Automated Trading with MATLAB®

Stuart Kozola
Computational Finance

© 2012 The MathWorks, Inc.

Challenges when developing and implementing trading strategies and systems

- Increasing complexity
  - More data
  - More complicated models
  - Integration and implementation to multiple system
- Increasing computational speed
  - Push to higher frequency
  - Changes in hardware
- Long deployment cycle
  - (Re)coding is costly and error-prone
  - Multiple system interfaces to consider

Agenda

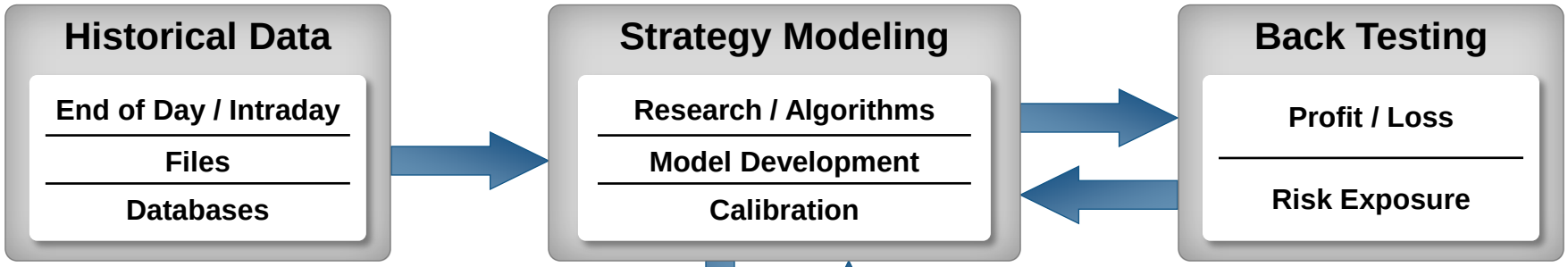
- Trading system demonstration
- Developing an automated trading decision engine
  - Identify a successful trading rule
  - Extend trading rule set
  - Automate trading rule selection
- Implementing an automated trading system
  - Testing (paper trading) the system
  - Integration and execution through a desktop trading system
  - Execution through a server based system (FIX)
- Wrap up and Q&A

The problem at hand: Identifying and implementing profitable trading strategies

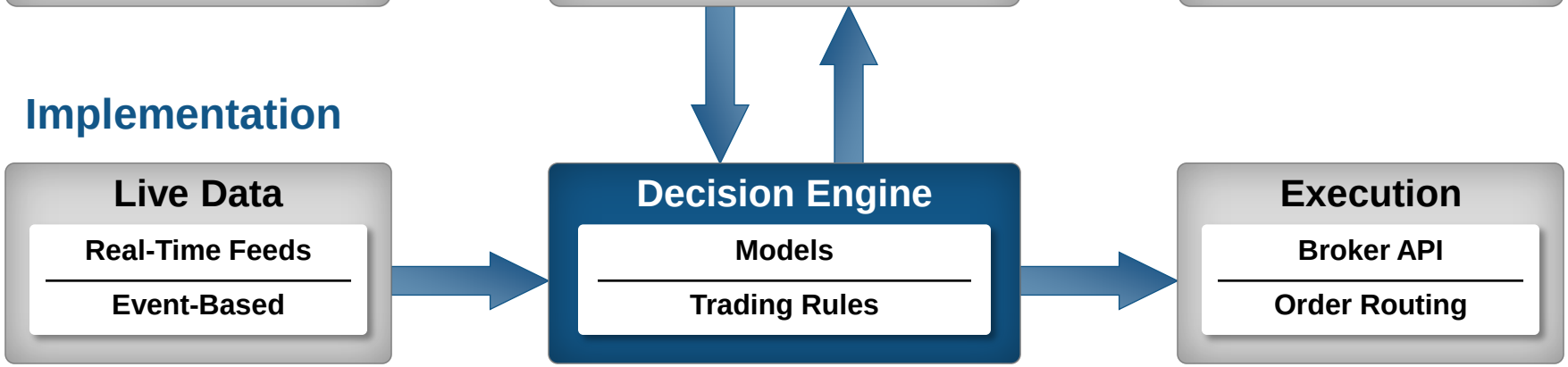
- Commodities analyst
- Developing a trading strategy
  - Multiple trading rules
  - High frequency
- Management requirements:
  - Tested on historical data
  - Uses sophisticated analytics to identify optimal trading rule combination
  - Integrates with existing data and execution APIs

