Unifying Front Office and Risk Analytics

qCon London 2011

Friday, 11 March, 2011

Kirk Wylie - CEO, CTO

Jim Moores - Head of Platform Development

Financial Analytics

Computational Analytics

Mathematical calculations key to computational finance

- Curves (Yield, Credit, etc.)
- Greeks/Sensitivities
- VaR/cVaR
- Portfolio Performance

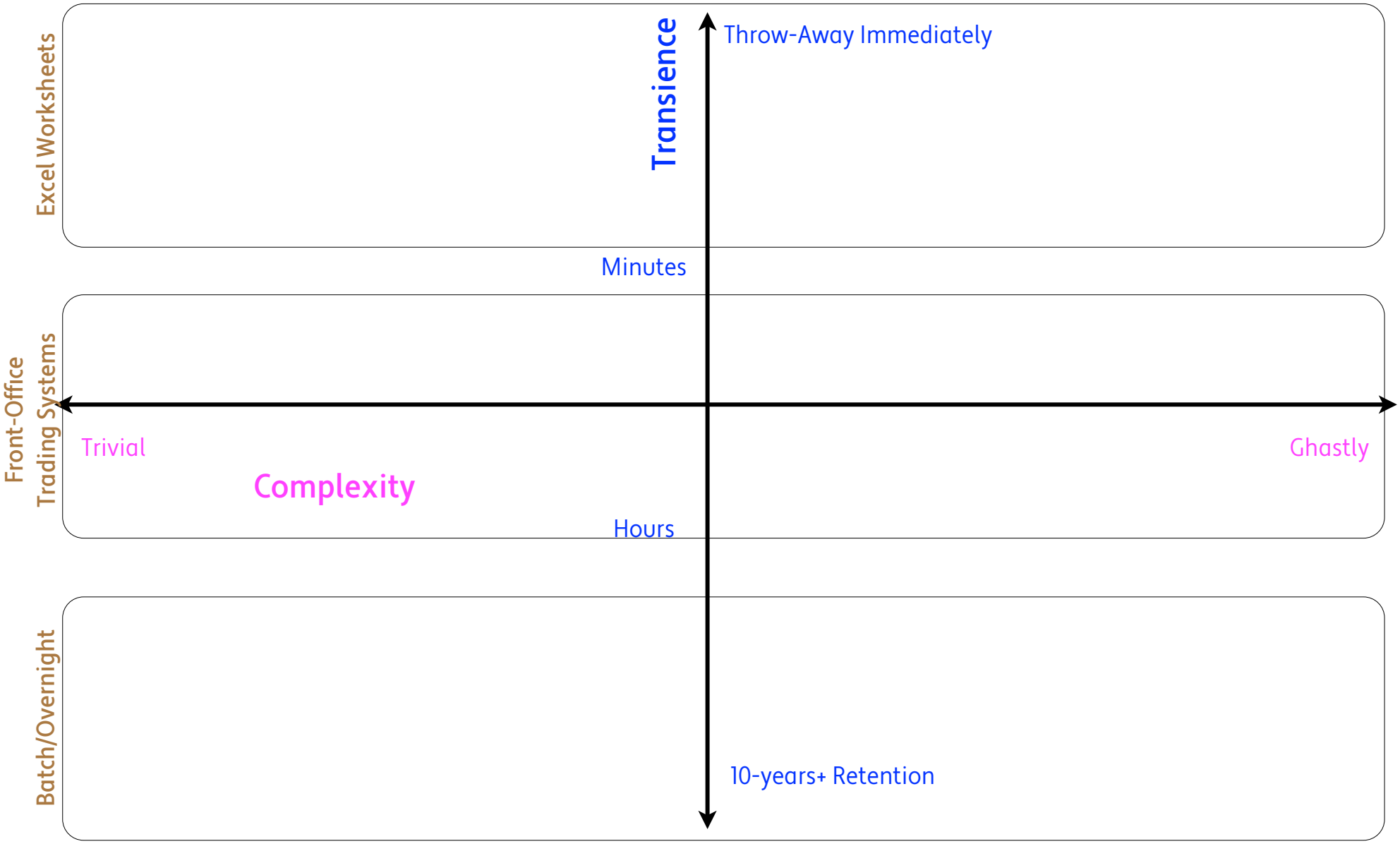
Focus of Talk

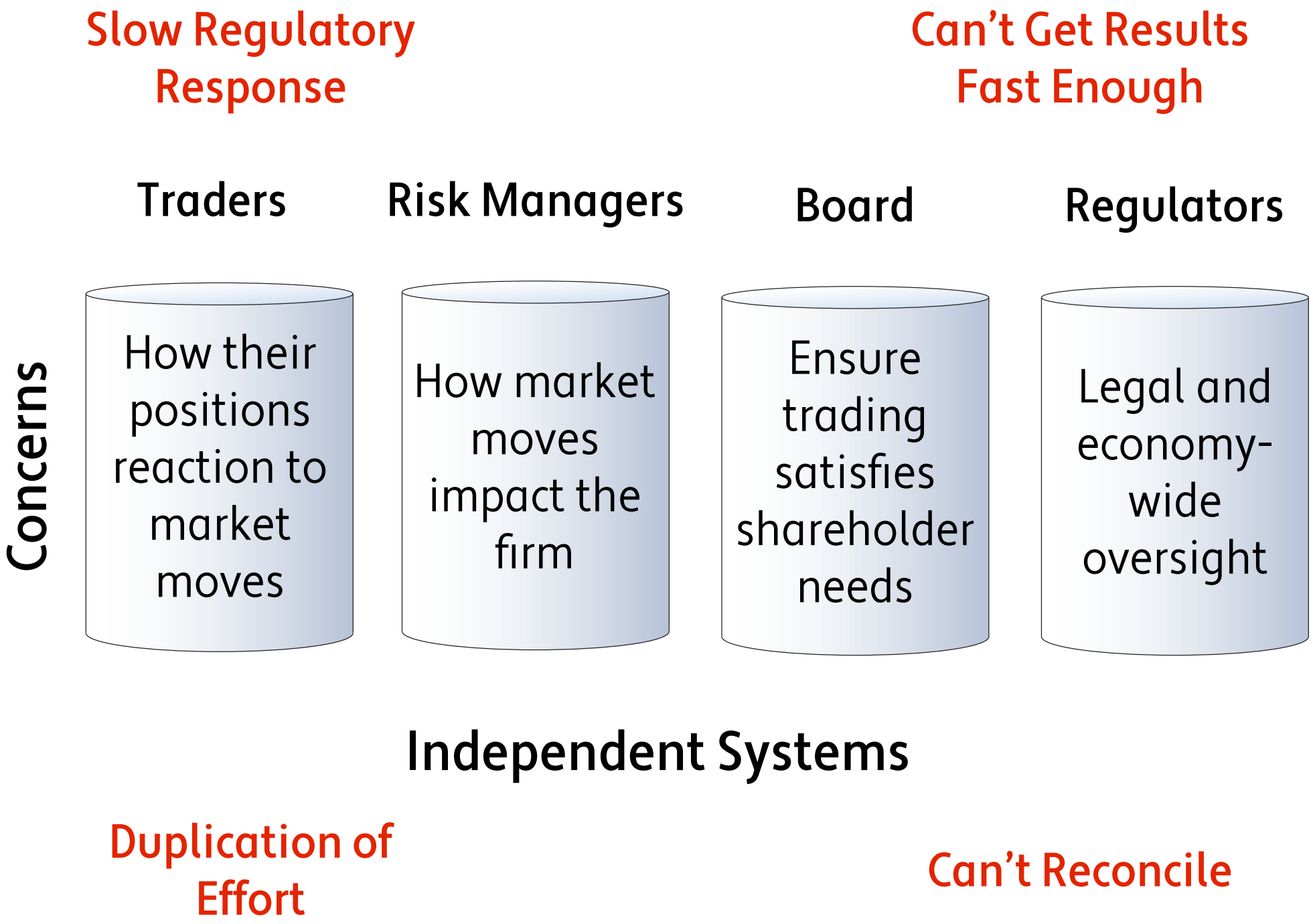
Data Analytics

Analysis of large-scale data sets

- Tick-stream analysis
- Fraud detection
- Customer profiling
- Analysis of Computational Analytics

Calculation Lifecycle




Slow Regulatory Response
Can't Get Results Fast Enough
Traders
Risk Managers
Board
Regulators
Concerns
How their positions reaction to market moves
How market moves impact the firm
Ensure trading satisfies shareholder needs
Legal and economy-wide oversight
Independent Systems
Duplication of Effort
Can't Reconcile

What Do We Need?

- **A software stack**
- **Capable of handling all calculation requirements**
  - From ad-hoc/one-off to the largest overnight batches
  - From trivial calculations to large-scale simulations
  - Speed for traders, sophistication for risk managers
- **With a modern distributed architecture**
  - All calculations server side
  - Support for user tools that end users want to use
- **Designed for integration**
  - Work with what's already on the ground
  - Use as the basis for complex applications

Why Don’t We Have It?

- **Existing Vendors Won’t Build It**
  - Their programmers aren’t good enough
  - Their whole business model is based around lock-in
- **Time/Resource Constraints**
  - Requirements go far beyond one single project’s needs
- **Internal Costing Silos**
  - Multiple desks **and** risk jointly paying for this type of development? ROFL
- **We Need Source Code**
  - Complex integration requires it
  - SLAs don’t help you when the system goes down for real
  - How do you know what’s under the hood?

