**Advanced Analytics with R & SQL: Part I - R Distributions**

**By** [**Frank A. Banin**](http://www.sqlservercentral.com/Authors/Articles/Frank__A._Banin/626010/)**, 2016/10/31**

**FROM:** [**http://www.sqlservercentral.com/articles/Data+Science/146555/**](http://www.sqlservercentral.com/articles/Data+Science/146555/)

**Introduction**

SQL Server currently has some very powerful and cool features, but the integration of R (the most used Analytics, data mining and data science software) into SQL Server is epic. The hope is that developers, analysts, data scientists, businesses and enterprises will begin to grasp the potential this offers.

I first published the article [Data Science for SQL Folks: Leveraging SQL and R](http://www.sqlservercentral.com/articles/116157/)  on 11/03/2014. Three months after that publication Microsoft announced its acquisition of Revolution Analytics (Revo), the statistical software company focused on supporting the free and open source software R. I did not see the acquisition coming but like many others (esp. those in the BI and advanced analytics space), it was not a surprise.

The question is why the acquisition? And better yet, why did Microsoft thought it best to integrate R into SQL Server? In that [article](http://www.sqlservercentral.com/articles/R/116157/) I sought to drive home the importance SQL (SQL Server), structured data, and R in value driven analytics, all of which underlines the basis of the acquisition and integration.

To put the importance of the acquisition and the integration into perspective once again; In KDnuggets poll which asks the data mining community and vendors for *“The most used Analytics, data mining and data science software”*, R and SQL were ranked 1st and 3rd respectively. In the 2015 Poll:

* 1st : R (46.9%)
* 3rd : SQL (30.9%)

The results of these poll mean that, *SQL Server R Services* as it stands today integrates two of the three most used programming/statistics languages for Advanced Analytics (data mining/data science). More importantly the interoperable R Server scales R analytics to Big Data with parallelized algorithms and distributed processing leading to unequaled performance and flexibility.

In this part of this series we will focus on the various R distributions available today and how they evolved before and after Microsoft acquisition of Revo. In the process, we will reconcile the R nomenclature which can be confusing even for veterans. For newbies, we will particularly learn how the flagship commercial R server came to be and how it *Parallelized Algorithms* enable enterprise scale predictive modeling and data mining solutions on Big Data.

**R Server & Advanced Analytics**

With R Server, Advanced analytics can be delivered where your data lives (in-database, in-Hadoop and in-Apache Spark) eliminating data movement and therefore reducing latencies, reducing operational costs and improving enterprise data security. Today, with SQL Server and the integrated R Server you can run experiments on big data, build a scalable model and operationalize it using SQL Server Management Studio (SSMS) or Visual Studio (VS) as the only development tool. By utilizing SQL Server in-memory transactional processing capabilities enterprises are using R & SQL to build intelligent Application in various domains including:

* **Sales and marketing**: Campaign effectiveness, Customer Acquisition, Cross-sell and upsell, Loyalty programs, Marketing mix optimization, Demand forecasting
* **Finance and risk:** Fraud detection, Credit risk management
* **Customer and channel**: Lifetime customer value, Personalized offers, Product recommendation, Customer Service improvement
* **Operations:** Remote Monitoring, Operational efficiency, Smart buildings, Predictive maintenance, Supply chain, optimization

Before these developments, on the average and for the most part, you might have had to use two or more different languages or development environments to effectively build and operationalize a statistical model that scales. This lack of flexibility and agility meant that;

* Tremendous effort was required to work across the often complex and fragmented technologies instead of concentrating on the business objective at hand. This discouraged participation in data exploring, experimentation and model building.
* Many models that cost a lot of money and time to build never achieved the intended business value because it was so difficult to operationalize and deploy into a production environments.

Of course, today you can also do all of this in the cloud with Azure Machine Learning (using R or without writing any code), but this is out of scope in this series.

**The Language**

The R language was started by Ross Ihaka and Robert at the University of Auckland, New Zealand as a statistical computing and graphics tool.

It is a [GNU project](http://www.gnu.org/) similar to the S language and environment, which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can actually be considered as a different implementation of S. Even though there are some important differences R and S, much of code written for S will still runs under R.