Trading decision engine

Development and testing



Implementation

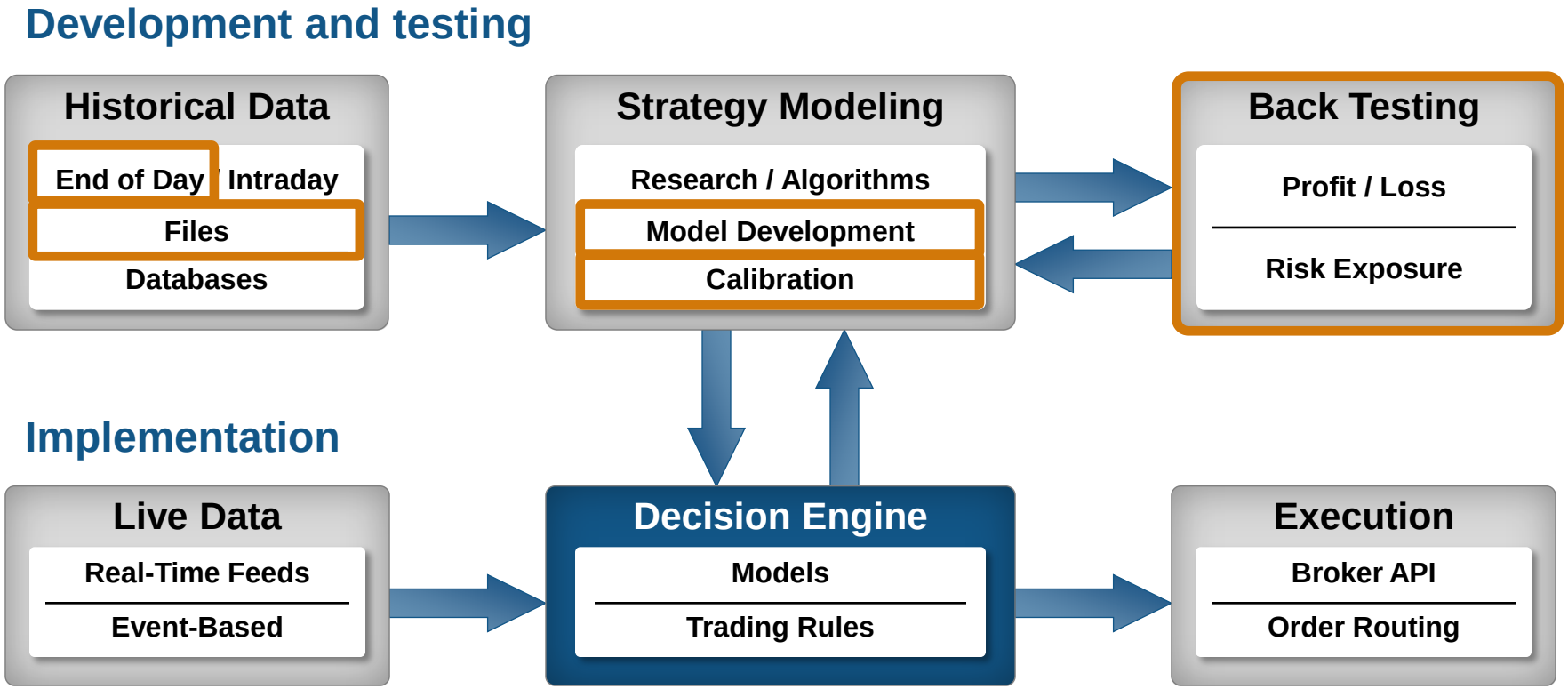


Requirements for the trading engine

- Sophisticated analytics
  - Custom rules & indicators
  - Non-traditional techniques
- Scalable speed
  - Higher frequency data
  - More trading rules
- Quick to develop and deploy
  - Try different strategies
  - Embed in trading engine

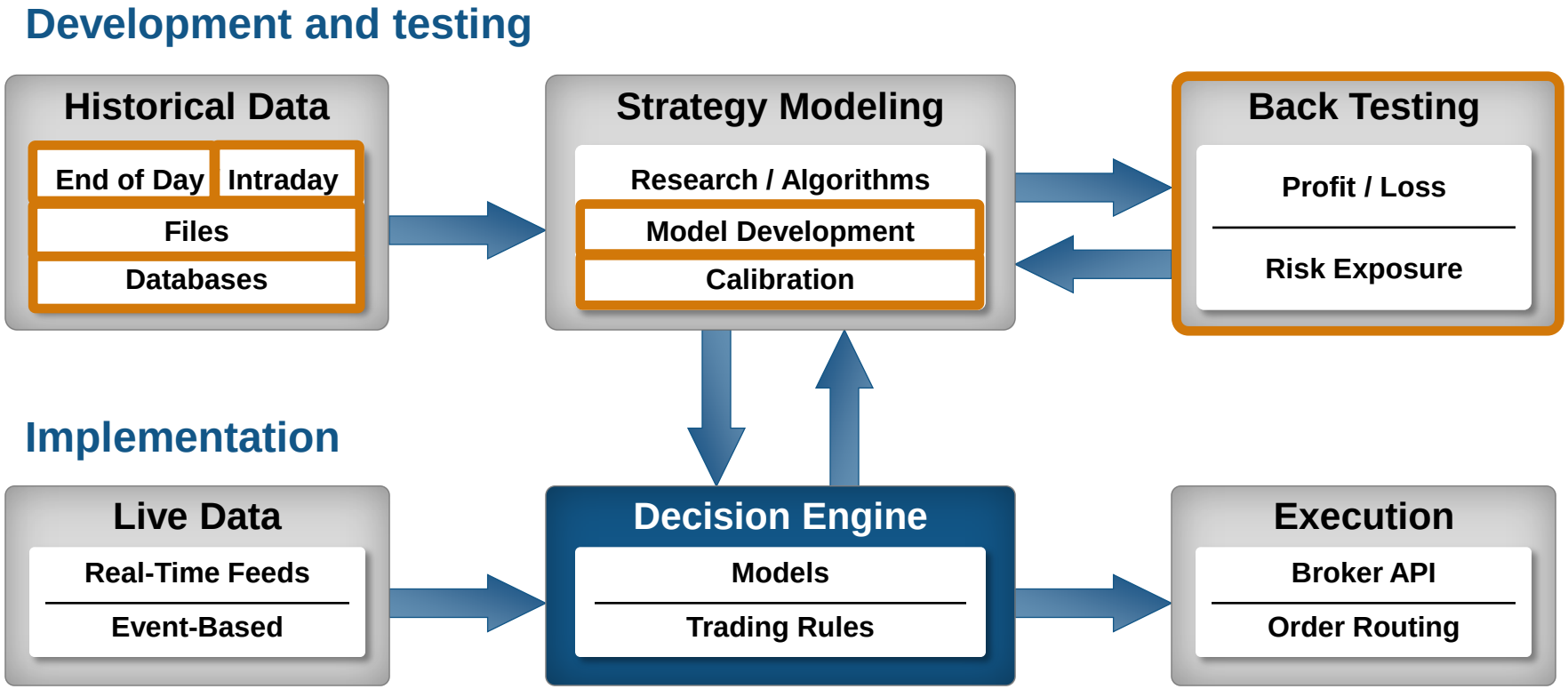
Task 1: Develop a back testing environment

Goal: Build a back testing environment around historical data and a preliminary trading rule



