

# LifeStats: An Interactive Environment for Teaching Statistics

J. Harner   D. Luo   J. Tan

Department of Statistics  
West Virginia University

Interface 2008  
Risk : Reality

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts a a front-end to R.

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts as a front-end to R.

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts a a front-end to R.

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts a a front-end to R.

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts as a front-end to R.

# LifeStats

## LifeStats:

- is an interactive book with embedded (popup) applets;
- has interactive notes with embedded (popup) applets;
- has a five-step simulator;
- illustrates probability and statistical concepts using applets;
- provides an interactive data analysis environment;
- acts as a front-end to R.

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# Assessment of Student Learning

## Student Scores

- Scores are simple and straightforward, but they provide little information on how to improve student learning.
- We need a way to provide personalized, targeted, and automated feedback to students.

# Knowledge Structures

- The fundamental units of knowledge consist of discrete cognitive attributes.
- Meta-cognitive attributes are built on fundamental cognitive attributes.
- Cognitive attributes are organized hierarchically.

# Measuring Knowledge

Questions provide an indirect measure of knowledge:

- A question is linked to one or more cognitive attribute(s)
- A correct answer to a question provides evidence that the associated cognitive attributes have been learned.
- The probability of mastery of an attribute, given a student's scores, is estimated.

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# Hierarchical Bayesian Cognitive Models

The cognitively diagnostic assessment system builds on the DINA (Deterministic Input; Noisy “And” Gate) and NIDA (Noisy Input; Deterministic “And” Gate) models.

- The DINA model is a simplified form of Anderson’s “knowledge tracing model.”
- The DINA model is relatively simple and forms the basis of several cognitive models, including ALEKS (<http://www.aleks.com>).
- The NIDA model is a relatively simple discrete attribute model that provides a foundation for cognitively more complex models. Specifically, the NIDA model can be augmented by multiplicative or additive IRT terms.

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

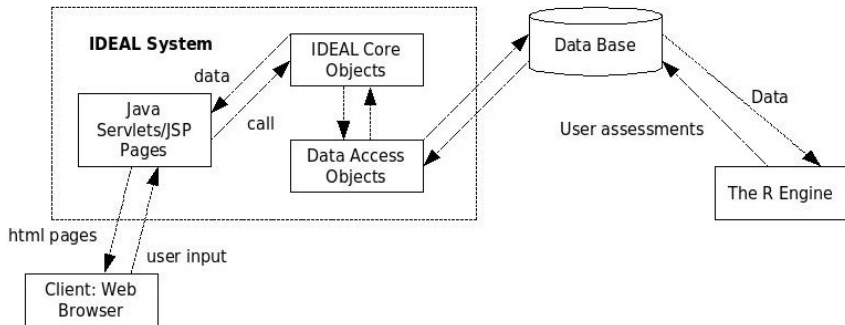
# IDEAL

(Intelligent Distributed Environemnt for Adaptive Learning)

## IDEAL:

- is an online course system developed and used by the WVU Dept. of Statistics;
- is a multi-tier Java web application based on open-source software;
- uses advanced Java frameworks such as Spring and Hibernate
- will provide personalized cognitive feed back.

# IDEAL/R Architecture



# Defining the Knowledge Structure

Two layers of abstraction: modules and cognitive attributes within modules

- Problem: Too many cognitive attributes with too little data (questions and student answers)
- Solution: Group cognitive attributes as the basic units of analysis (estimation)

# Screen Shot Defining Module Dependences

The screenshot shows a window titled "DiagramEditor v0.1" with a diagram on the left and a configuration panel on the right. The diagram shows a node "Probability Laws" (highlighted in blue) with arrows pointing to four other nodes: "Central Limit Theorem", "Normal Distribution", "Poisson Distribution", and "Binomial Distribution".

The configuration panel on the right has the following sections:

- Node:** A dropdown menu showing "Probability Laws".
- Out Edges:** A list of nodes: "Central Limit Theorem", "Normal Distribution", "Binomial Distribution", and "Poisson Distribution". The "Central Limit Theorem" entry is highlighted in blue. Below the list are "+" and "-" buttons.
- In Edges:** A list of nodes: "Probability Definitions" and "Calculations". Below the list are "+" and "-" buttons.
- At the bottom are "Update" and "Cancel" buttons.

