UNIVERSITY OF Southampton

euindiagrid इउभारतग्रिड

Stock Analysis Application

Augmenting the GRID with Application-Level infrastructural services beyond core MW services.

Stock Analysis Application

Summary:

- The Stock Analysis research problem.
- The software solution in terms of the Problem-Domain.
- The software architecture of the solution.
- Development process and roadmap.

The Stock Analysis research problem

• In general Finance Research involves ***statistical analysis*** of individual Financial Instruments.
• For this particular occasion: thorough statistical analysis of 700 ***securities*** (stocks, bonds, options, etc.).

- For each security: several years data, in common format, about all *market events*.
- i.e. latest trade volume, best-buy price, best-sell price, etc. *whenever* an event occurs such as *new order arrival, etc.*
- Overall: 4 TB of data – 100 GB *compressed* (factor of 40!)
- In practice: for each security, there is one ZIP file containing several years data for that security. Size ~10s or ~100s of MB.

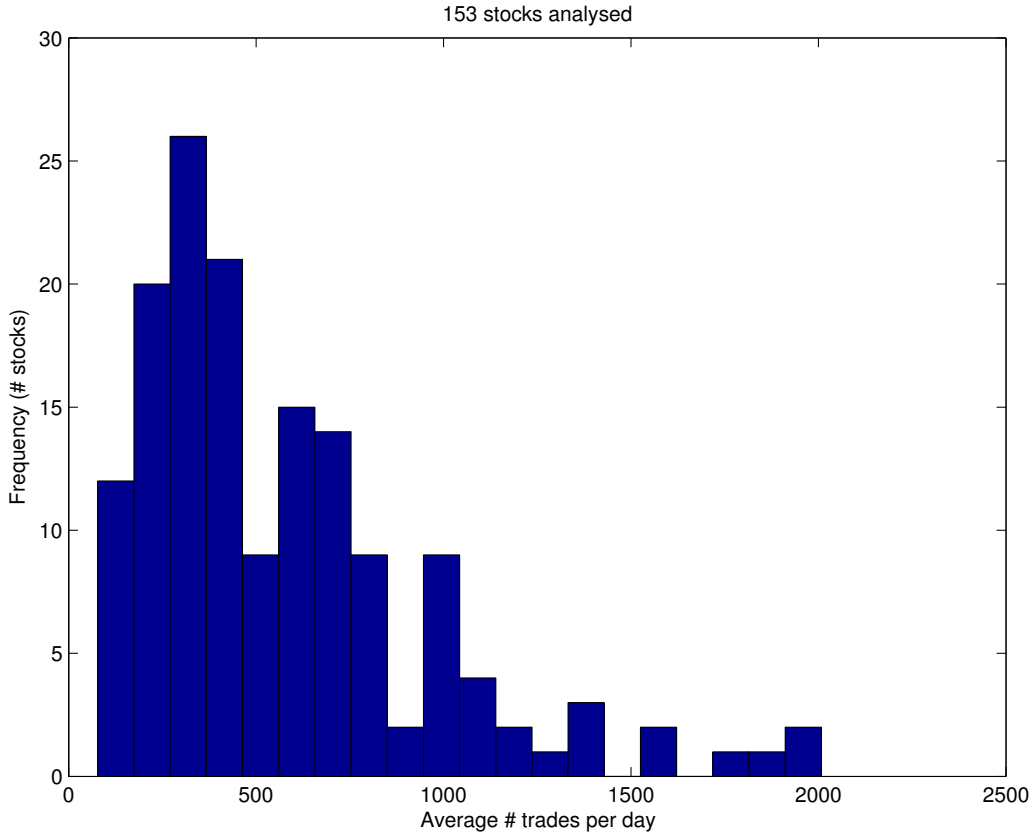
The Stock Analysis research problem

To analyse *one security*, ***two steps*** are necessary:

1. Produce ***150 time series*** characterising the security's behaviour (e.g.: trade prices, returns, volatility, waiting time between trades, etc.)
   - Output produced: 200 GB of ZIP files (~8 TB uncompressed)
   - *Custom C++ code* in active development by the researchers.