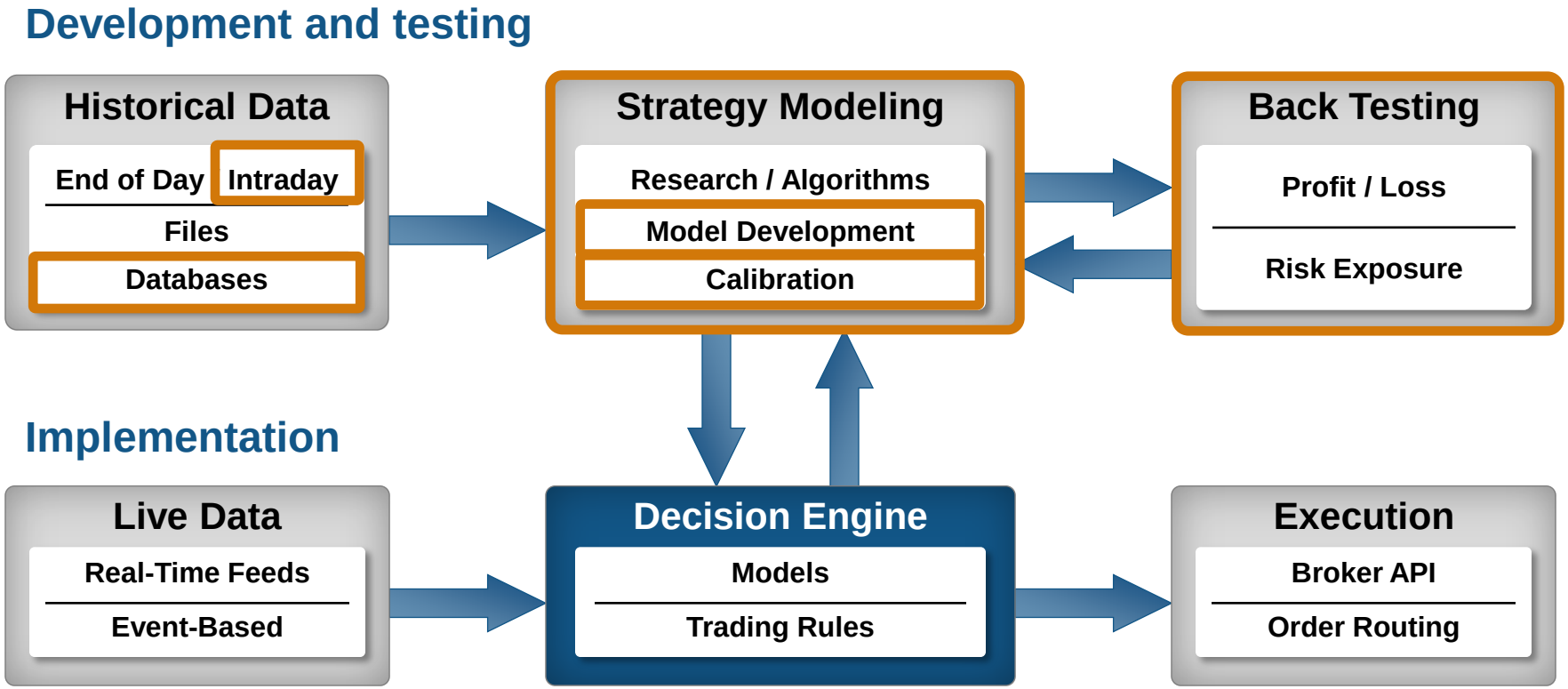
Task 2: Expand to a higher frequency environment

Goal: Move to a higher frequency (minute-by-minute) and re-calibrate the model



Task 2: Expand the decision logic of the engine

Goal: Develop a rule selection system for instruments using evolutionary learning

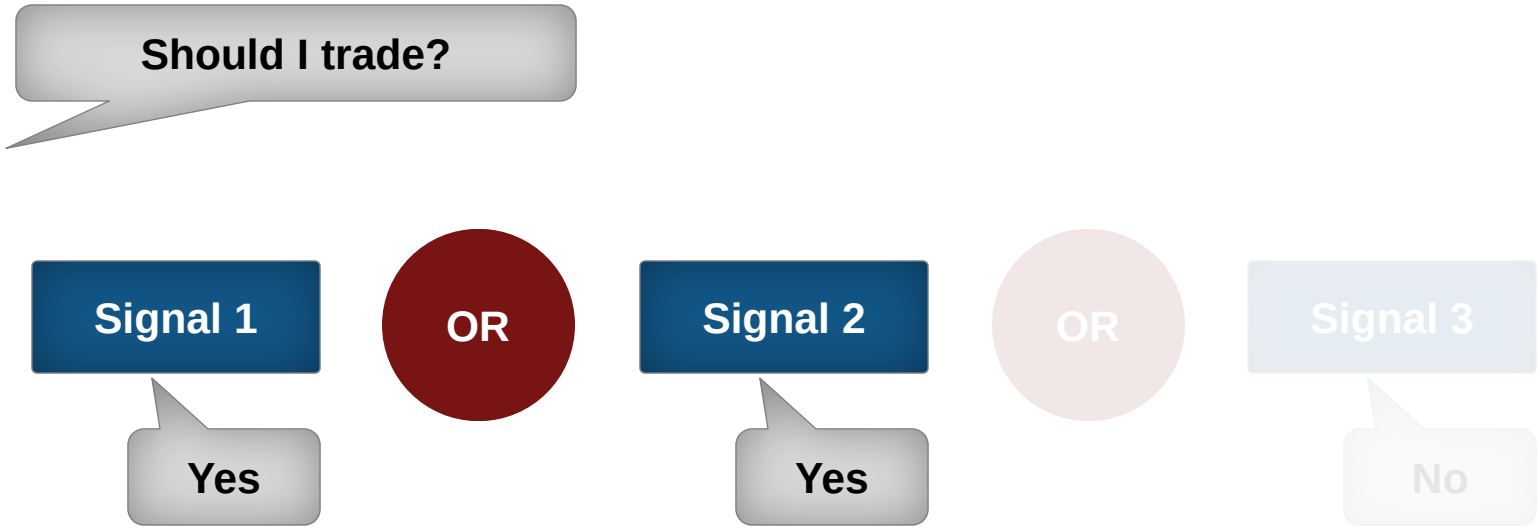


Key tasks

Key tasks

- Increase number of rules
- Incorporate advanced analytics to select best combination