**R Packages**

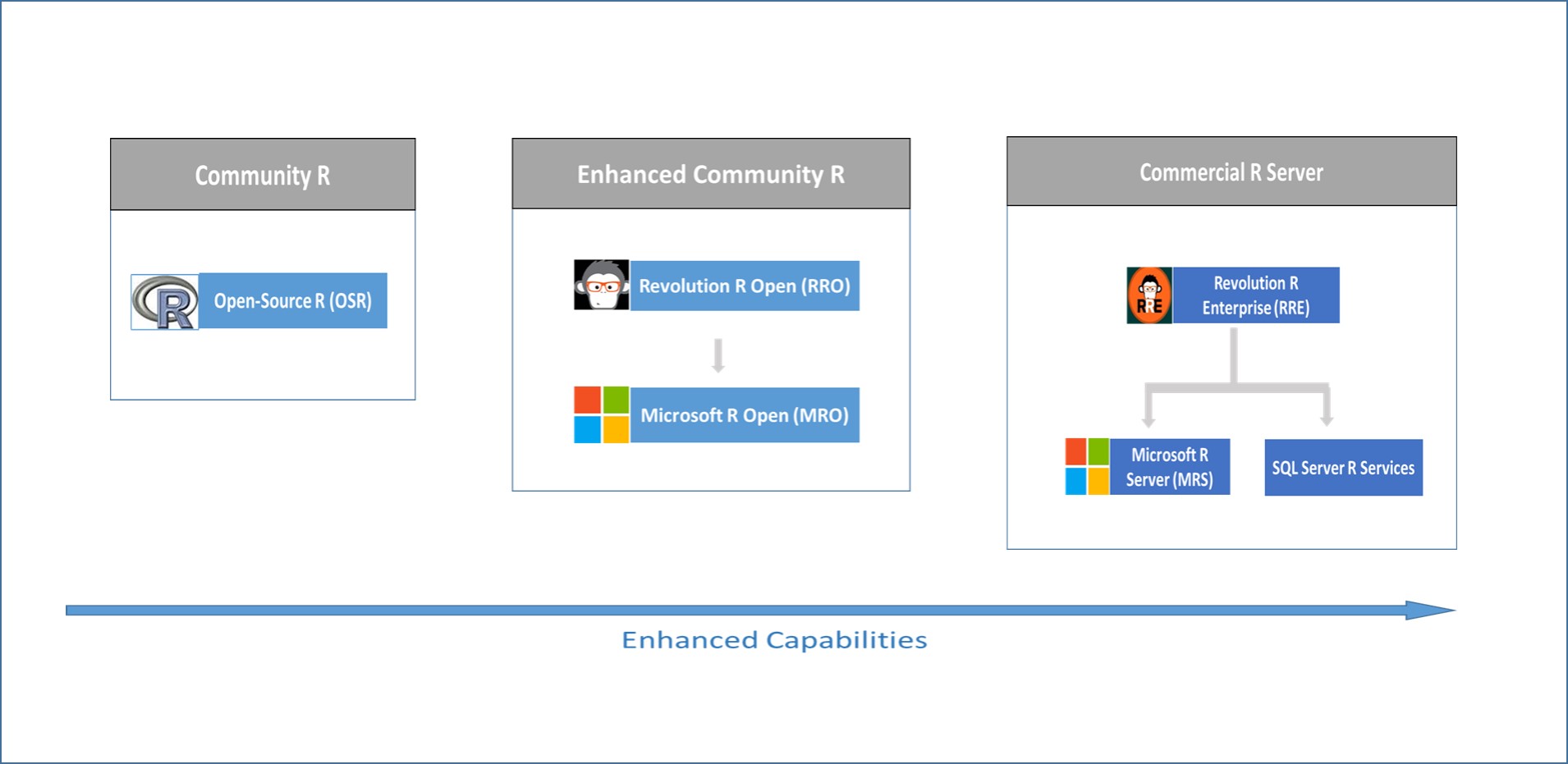
R packages are modules developed to extend the capabilities of the base R source code. The capabilities of these community-created extensions ranges from tailored statistical techniques and data exploration to graphical and reporting tools. Whatever operation you can think of in the statistical modeling life-cycle there is probably a package for it. Normally, a selected set of some core packages are bundled with the base R installation at every release. The over 7500 remaining packages (as of January 2016) that not bundled with the base installation are primarily stored in the Comprehensive R Archive Network (CRAN) and other repositories where they can be freely download.