2. Process time series to obtain output in form of tables *and* graphs
   - *Uncompressed* data files in the *overall* order of 10s of MB.
   - *MATLAB executable code* in active development by researchers.

The Stock Analysis research problem



Average number of trades of a given stock

The Stock Analysis research problem

The problem is ideally suited to a GRID environment since it is *embarassingly parallel*: have each WN process one security's data.

Software Problem-Domain solution

Problem Domain we are modelling involves *two* components:

- The gLite EGEE GRID (*with WMS, WN, SE, etc.*)
- The specific Finance research just described.

This is context within which the software solution is looked for.

A solution in 3 *distinct* parts:

- *Business logic*: code that encapsulates the finance/statistics knowledge
- *Analysis interface*: accepts analysis requests ***and*** manages GRID jobs
- *Local environment manager*: encapsulates GRID knowledge when interacting with Business Logic ***in*** WN.

Software Problem-Domain solution

Business-logic code:

- Will assume all required data is *locally available*.
- Will have *no knowledge* of *GRID* or *Network Environemnt.*
- Will run as an *executable*: not a library.
- Will be C++ but also ***MATLAB*** executable!

Motivated by:

- Need to run without modification on normal PC and GRID (facilitates debugging and maintenance)
- Need to run on GRID without presence of commerrcial *development environments* such as that for MATLAB
- Need secure storage for data

Software Problem-Domain solution

Analysis interface GRID code:

To allow researchers to *run* and *control* ***full analyses*** that automatically process ***large numbers*** of files, according to:

- List of *security names*:
  - To act as *filename discriminants* when specifying subset of files to analyse.
- Business-logic code to use
- Parameters of Business-logic code (i.e. time interval to analyse, etc.)

To automatically manage the jobs:

- Monitor completion of analysis (one job per security)
- Fetch *meta-data* about each security from job's input sandbox (analysis output files go straight to SE, *not* in sandbox)
- Fetch any grid error log in case of job failure
- Compile list of completed stocks, failed stocks, interrupted stocks.

Software Problem-Domain solution

Local environment manager GRID code:

To *create* and *manage* the ***local environment*** in WNs, for the business-logic code:

- *Get* business-logic engine and *install* it in the WN (environment variables, +x attribute, etc.)
- *Fetch* input files to be processed by business-logic engine.
- *Execute* business-logic engine invoking it with the correct parameters.
- *Save* selected produced output files in SE
- *Clean-up* WN environment.
- *Monitor* operation in WN + *produce* meta-data.

Software Problem-Domain solution

The two most critical ***engineering issues*** since they are *points of variation* within the application:

- To achieve automatic processing of large numbers of files… ***Parameterise the input and output file names on the security***
  - Implies organisation of the input files!
- Manage the business-logic engines in the WNs… ***Encapsulate the need for installation into a generic step, that is then specialised for each specific Business-logic engine***
  - C++ use case simple: setting x attribute on executable, transfer within input sandbox.
  - MATLAB executable more elaborate: environment variables, untarring, fetching executable from SE.

Software Problem-Domain solution

Important secondary objective: to strive for sufficient generality to be able to use the application beyond Finance i.e. in Computational Statistics

- Delicate balance between total generality and finely tailored operations:
  - JDL + command line tools + heavy use of ad-hoc scripting.
  - Highly specialised tool that accepts as input simply a study parameter.
- Yet overall goal of Application is: *reliable automatic mass processing of large numbers of files*.

Until now the most reasonable compromise on flexibility seems: ***parametrisation on stock*** + ***encapsulation of engine installation***.

Software Problem-Domain solution

Extreme view of application:
software that allows use of any *engine* whose input and output arguments can be parametrised on *one* attribute… ***but it is too early to say! More real world use cases are needed: devil is in the details!***

Some examples under consideration

- *Monte Carlo simulation*: Each WN *runs a separate simulation* which differs across WN based on parametrised input file and is saved in parametrised output file
- *Grid search*: Each WN *evaluates a separate objective function* which differs across WN based on parametrised input file and is saved in parametrised output file

These cases can be handled simply by modifying the content of the various files!

Software Architecture of the solution

Given:

- The problem domain just described
- That a GRID is not just computation + storage + information system
- That the application should be usable beyond Finance

It is a matter of balancing *usability* with *generality*, given the extra room warranted by ***co-operating services GRID view***.