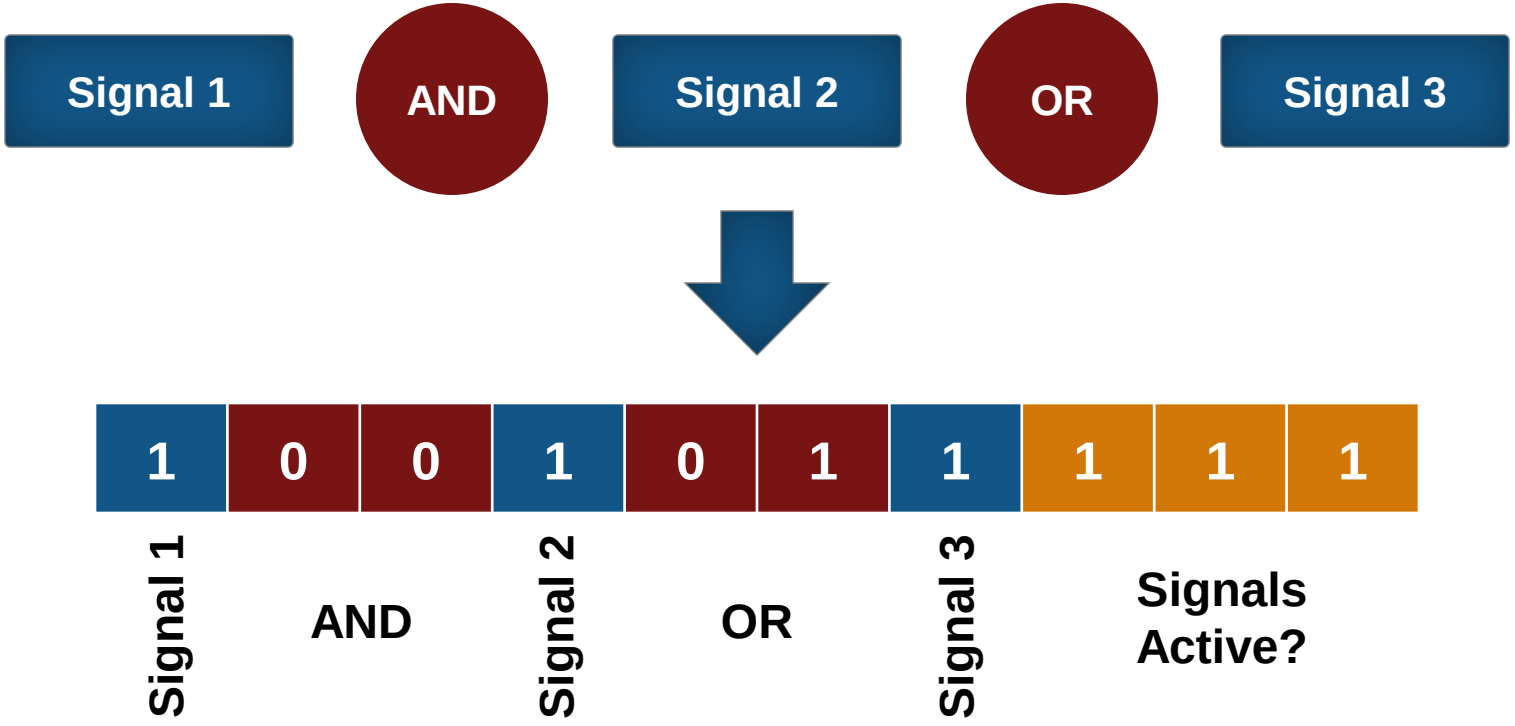
Working with multiple strategies



Dempster et. al., *Computational learning techniques for intraday fx trading using popular technical indicators*, IEEE Transactions on Neural Networks (2001).

Working with multiple strategies

- Represent different combinations as *bit strings*

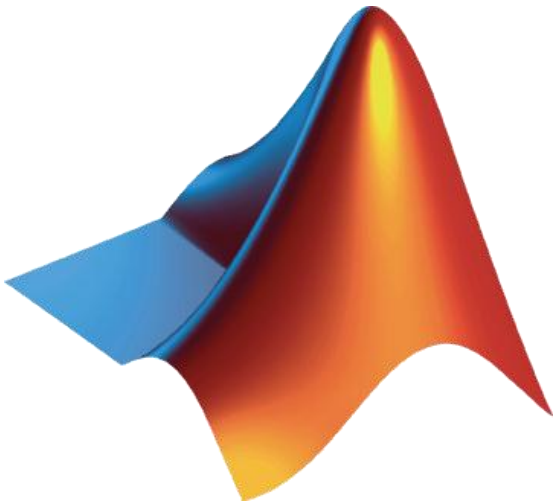


Building Custom Evolution Algorithms

- Selection
  - *Retain* the best performing bit strings from one generation to the next. *Favor these for reproduction*

- Crossover
  - parent1 = [1 0 1 0 0 1 1 0 0 0]
  - parent2 = [1 0 0 1 0 0 1 0 1 0]
  - child = [1 0 0 0 0 1 1 0 1 0]

- Mutation
  - parent = [1 0 1 0 0 1 1 0 0 0]
  - child = [0 1 0 1 0 1 0 0 0 1]