**R Distributions**

At a high-level there are three major R distributions today:

1. **Opens-Source R (OSR**): A free and completely open-source distribution supported entirely by the R community.
2. **Microsoft R Open (MRO):** An enhanced version of the community distribution currently supported by Microsoft, which is also free.
3. **Microsoft R Server (MRS):** A commercial R Server distribution built on top of the enhanced community version designed for big data and heavy computations because of the limitations of the other two versions. MRS is ported Inside multiple platforms including Hadoop, Teradata, Linux and Azure Cloud.  For Windows, MRS ships as R Services in SQL Server 2016, which will be the center of our discussions in Part III of this series.

The following sections explain how the three distributions evolved to their present state. Note that the distribution represents enhancements on each, for instance MRO has all the functionalities and features of OSR and the commercial R Server (MRS) was built on top of MRO and therefore has all the its functionalities.



http://www.sqlservercentral.com/Resources/Images/zoom.gif[Zoom in](javascript:;)  |  [Open in new window](javascript:;)

*Figure 1: Evolution of the purely open-source and commercial version of R distributions.*

**Open-Source R (OSR)**

Open-Source R (OSR) , also referred to as Base R, is the purely community supported R, it open-source with the R community contributing the numerous modeling algorithm and machine-learning capabilities. The community is also responsible for the ecosystem of more than 7500 packaged algorithms, data exploration and model evaluations, and graphical modules that extend the capabilities of the base R.

R as a language had inherent performance and scalability limitations because it runs in-memory, besides the limitations associated with purely open-source software. The various limitations of OSR are as outlined below.

* In-memory based data access model limited performance.
* Lack of parallel computation to utilize processors and cores limited scaling and Big Data processing.
* Data movement from data sources into R environments results in latencies, duplicated cost and also posts security risk.
* The purely community based version limits enterprise utilization because of lack commercial support and/or extensions for enterprise users.
* Corporate governance oversight policies prevented many corporations to use open-source software.
* Complex deployment process prevented the realization of business value of often expensive models.

**Microsoft R Open (MRO)**

In 2007, Revolution Analytics (Revo) was founded to provide commercial support for R. Revo started an enhanced R distribution called Revolution R Open (RRO)  which added additional capabilities for improved performance, reproducibility, as well as support for Windows and Linux-based platforms.

After the Microsoft acquisition of Revo, RRO was later renamed Microsoft R Open (MRO) with more enhancements. Today MRO represents Microsoft Distribution of R with a high-performance engine which enables multi-threading as oppose to OSR. Other enhancement include support for Intel's Math Kernel Library of optimized math routines for science, engineering, and financial applications. It also comes with a reproducible R Toolkit, which ensures that results of R code execution are repeatable over time and that others who run the same code will achieve precisely the same results. It is fully compatibility with all packages, scripts and applications that work with OSR. Like OSR you are free to download, use, and share MRO.