Since:

- Intrinsically more difficult to define and write a software that it will fit ***well*** many classes of problems
- Less so to define and write a software that will fit ***well*** at least ***part*** of problems
- Software that *does so* is more likely to be usable: it can *co-operate* in more elaborate solutions (trick is: *to identify* such parts).

Software Architecture of the solution

Instead of a *monolithic* application that attempts to solve as many classes of problems as possible:

Full GRID Web Service:
- To carry out the analysis characterised by ***two*** highlighted variational points.
- That interacts with the other core GRID services.

To be used by:
- User-friendly web applications.
- *Other web services for more elaborate operations.*
- Directly by simple command-line clients

Software Architecture of the solution

Information System integration

- Reading needed information such as the WMS for the VO
- Supplying information such as advertising its presence.

WSx standard compliant

- Real time Remote Logging.

Security

- GSI + VOMS
- Implies a ***declarative security model*** for the web service!
  - Has still to be studied in detail
  - Likely to include a role for loading business-logic code.

QoS

- It is expected from a GRID Web Service
- Has to be studied in detail

Software Architecture of the solution

Stock Analysis Application is an attempt to put in practice the architectural view of achieving a solution through co-operating services.

- No certainty if identified *variational points* are good for many classes of practical situations
- Further fine tuning likely: we are open to changes.

It is a step towards augmenting core MW services with higher level ones

Development process and roadmap

Researchers need results now:

- Cannot wait for complete final solution
- Requirement definition tricky

An ***Iterative development process*** in place: quick initial solution is being continuosly evolved.

Development process and roadmap

Currently:

- Classic client-server application is in operation
- Full support for C++ use case
- Real-time remote logging on WNs is operative
- MATLAB use case is in advanced prototypical stage

Developement began in November:

- 9GB of ZIP data used to gauge solution
- About 1000 runs launched
- Each stock processed for 5-6 hours, with C++

Development process and roadmap

- ***February***
  - MATLAB use case fully integrated.
  - Application will run in HellasGRID at SEE VO.
- ***March***
  - External UI command invocation eliminated: service installable outside of a UI host.
- ***April***
  - Server dressed up as a full Web Service
  - Integration into the GRID information system.
- ***May***
  - Improved robustness.
- ***June***
  - Draft on *declarative security model* + QoS; no implementation.

The team

http://euindia.ictp.it/stock-analysis-application

- Dr. Stefano Cozzini cozzini@democritos.it EUINDIA Technical co-ordinator.
- Prof. Spyros Skouras of Athens University of Economics and Business skouras@aueb.gr Finance research leader
- Dr. Giorgios Michalareas of Southampton University gm@ecs.soton.ac.uk C++/MATLAB Finance code developer.
- Eng. Ezio Corso ecorso@ictp.it GRID code developer.

MATLAB executable deployment

Requires 3 files

MCRInstaller.zip

File needed to be created only once by MATLAB, contains generic libraries of MATLAB for standalone executables. NOT application specific.

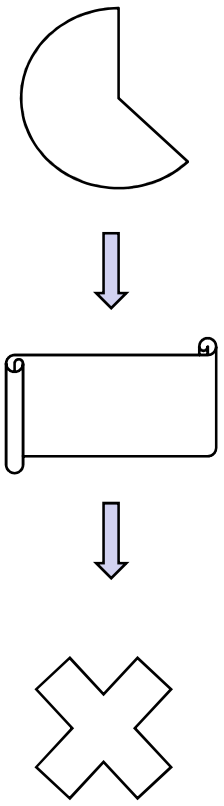
executable

binary file representing the main execution of the application. Application Specific. Created by MATLAB Compiler.

executable.ctf

encrypted archive with user libraries , not contained in MATLAB, which are needed by the application. Application Specific. Created by MATLAB Compiler.

Standalone Deployment Process LiNux



- Create Directory in WN
- Get MCRInstaller.zip (from UI or SE)
- Unzip it there

- Add new Directory Subdirectories in WN System's Environment Path Variables (i.e. LD_LIBRARY_PATH)

- Get executable and its ctf companion(from UI)
- Run executable