RESTful Digital Web Services for Market Price Forecasting with R

An Application on Equity Markets Using Garch Models

Mihai CALCIU
LEM – IAE
University Lille 1
Lille, France
michel.calciu@univ-lille1.fr

Cristiana TUDOR
International Business and Economics
Bucharest University of Economic Studies
Bucharest, Romania
cristiana.tudor@net.ase.ro

Abstract— **Web services (WS) are originally and typically SOAP (Simple Object Access Protocol) based. SOAP based web services offer a powerful architecture and communication protocol that allows model based marketing decisions support (MMDS) to be integrated, served and get discovered over the Internet. Lately REST (REpresentational State Transfer) has emerged as an alternative Web service design model that is less typed and in many aspects less "sophisticated" and therefore literally "restful" for decision support builders. RESTful MMDS web services are easier to implement by marketing scientists and more straightforward and "naturally, intuitively" discoverable by managers. This paper presents the differences between the two approaches and introduces a RESTful digital web service for market price forecasting using the GARCH financial model.**

Keywords-REST web service, price forecasting, equity markets, GARCH model

I. Introduction

Web services are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. They can easily integrate with other services, from the same or different companies, to create a complete business process. This interoperability allows to dynamically publish, discover, and bind a range of Web services through the Internet. The following standards play key roles in Web services: Universal Description, Discovery and Integration (UDDI), Web Services Description Language (WSDL), Web Services Inspection Language (WSIL), Simple Object Access Protocol (SOAP). They constitute together with some additional technologies such as WS-Addressing, WS-ReliableMessaging, WS- Security the "Big" Web services technology stack (Pautasso & al., 2008, Richardson & Ruby, 2007) that defines the WS-* standard.

A. Towards an alternative standard

This WS-* standard came under criticism as to its presumed complexity. The alternative REST solution takes its inspiration from WWW itself and shows that the same principles that have made the success of the World Wide Web can be used to solve enterprise application integration problems and to simplify service-oriented architectures. MMDS are aimed to be integrated as enterprise applications and can therefore take benefit from such technological trends.

REST is a style of software architecture for distributed systems such as the World Wide Web. The WWW itself is probably the largest implementation of REST principles.

REST was introduced and defined by Roy Fielding (2000) , one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification versions 1.0 and 1.1.[3][4]. A service is considered "RESTful" if conforming to the REST constraints.[5]

"RESTful Web services are gaining increased attention not only because of their usage in the Application Programming Interface (API) of many Web 2.0 services (xxx, 2007), but also because of the presumed simplicity of publishing and consuming a RESTful Web service (Vinowski, 2002)."

This paper shows that the REST Web service design model is less typed and in many aspects less "sophisticated" and therefore literally "restful" for decision support builders. RESTful MMDS web services are easier to implement by marketing scientists and more straightforward and "naturally, intuitively" discoverable by managers. This paper presents the differences between the two approaches and introduces a RESTful digital web service for market price forecasting .

II. Type Style and Fonts

A. Compare SOAP and REST

"There are two analogous ways of classifying the services that inhabit the programmable web: by the technologies they use (URIs, SOAP, XML-RPC, and so on), or by the underlying architectures and design philosophies." (Richardson & Ruby, 2007, p. 5).

1) Vocabulary re-use vs. its arbitrary extension: HTTP and SOAP

"An RPC-style web service accepts an envelope full of data from its client, and sends a similar envelope back. The method and the scoping information are kept inside the envelope, or on stickers applied to the envelope. … HTTP is a popular envelope format, since any web service worthy of the name must use HTTP anyway. SOAP is another popular envelope format (transmitting a SOAP document over HTTP puts the SOAP envelope inside an HTTP envelope). Every RPC-style service defines a brand new vocabulary. Computer programs work this way as well: every time you write a program, you define functions with different names. By contrast, all RESTful web services share a standard vocabulary of HTTP methods. Every object in a RESTful service responds to the same basic interface." (Richardson & Ruby, 2007, p.14)

While REST re-uses (a rich) vocabulary and methods already available in the well-known, well-defined HTTP protocol, SOAP encourages service designers to define their own methods. REST vocabulary includes URIs, Internet media types, request and response codes and methods like: GET, POST, PUT, DELETE. Additional features of the protocol like layered proxy and gateway components are used to perform additional functions on the network, such as HTTP caching and security enforcement.