**Microsoft R Server (MRS)**

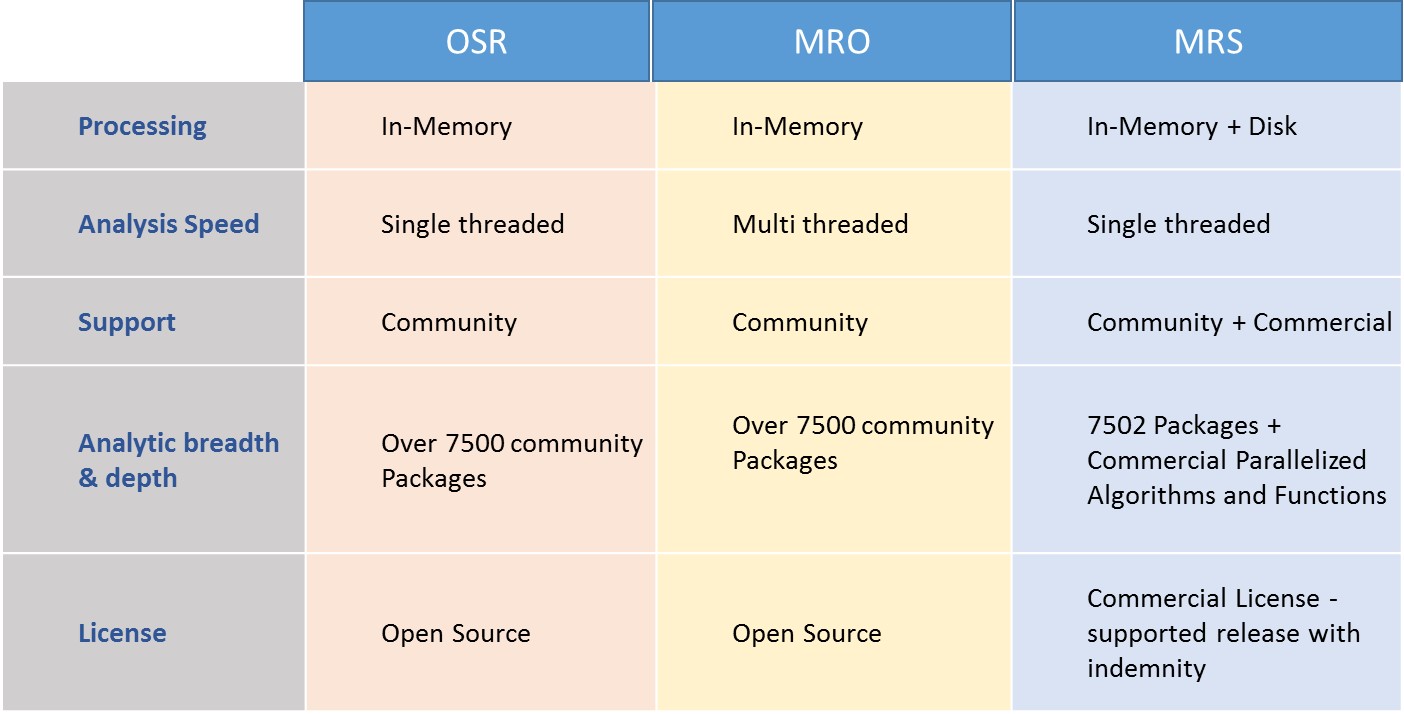
Even though RRO added some multi-threading functionalities to the OSR, it still had some of the OSR limitations because the processing was done in-memory. Revo also wanted to build a completely commercial and enterprise distribution to address these data size limitations besides the open-source ones. The resulting distribution was called Revolution R Enterprise (RRE) which added unique modules that included:

* RevoScaleR, a comprehensive library of big data analytics algorithms that support parallelization of computations and data analysis.
* RevoDeployR, a web services framework which web service based integration compatible with a broad array of tools

Enhancements continued after Microsoft's acquisition and RRE was later renamed Microsoft R Server (MRS). The name of the modules has also changed. The RevoScaleR module is now simply referred to as ScaleR and RevoDeployR as DeployR. MRS is further differentiated from the free, open distributions by availability of 24x7 support.

R Server is currently ported inside multiple platforms including Hadoop, Teradata, Linux and the Azure Cloud. Because R Server is built around RRO, it provide an enhanced experience for the R User without loss of compatibility. R Scripts built for one platform using R Server can be easily run on another platform running R Server i.e., *Write Once, Deploy Anywhere* (WODA). Develop on the desktop and immediately deploy to RDBMS – SQL Server, EDW (SQL Server & Teradata), Hadoop (Microsoft, Cloudera, Hortonworks and MapR) and Azure (HDInsight and as VM)

Table 1 below shows a side by side look at key features  of the three R distributions.



