noVolc1980	☐ noVolc1985	☑ obs	☐ past1000
☐ piControl	☐ rcp26	☐ rcp45	☐ rcp60	☐ rcp85

Frequency

○ day	● mon

ModelingRealm

● ocean	○ seaIce

EnsembleMember

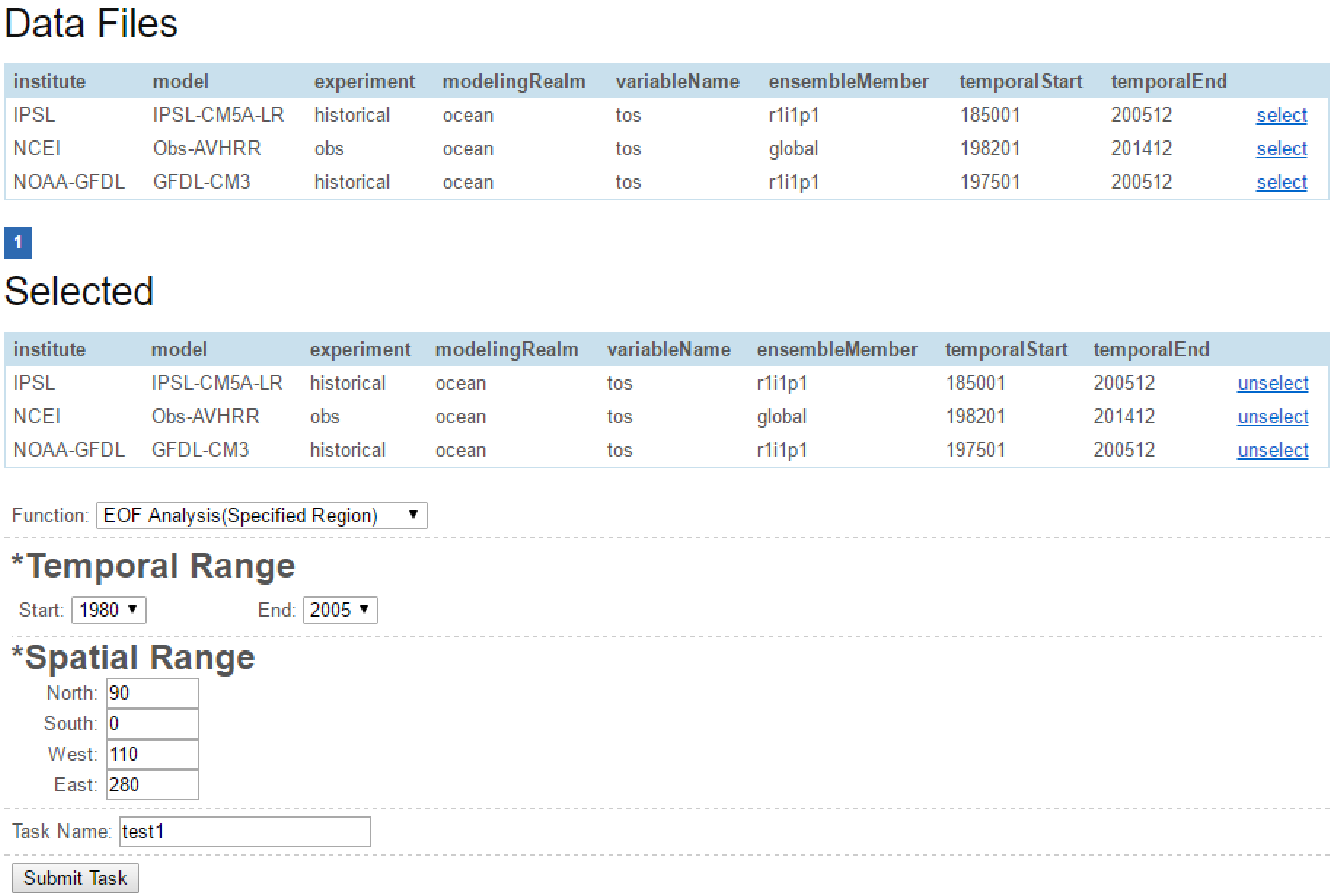
☐ global	☐ NH	☐ r1i1p1	☐ r2i1p1	☐ r3i1p1
☐ r4i1p1	☐ r5i1p1			

VariableName

○ bmelt	○ divice	○ evap	○ grCongel	○ grFrazil
○ hflssi	○ hfssi	○ ialb	○ pr	○ prsn
○ rldssi	○ rlussi	○ rsdssi	○ sblsi	○ sic
○ sim	○ sit	○ snd	○ snomelt	○ snoToIce
○ so	○ strairx	○ strairy	○ streng	○ strocnx
○ strocny	○ thetao	○ tmelt	● tos	○ transifs
○ transix	○ transiy	○ tsice		

Prototype and Use Case-use case

- At first, user can search for data by specifying querable attributes. Selected three datasets may be archived on several nodes.