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# Using R as a Background Computational Engine

- Difficult to build statistical software from scratch
- Simplify the process by building the software on R
- Launch R in batch mode
- Share data through a database
  - Access databases from Java using JDBC
  - Access databases from R using RMySQL

# Using R as a Background Computational Engine

How and when to launch R

- Periodically launch R (e.g., using a cron tab under unix)
- Launch R on demand (e.g., waiting for a request from IDEAL)

Other issues:

- Operational vs. analytical system
- Data pre-processing may be necessary before launching R

# JavaStat

## JavaStat:

- is a highly interactive data analysis environment;
- is a Java front-end for R;
- uses R as a compute engine (the basic version);
- allows the user to have a permanent workspace (the enhanced version);
- has a common code base with LifeStats.

# JavaStat

## JavaStat:

- is a highly interactive data analysis environment;
- is a Java front-end for R;
- uses R as a compute engine (the basic version);
- allows the user to have a permanent workspace (the enhanced version);
- has a common code base with LifeStats.

# JavaStat

## JavaStat:

- is a highly interactive data analysis environment;
- is a Java front-end for R;
- uses R as a compute engine (the basic version);
- allows the user to have a permanent workspace (the enhanced version);
- has a common code base with LifeStats.

# JavaStat

## JavaStat:

- is a highly interactive data analysis environment;
- is a Java front-end for R;
- uses R as a compute engine (the basic version);
- allows the user to have a permanent workspace (the enhanced version);
- has a common code base with LifeStats.

# JavaStat

## JavaStat:

- is a highly interactive data analysis environment;
- is a Java front-end for R;
- uses R as a compute engine (the basic version);
- allows the user to have a permanent workspace (the enhanced version);
- has a common code base with LifeStats.

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# Communicating with R

- R batch mode (stdin/out/err)
- TCP/IP sockets
- C/Fortran interface
- JRI (Java/R Interface)

# JRI (Java/R Interface)

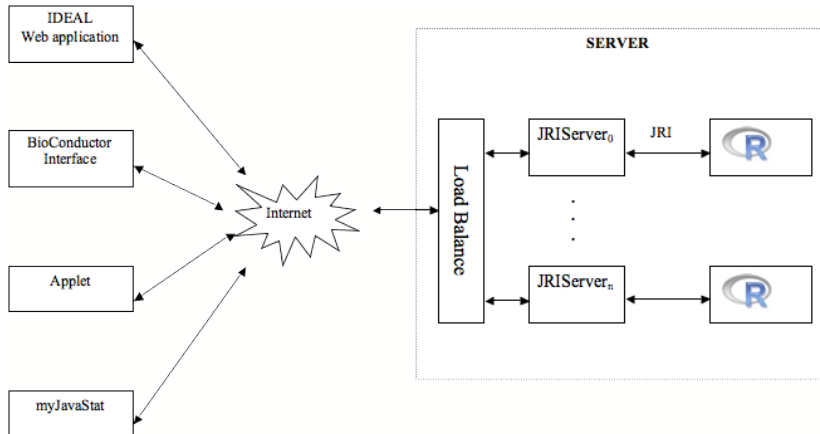
JRI:

- runs R within Java as a single thread  
(<http://rosuda.org/JRI/>)
- loads the R dynamic library and provides a Java API to R
- converts R objects (limited) into Java objects

# Outline

- 1 LifeStats
  - Evaluating Learning Performance
  - Cognitive Model
- 2 IDEAL
  - IDEAL Architecture
  - R as a Background Compute Engine
- 3 JavaStat
  - Java/R Connections
  - JavaStat/R Architecture

# JavaStat/R Basic Architecture



# JavaStat/R Basic Architecture

The following are the principal characteristics of the JavaStat/R basic architecture:

- R is on the server.
- R results are wrapped to increase efficiency and speed
- Multiple servers with multiple instances of R
  - running in parallel with separate workspaces
- Load balanced
- Stateless
  - no relationship among consecutive requests