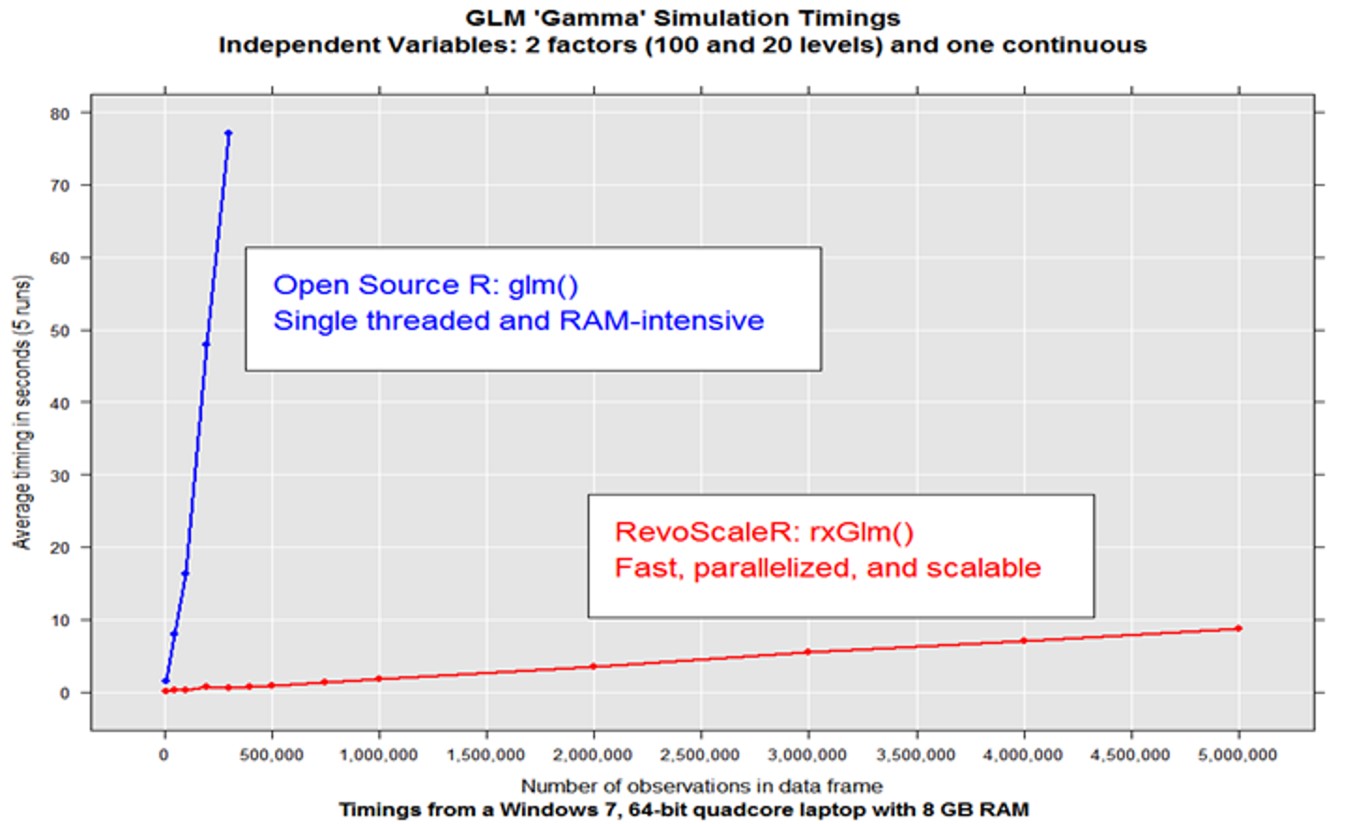
http://www.sqlservercentral.com/Resources/Images/zoom.gif[Zoom in](javascript:;)  |  [Open in new window](javascript:;)

*Table 1: shows side-by-side comparison of the three distributions.*

**MRS Vs OSR - Performance comparison**

Figure 2 below shows the results of a benchmark test comparing the performance of OSR and a ScaleR algorithm. as can be seen on the fgure, When OSR operates on data sets that exceed RAM (about 300K observations in a dataframe), it will fail.

On the other hand it can be seen from the plot that ScaleR has no data size limit in relation to the size of the RAM. The ScaleR algorithm is seen to scales linearly well beyond the limits of the RAM (over 5M observations in a dataframe) and the parallel algorithms are much faster.