Data Files

institute	model	experiment	modelingRealm	variableName	ensembleMember	temporalStart	temporalEnd	
IPSL	IPSL-CM5A-LR	historical	ocean	tos	r1i1p1	185001	200512	select
NCEI	Obs-AVHRR	obs	ocean	tos	global	198201	201412	select
NOAA-GFDL	GFDL-CM3	historical	ocean	tos	r1i1p1	197501	200512	select

1

Selected

institute	model	experiment	modelingRealm	variableName	ensembleMember	temporalStart	temporalEnd	
IPSL	IPSL-CM5A-LR	historical	ocean	tos	r1i1p1	185001	200512	unselect
NCEI	Obs-AVHRR	obs	ocean	tos	global	198201	201412	unselect
NOAA-GFDL	GFDL-CM3	historical	ocean	tos	r1i1p1	197501	200512	unselect

Function: EOF Analysis(Specified Region) ▾

*Temporal Range

Start: 1980 ▾ End: 2005 ▾

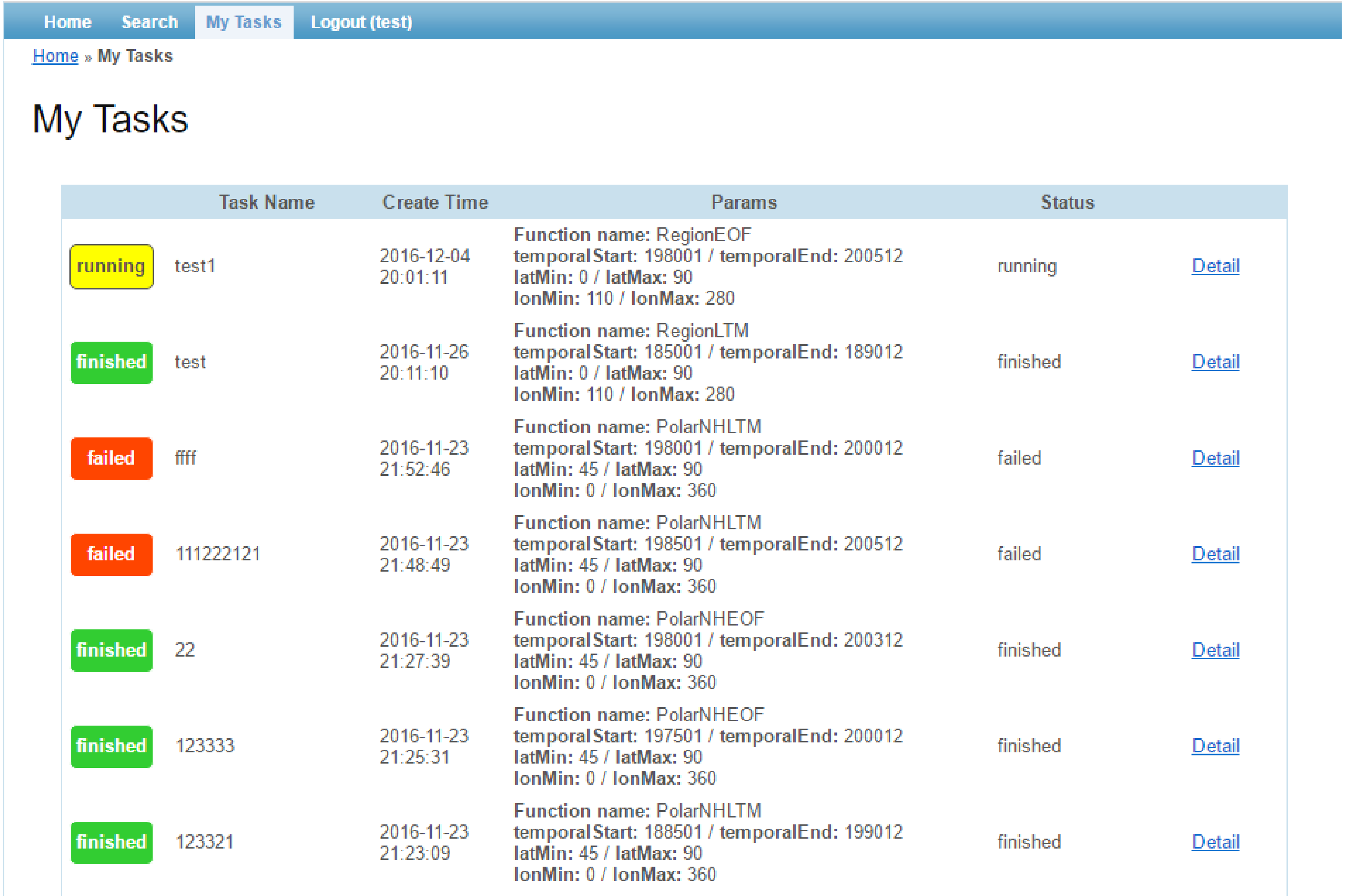
*Spatial Range

North: 90
South: 0
West: 110
East: 280

Task Name: test1

Prototype and Use Case-use case

- After submitting the task, user can check status of each task.
- User can obtain the results when the status is *finished*.