# JavaStat/R Basic Architecture

The following are the principal characteristics of the JavaStat/R basic architecture:

- R is on the server.
- R results are wrapped to increase efficiency and speed
- Multiple servers with multiple instances of R
  - running in parallel with separate workspaces
- Load balanced
- Stateless
  - no relationship among consecutive requests

# JavaStat/R Basic Architecture

The following are the principal characteristics of the JavaStat/R basic architecture:

- R is on the server.
- R results are wrapped to increase efficiency and speed
- Multiple servers with multiple instances of R
  - running in parallel with separate workspaces
- Load balanced
- Stateless
  - no relationship among consecutive requests

# JavaStat/R Basic Architecture

The following are the principal characteristics of the JavaStat/R basic architecture:

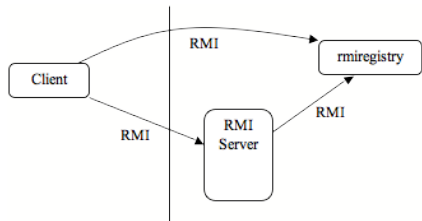
- R is on the server.
- R results are wrapped to increase efficiency and speed
- Multiple servers with multiple instances of R
  - running in parallel with separate workspaces
- Load balanced
- Stateless
  - no relationship among consecutive requests

# JavaStat/R Basic Architecture

The following are the principal characteristics of the JavaStat/R basic architecture:

- R is on the server.
- R results are wrapped to increase efficiency and speed
- Multiple servers with multiple instances of R
  - running in parallel with separate workspaces
- Load balanced
- Stateless
  - no relationship among consecutive requests

# Java RMI



## Remote Method Invocation

- client-server architecture
- pure Java-to-Java talk
- rmiregistry (for server reference list)
- load balancing

# JavaStat/R Basic Architecture vs. Rserve

	JavaStat	Rserve
R backend	✓	✓
Connection	RMI	TCP/IP
Language	Java	C/C++, Java
Transmission	Wrap up many commands	Each command
Purpose	App dependent	General

Table: Comparing JavaStat/R to Rserve

# JavaStat/R Interface Limitations

JavaStat/R currently cannot be used as a general interface.

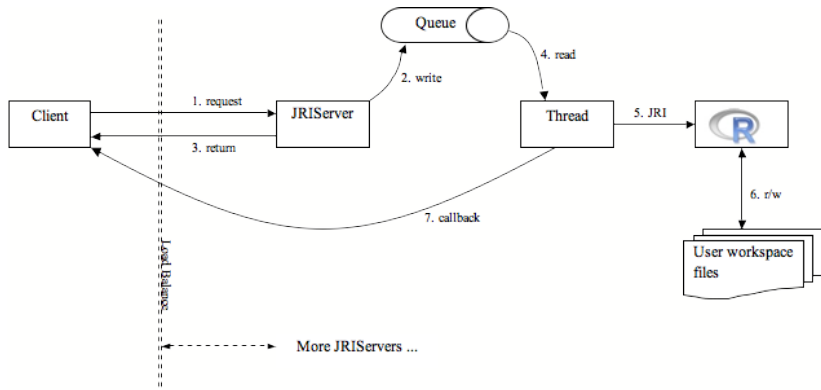
- Drawbacks:
  - Time consuming for complex models
  - Large datasets not feasible
- Solution:
  - Use RMI Callback.
  - Save & load workspace on server.

# JavaStat/R Interface Limitations

JavaStat/R currently cannot be used as a general interface.

- Drawbacks:
  - Time consuming for complex models
  - Large datasets not feasible
- Solution:
  - Use RMI Callback.
  - Save & load workspace on server.

# JavaStat/R Enhanced Architecture



# Summary

- LifeStats is a powerful, interactive environment for learning statistics.
- LifeStats is a front-end to IDEAL/R.
- IDEAL/R is a statistical assessment system and potentially a statistical tutoring system.
- JavaStat/R is a powerful, interactive data analysis environment.
- The enhanced version of JavaStat/R is an interactive R front-end for complex models and large datasets, e.g., analyses using Bioconductor.