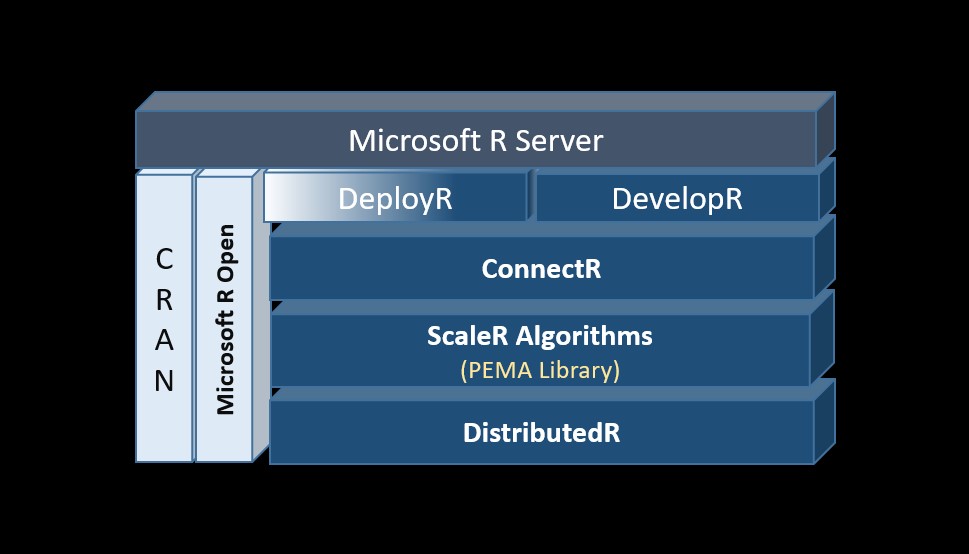


http://www.sqlservercentral.com/Resources/Images/zoom.gif[Zoom in](javascript:;)  |  [Open in new window](javascript:;)

*Figure 2: Showing Performance between OSR and a ScaleR algorithm in an Advanced analytics simulations.*

**MRS Architecture**

As can be seen from Figure 3 below, MRS is built on MRO, but the underlying technologies that makes it capable of handling a massive amount of data and Enterprise Scale advanced analytics are ScaleR, DistributedR, and ConnectR.



http://www.sqlservercentral.com/Resources/Images/zoom.gif[Zoom in](javascript:;)  |  [Open in new window](javascript:;)

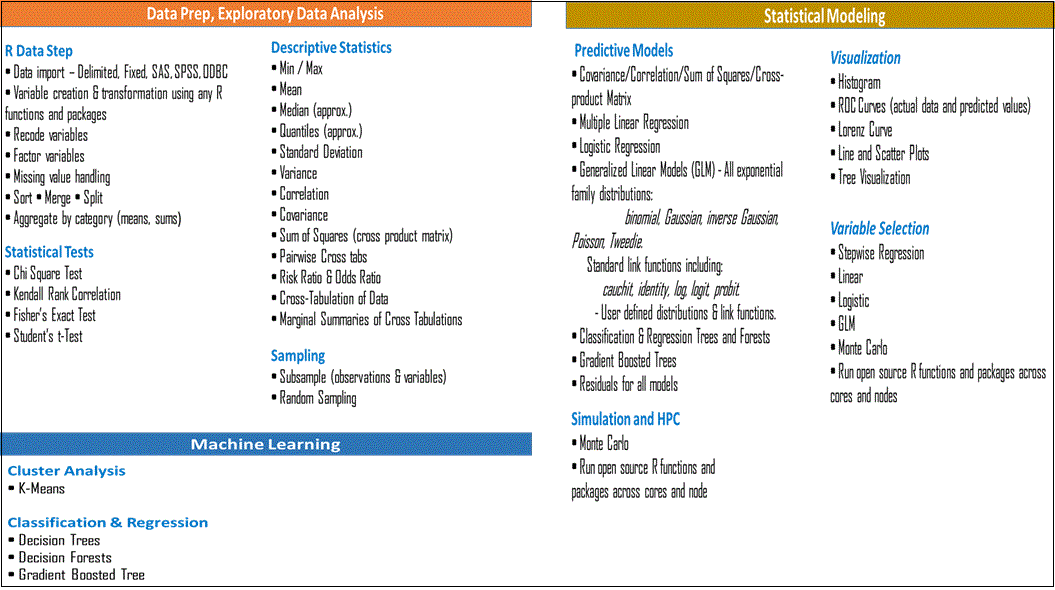
*Figure:3 Showing the various components in MRS Architecture.*

**ScaleR**

ScaleR proprietary package distributed with MRS. It provides tools for both High Performance Computing (HPC) and High Performance Analytics (HPA) with R.  HPC capabilities allow you to distribute the execution of essentially any R function across cores and nodes, whiles HPA enables analytic computation on ‘big data’.