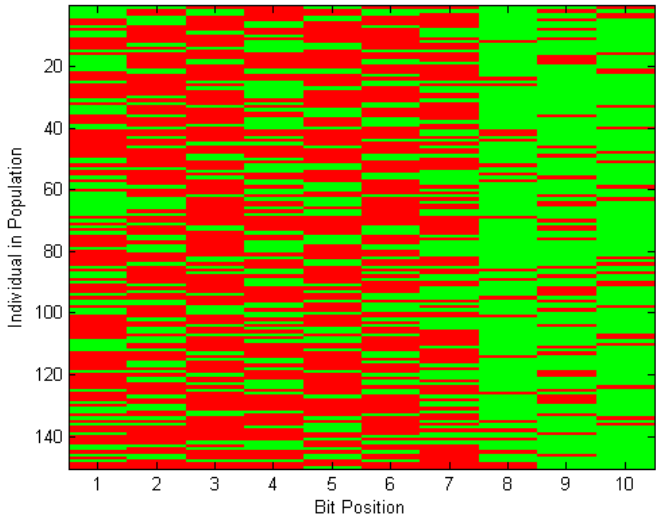
Key tasks

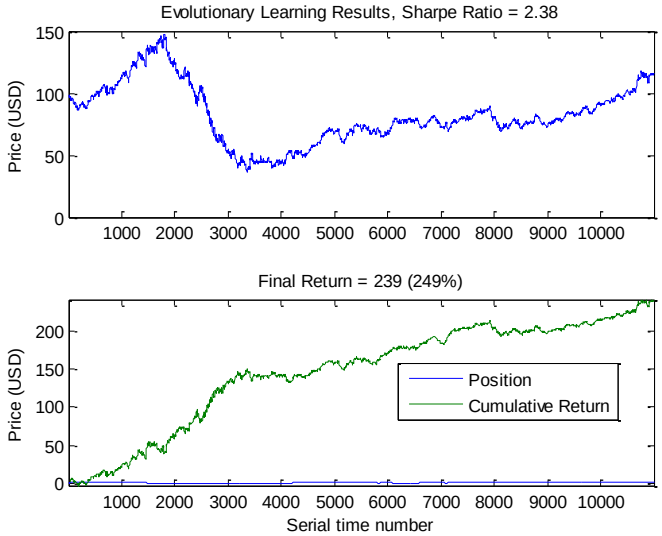
Key tasks

- Increase number of rules
- Incorporate advanced analytics to select best combination

Solutions

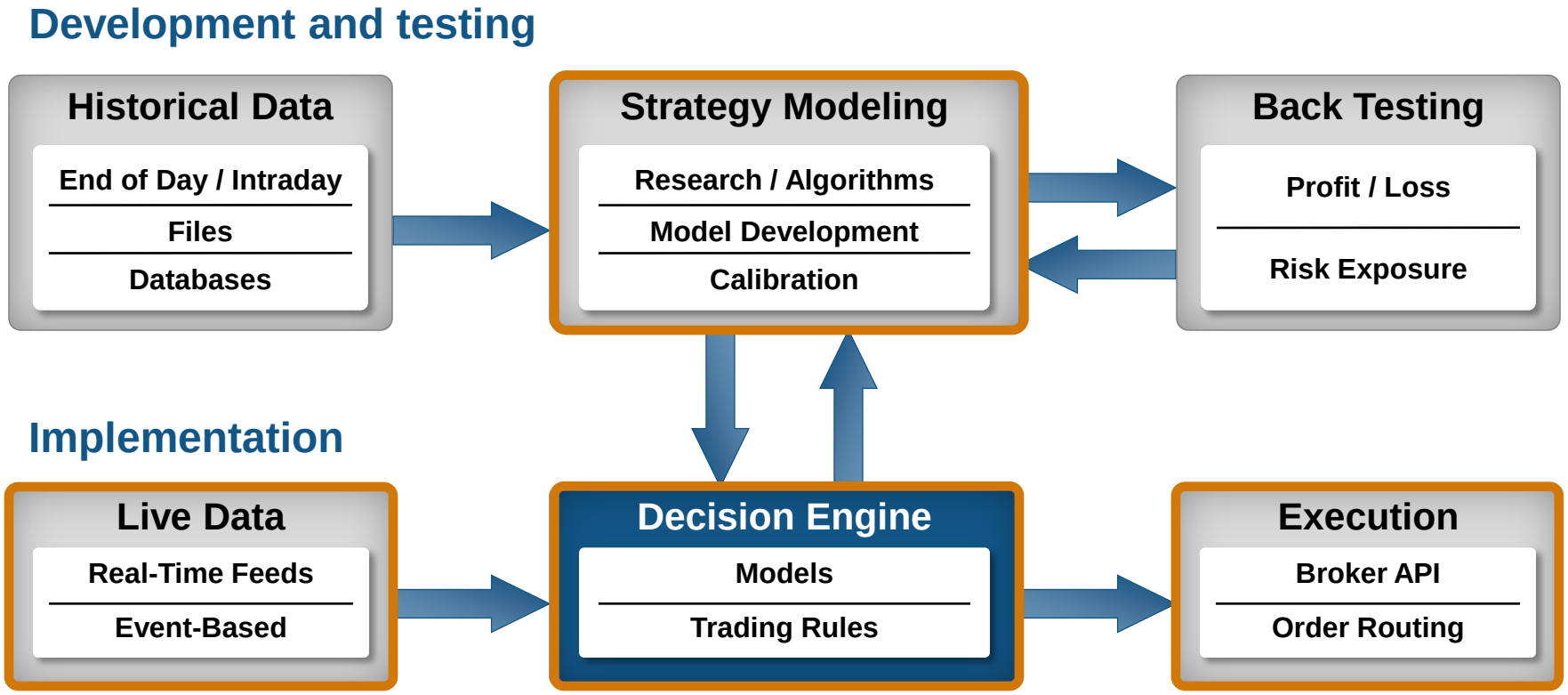
- High-level programming
- MATLAB Toolboxes: Global Optimization, …





Task 4: Integrate into real-time trading system

Goal: Integrate the trading decision engine into real-time simulation environment for “paper trading”