2) Rest's Only HTTP vs SOAP's multi-communication protocol.

By sticking to HTTP re-uses its capabilities such as authentication, caching, and content-type negotiation. While also using HTTP, SOAP doesn't stick to it works equally well over raw TCP, named pipes, message queues, etc. By favoring application specific vocabulary it may favor "re-invention of the wheel" functionality that disregards many of HTTP's existing capabilities.

3) OpenCPU a promising attempt to offer statistical computing as a RESTful web service

OpenCPU (Ooms, 2011) is a promising attempt to offer statistical computing as a RESTful service. It literally opens up the R statistical software as a REST style service. R is unique as it is open-source and backed by a professional and experienced team (called the r-core group) as well as a huge community of users from both academics and industry. It provides both a reliable general functional computing language as well has specialized packages for almost any scientific field. OpenCPU uses URIs combined with HTTP methods like GET having directories or R object names as endpoints in order to list objects or render objects when the indicated URI (Universal Ressource Identifier) includes mime-type and POST in order to call R functions. In order to discover the requiered parameters of a R function the GET URI would be /R/pub/[package name]/[object name]/[output format] for the garch function that is available in the tseries package the command would be GET /R/pub/tseries/garch/ascii.

Two of the most widely used models in investigating return volatility are the Autoregressive Conditional Heteroskedasticity (ARCH) and Generalized ARCH (GARCH) developed by Engle (1982), and extended by Bollerslev (1986) and Nelson (1991). Two important empirical traits of financial time series i.e. fat tails and volatility clustering, or the tendency for large (small) chnages in prices to be followed by random large (small) changes, can be captured by the GARCH class models. The GARCH (p,q) model is estimated with the following simultaneous equations:

$$R_t = \mu + \varepsilon_t$$

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2 \quad (1)$$

where p is the order of GARCH while q is the order of ARCH process. The error term ε_t is assumed to be normally distributed with zero mean and conditional variance, σ_t^2. All parameters in variance equation must be positive, and $\alpha + \beta$ is expected to be less than, but close to, unity, with $\beta > \alpha$.

The ARCH term reflects news about volatility from the previous period and is measured as the lag of the squared residual from the mean equation. The GARCH term (β) shows the persistence of volatility to a shock or the impact of old news on volatility.

The mean equation is written as a function of exogenous variables with an error term (see Miron and Tudor, 2010 for a more thorough discussion regarding GARCH models and empirical applications on stock returns data).

In order to execute the above mentioned function with apropriate arguments the call would be POST R/pub/tseries/garch/json?order="c(1,1)"&data="closes" where order indicates the lags for the ARCH and for the GARCH term (see later in this paper) and the data are closing price quotations on futures markets. This would produce output in the convenient JSON format. If a subsequent function (like summary) needs to be applied to the output object a double step request could be done. In a first step the previous command with ouput type /save (instead of /json) would place the resulting object in the R/tmp directory with a unique name (hashkey) and the second step would apply the subsequent function (summary) to the saved object in this way:POST /R/pub/base/summary/print?object=[hashkey]. Detailed and highly illustrative information can be found at https://public.opencpu.org/.

III. TOWARDS A RESTFUL MARKET PRICE FORECASTING WEB SERVICE

The web service we introduce here takes a list of futures market and stock price quotation that are available on Yahoo-finance web service and displays it in the left panel in a selectable tree format. The user can drag the desired futures product from the tree and drag it in display panel where price quotations together with forecasted values from our models are drawn. Graphs and forecasts are produced within the web service using the statistical system R by exploiting capabilities from its Rapache and brew packages. Fig.1 shows the representation of Air France-KLM stock price in this manner.