Home » My Tasks

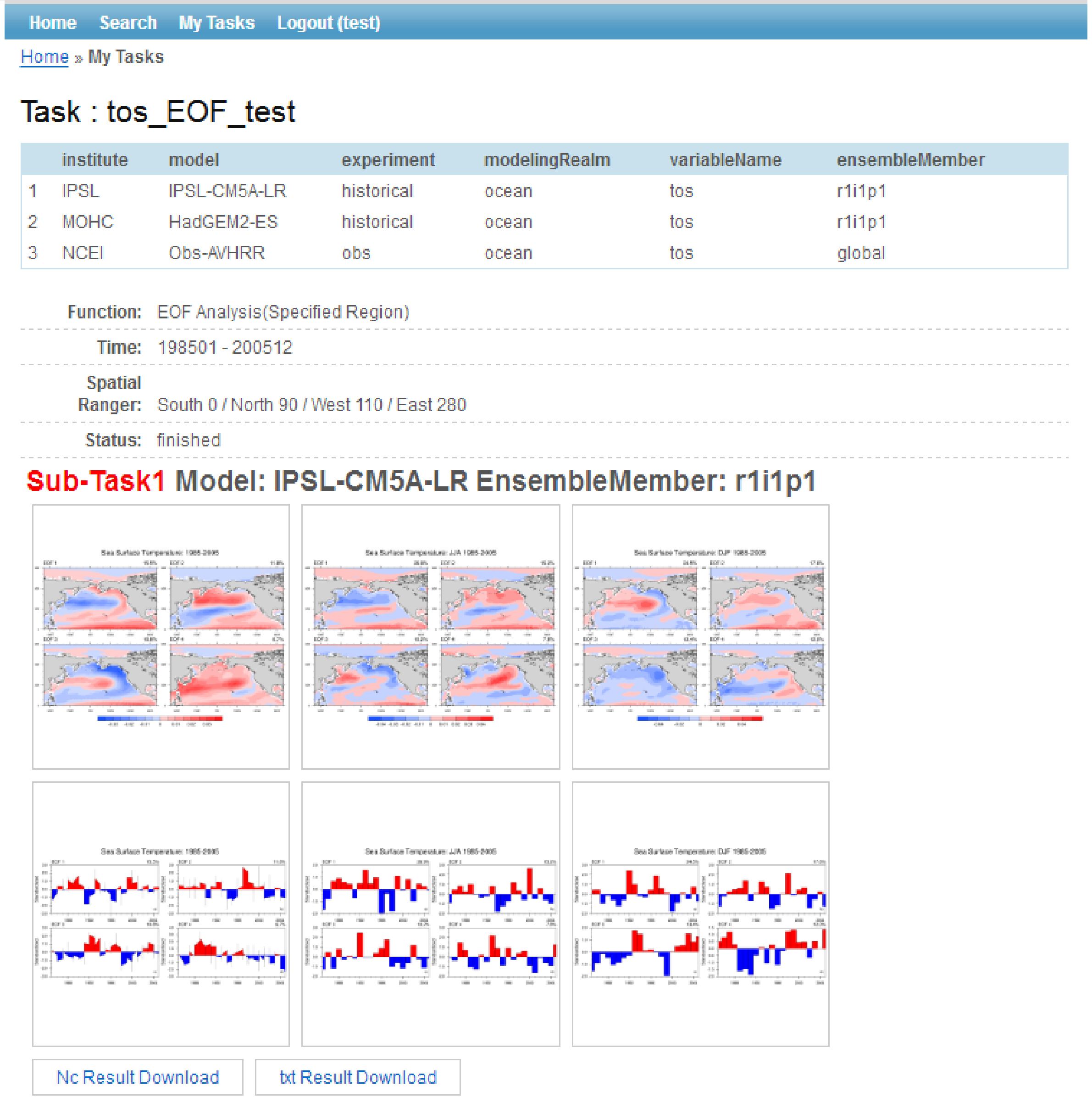
My Tasks

	Task Name	Create Time	Params	Status	
running	test1	2016-12-04 20:01:11	**Function name:** RegionEOF **temporalStart:** 198001 / **temporalEnd:** 200512 **latMin:** 0 / **latMax:** 90 **lonMin:** 110 / **lonMax:** 280	running	Detail
finished	test	2016-11-26 20:11:10	**Function name:** RegionLTM **temporalStart:** 185001 / **temporalEnd:** 189012 **latMin:** 0 / **latMax:** 90 **lonMin:** 110 / **lonMax:** 280	finished	Detail
failed	ffff	2016-11-23 21:52:46	**Function name:** PolarNHLTM **temporalStart:** 198001 / **temporalEnd:** 200012 **latMin:** 45 / **latMax:** 90 **lonMin:** 0 / **lonMax:** 360	failed	Detail
failed	111222121	2016-11-23 21:48:49	**Function name:** PolarNHLTM **temporalStart:** 198501 / **temporalEnd:** 200512 **latMin:** 45 / **latMax:** 90 **lonMin:** 0 / **lonMax:** 360	failed	Detail
finished	22	2016-11-23 21:27:39	**Function name:** PolarNHEOF **temporalStart:** 198001 / **temporalEnd:** 200312 **latMin:** 45 / **latMax:** 90 **lonMin:** 0 / **lonMax:** 360	finished	Detail
finished	123333	2016-11-23 21:25:31	**Function name:** PolarNHEOF **temporalStart:** 197501 / **temporalEnd:** 200012 **latMin:** 45 / **latMax:** 90 **lonMin:** 0 / **lonMax:** 360	finished	Detail
finished	123321	2016-11-23 21:23:09	**Function name:** PolarNHLTM **temporalStart:** 188501 / **temporalEnd:** 199012 **latMin:** 45 / **latMax:** 90 **lonMin:** 0 / **lonMax:** 360	finished	Detail

Prototype and Use Case-use case

- Maps and time series graph are provided.

- Results as NetCDF files and Text files are provided for downloading.



Home » My Tasks

Task : tos_EOF_test