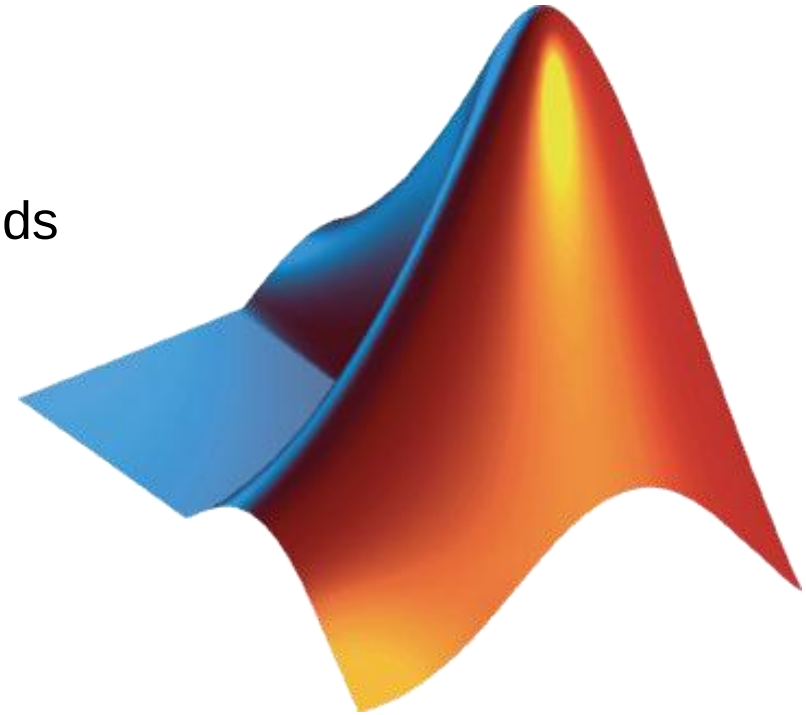


Review: Requirements for the trading engine

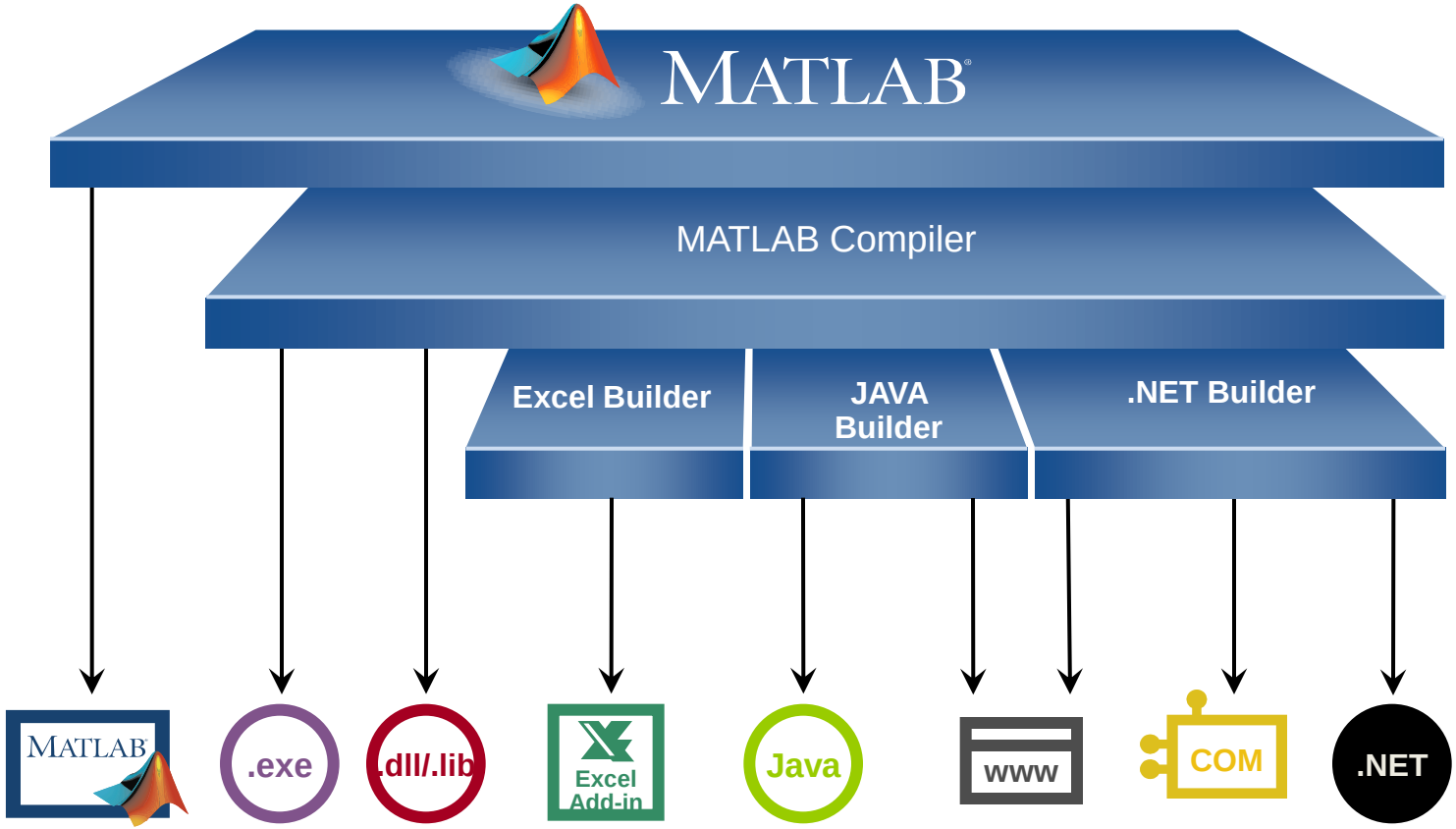
- Sophisticated analytics
  - Custom rules & indicators
  - Non-traditional techniques
- Scalable speed
  - Higher frequency data
  - More trading rules
- Quick to develop and deploy
  - Try different strategies
  - Embed in trading engine