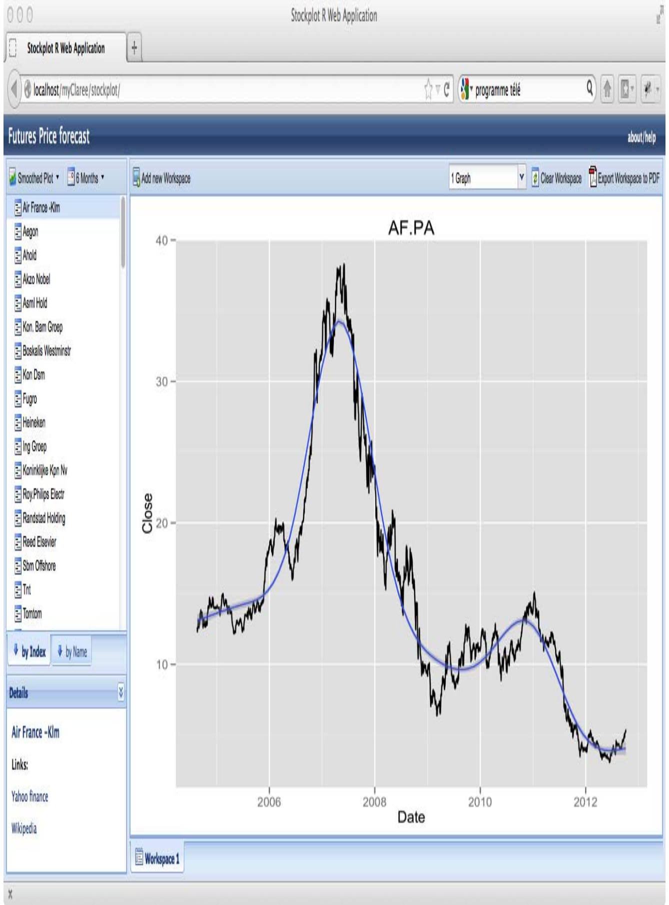

Figure 1 - A RESTful market price forecasting web service

IV. Conclusions

Besides applying better models for forecasting future market prices, this research is probably the first attempt in marketing literature to package model based decision support as a RESTful web service.

We introduce RESTful web services and compare them to their more sophisticated alternative the SOAP based or WS* web services that we have presented in a previous research dealing with web technologies for building MDSS (Calciu, 2007). Our purpose is to keep marketing scientist updated with new technologies affecting both statistical computing and model based decision support systems.

We demonstrate our approach with a RESTful MDSS and present OpenCPU a very recent and promising initiative, part of an ongoing doctoral research (Ooms, 2011) , that in our opinion substantially contributes to democratize statistical computing by offering a REST style web service architecture to one of today's most elegant statistical systems.

Acknowledgment

This research was supported by CNCS-UEFISCDI, Project number IDEI 303, code PN-II-ID-PCE-2011-3-0593.

References

[1] Fielding, R.T., "Architectural Styles and the Design of Network-based Software Architectures", Doctoral dissertation, University of California, Irvine, 2000..

[2] Fielding, R.T.; Taylor, R.N., "Principled Design of the Modern Web Architecture" (PDF), ACM Transactions on Internet Technology (TOIT) (New York: Association for Computing Machinery) vol. 2 (2), pp. 115–150, doi:10.1145/514183.514185, ISSN 1533-5399, 2002.

[3] Miron, D., C. Tudor., "Asymmetric Conditional Volatility Models: Empirical Estimation and Comparison of Forecasting Accuracy". Rom Journal Ec Forecast vol. 13 (3), pp. 74-92, 2010.

[4] Ooms, J.C.L, "Open Computing: Challenges and Opportunities". Phd. Proposal Draft. UCLA Dept. of Statistics, December 2011.

[5] Richardson, L.; Ruby, S., "RESTful Web Services", O'Reilly, ISBN 978-0-596-52926-0, 2007.

[6] Pautasso, C.; Zimmermann, O. Leymann, F., "RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision", 17th International World Wide Web Conference (WWW2008) (Beijing, China), 2008.

[7] Vinoski, S., "Putting the "Web" into Web services: Interaction models", part 2. IEEE Internet Computing, vol. 6(4), pp. 90–92, July 2002.

[8] Xxx "Programmable Web. API Dashboard", 2007. http://www.programmableweb.com/apis