The framework consist of algorithms and functions written in C++ and implemented as a library of optimized Parallel External Memory Algorithms (PEMA), which manages available RAM and storage together. They do not require all data to be in RAM, data is processed in parallel and sequentially, a chunk at a time, with intermediate results produced for each chunk. Intermediate results for one chunk of data can be combined with those of another chunk, allowing distribution of the execution across cores and nodes.

The Algorithms and functions  supports data preparation, descriptive statistics and exploratory Data analysis, Statistical Modeling and machine learning capabilities as outlined on Figure 4 below.



http://www.sqlservercentral.com/Resources/Images/zoom.gif[Zoom in](javascript:;)  |  [Open in new window](javascript:;)

*Figure:4 Showing categorized parallelized Algorithms and functions .*

ScaleR is also equipped with open database connectivity (ODBC) drivers and other connector which enable it to integrate with SQL Server 2016 databases, Hadoop (Cloudera, Hortonworks, or MapR), HDInsight clusters with Spark and Teradata data warehouses. Besides providing support for data import, data transformations, sorting, merging etc., it also It enables read and write to the ScaleR .xdf data file format on these platforms. XDF files are 5 times smaller than say csv and can therefore enhance performance significantly.

**DistributedR**

DistributedR provides the parallel execution framework that enables ScaleR algorithms to analyze vast data sets and scale from single-processor workstations to clustered systems with hundreds of servers. Also included in the frameworks are services like communications, storage integration and memory management.

The platform exposes distributed data structures such as arrays, data frames, lists and loops to store data across a cluster.

Machine Learning algorithms and graph algorithms primarily use matrix and adjacency matrix operations respectively, therefore for these, arrays act as a single abstraction to efficiently implement them. Using Distributed R constructs, data can be loaded in parallel from any data source.

**ConnectR**

ConnectR provides access to the many compatible data source ranging from simple workstation file systems to complex distributed file systems and MPP databases. These high-speed & direct connectors are available for high-performance XDF, SAS, SPSS, delimited & fixed format text data files, Hadoop HDFS (text & XDF), Teradata Database & Aster,  EDWs and ODBC.

**DeployR**

The DeployR is the framework that is used operationalize R models, making it possible to add the power of R Analytics to any application, dashboard, backend etc. DeployR enables a web service based integration compatible with a broad array of tools, converting R scripts into [analytics web services](https://deployr.revolutionanalytics.com/documents/getting-started/about/#g_analyticswebservice), so R code can be easily executed by applications running on a secure server.

**Conclusion**

In Part I we have learned about how the R evolved and how the community distributions (OSR and MRO) fail in operations on data sets that exceed RAM due to their in-memory data access model.

We also learned that, unlike the community distributions, MRS (which enjoys all the benefits of the community distributions) is able to scales linearly well beyond RAM limits with exceptional performance. It does this by processing advanced analytics workload on chunks of data sequentially and in parallel across numerous nodes in a cluster made possible by scaleR package. MRS ability to push processing to interoperable databases (SQL server, Hadoop, Teradata) means it is able to work and utilize cores on commodity hardware in the process. This ability to execute remotely (in-database, in-MapReduce and in-Apache Spark) and the integrated DeployR web services framework makes it aslso possible now to quickly operationalize scalable models from the same development environment.

By capitalizing on the flexible and agility provided by MRS, businesses can now run experiments, build and operationalize scalable advanced analytics (data science, machine learning) solutions and intelligent applications with costs keep low and contained and with minimum investment required to train developers in R.

**Next**

In the next installments of this series, we will look at the various data science scenarios and context under which the R distributions can be used with SQL Server and then subsequently look at how to build and operationalize a predictive model entirely in SQL server 2016.