MATLAB's solutions

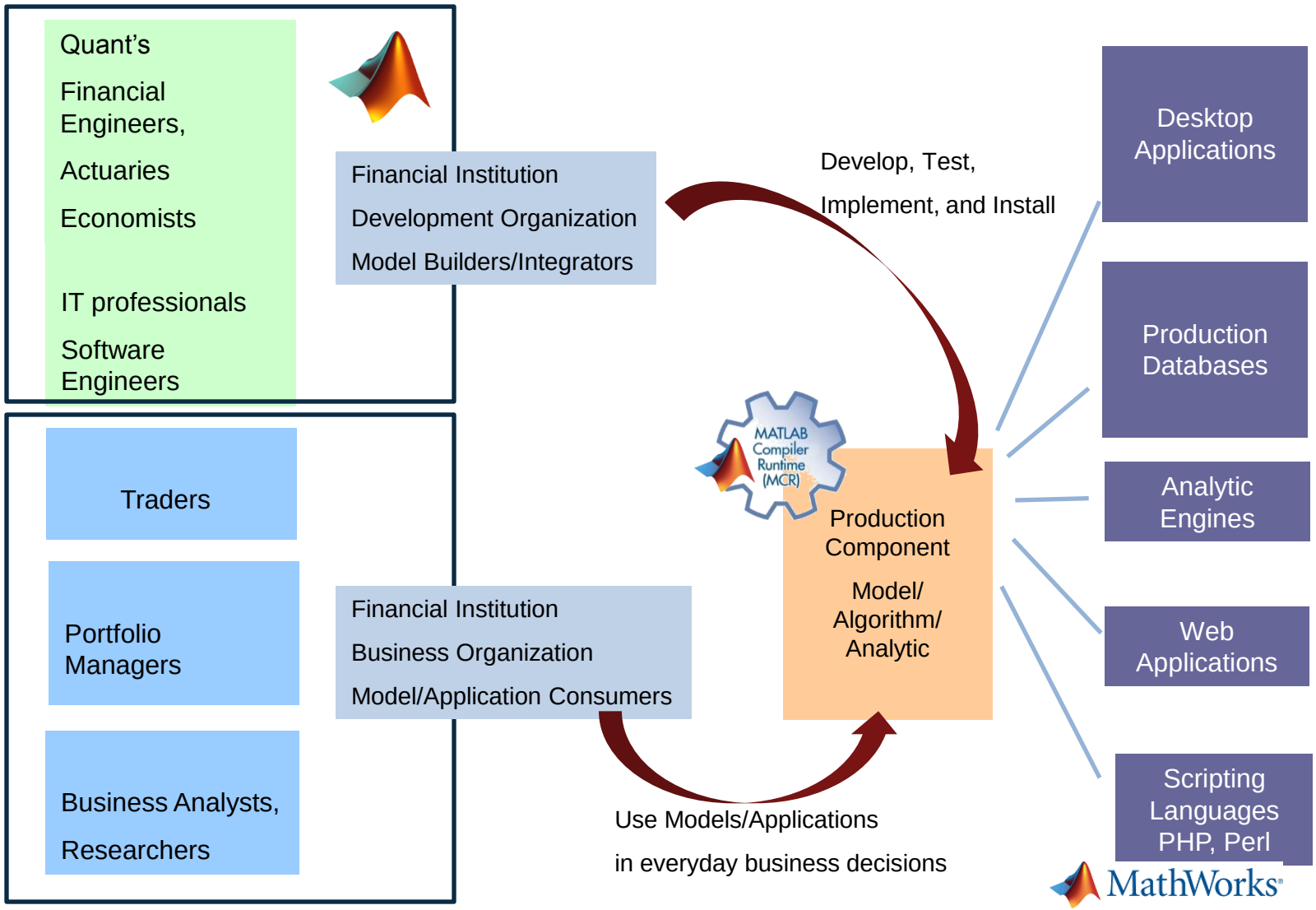
- Sophisticated analytics
  - Advanced graphics environment
  - Toolboxes give access to hundreds of new techniques
  - Flexible and customizable
- Scalable speed
  - Parallel computing solution
- Quick to develop and deploy
  - High-level programming
  - Automated deployment to desktop or server-based systems