	institute	model	experiment	modelingRealm	variableName	ensembleMember
1	IPSL	IPSL-CM5A-LR	historical	ocean	tos	r1i1p1
2	MOHC	HadGEM2-ES	historical	ocean	tos	r1i1p1
3	NCEI	Obs-AVHRR	obs	ocean	tos	global

Function: EOF Analysis(Specified Region)

Time: 198501 - 200512

Spatial Ranger: South 0 / North 90 / West 110 / East 280

Status: finished

Sub-Task1 Model: IPSL-CM5A-LR EnsembleMember: r1i1p1

Conclusion

- CAFE can support web-based online batch analysis of distributed environmental data, as well as multi-node collaboration for data intercomparison.
- CAFE can dramatically decrease the amount of data need to be transferred from data centers to data users.
- CAFE can be easily extended to support more data analytics functions and more data formats.
- CAFE is very promising in facilitating overall research efficiency when dealing with large volume of environmental data that are distributedly maintained.

Future Work

- Look forward to a close collaboration with the development of ESGF
- Support for OPeNDAP, WMS and WPS
- Support for user work space and management of tasks
- Autogeneration of XML descriptions of the analytic functions from user-input web page forms.
- Tracing data provenance and provide intermediate results
- Adding a third-party authentication mechanism and encryption for the APIs between the Web-based UI and CAFE nodes

THANKS FOR YOUR ATTENTION

CAFE working team:
Hao Xu(CESS,xuhao13@mails.tsinghua.edu.cn)
Sha Li(CESS)
Wenhao Dong(CESS)
Shiming Xu(CESS)
Wenyu Huang(CESS)
Yanluan Lin(CESS)
Bin Wang(LASG,CESS)
Fanghua Wu(BCC)
Xiaoge Xin(BCC)
Li Zhang(BCC)
Zaizhi Wang(BCC)
Tongwen Wu(BCC)
Yuqi Bai (CESS, yuqibai@mail.tsinghua.edu.cn)