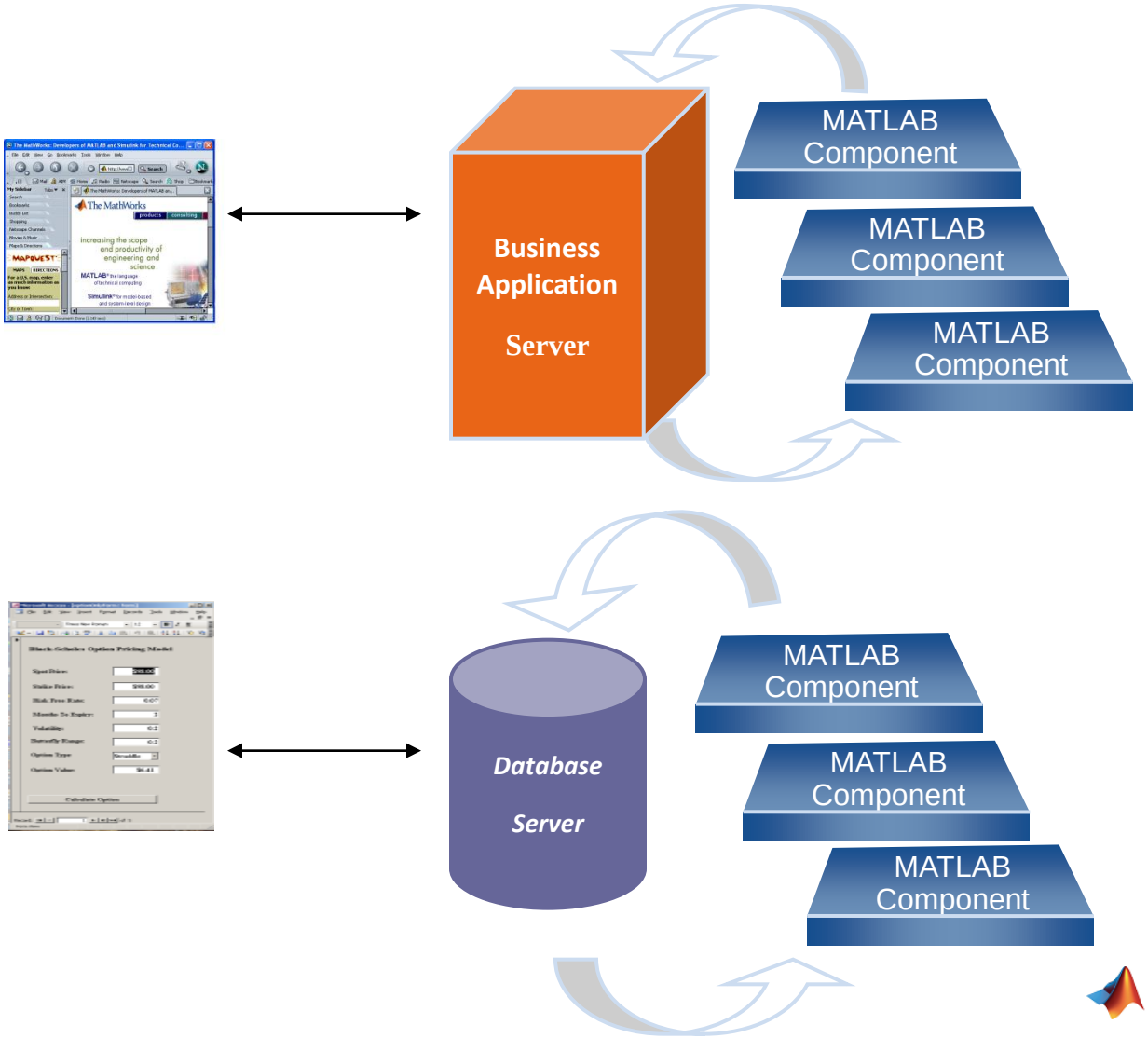
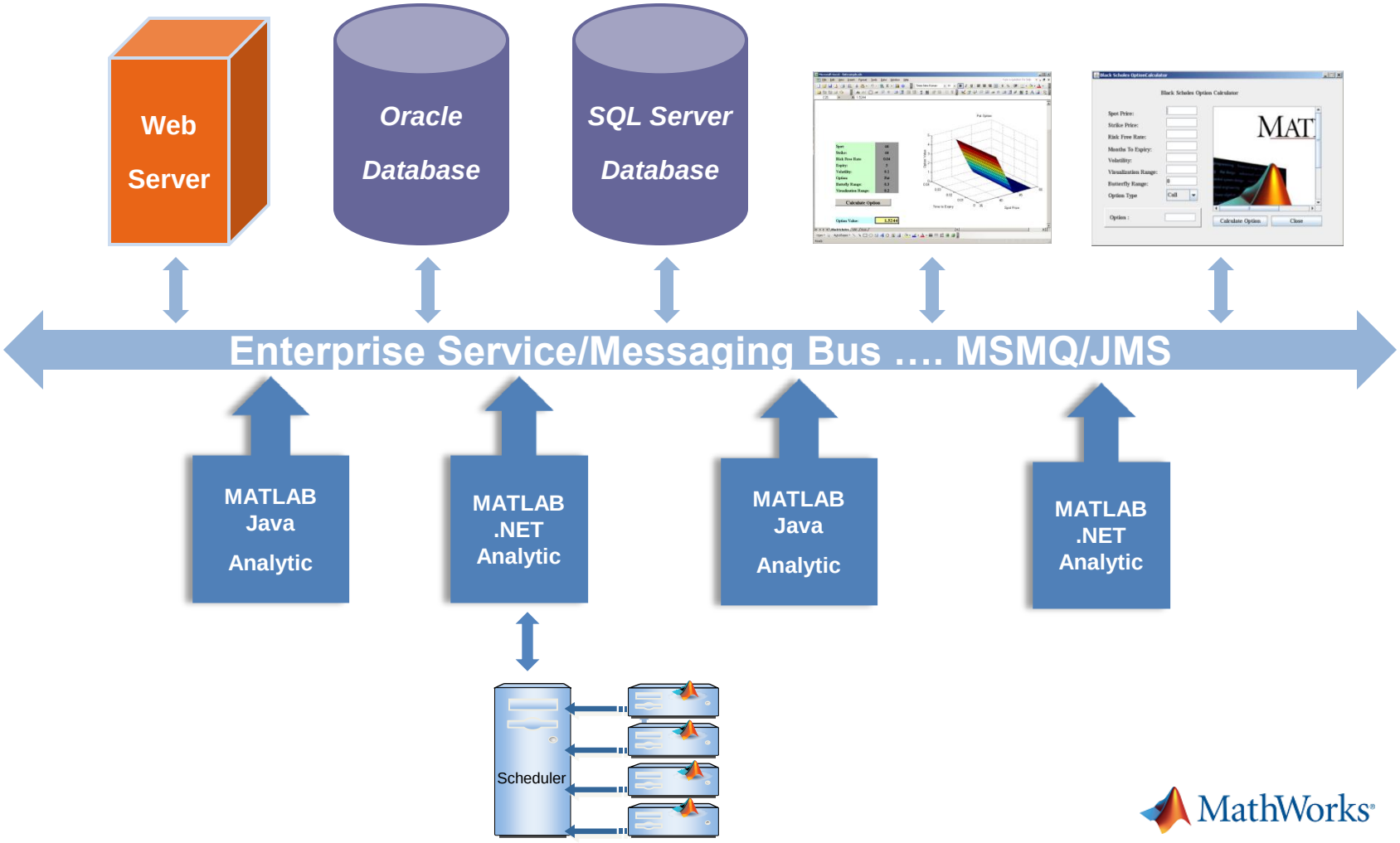
Deployment products for targeted environment



Sharing your ideas across the organization



Bringing the algorithm to the data



System integration through messaging services