OpenGamma Platform

- **Single solution for all analytics applications**
  - Live trading applications
  - Ad-Hoc Pre-Trade analytics
  - Near-Real-Time risk management
  - Batch/Overnight risk management
- **Designed for integration**
  - Leverage proprietary in-house systems
  - Integrate with existing vendor solutions
  - White-box as part of a comprehensive offering
- **Comprehensive Solution**
  - Everything for a comprehensive risk and analytics platform for trading
  - Scales from individual trader to enterprise-wide use
- **Asset-Class Neutral**

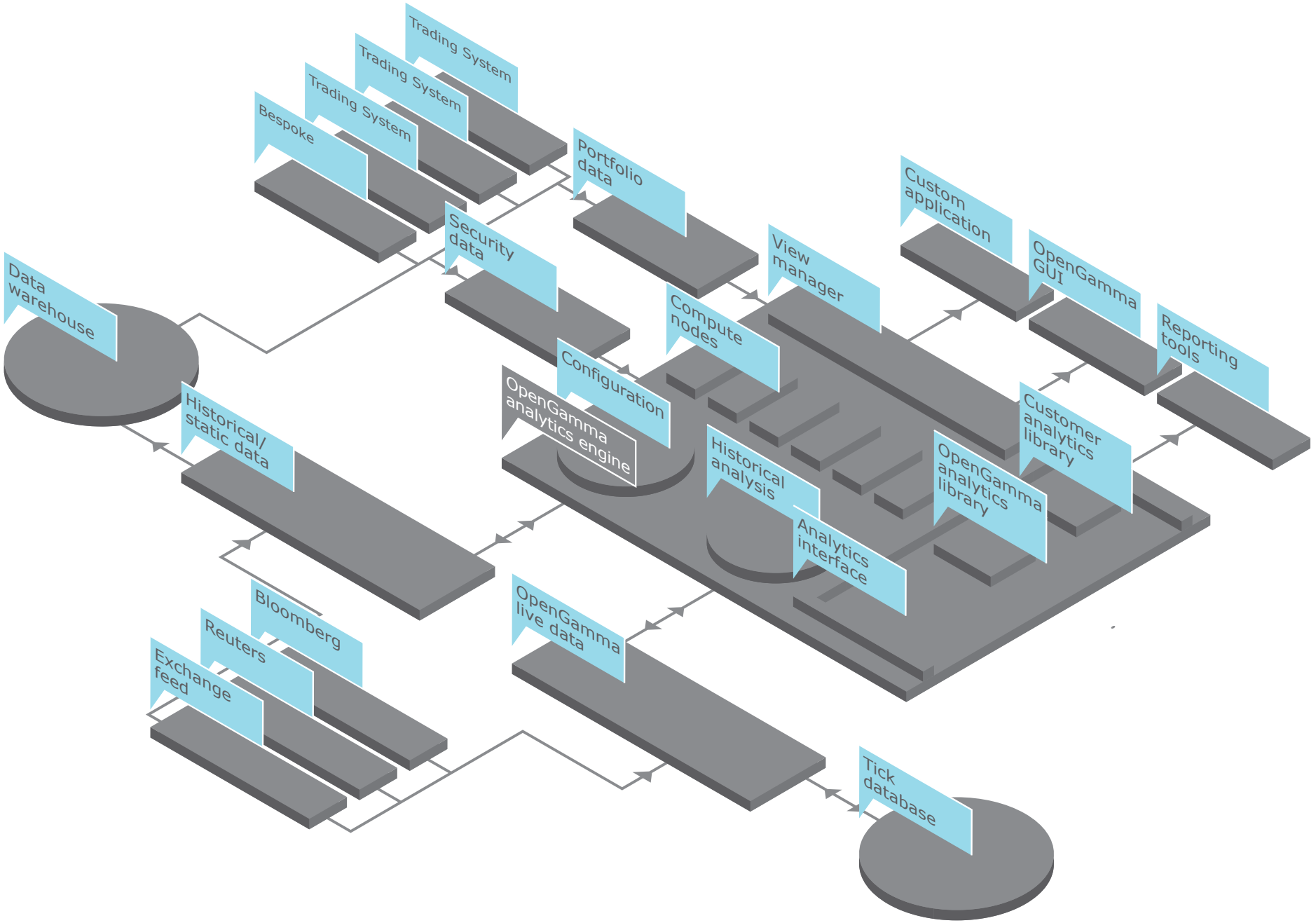
Just How Open?

- **Core platform released under APLv2**
  - Public repository on GitHub
  - Full releases downloadable with/without libraries and source
- **JIRA open to the world**
- **Documentation open to the world**
- **Expected first Open Source release April/May 2011**

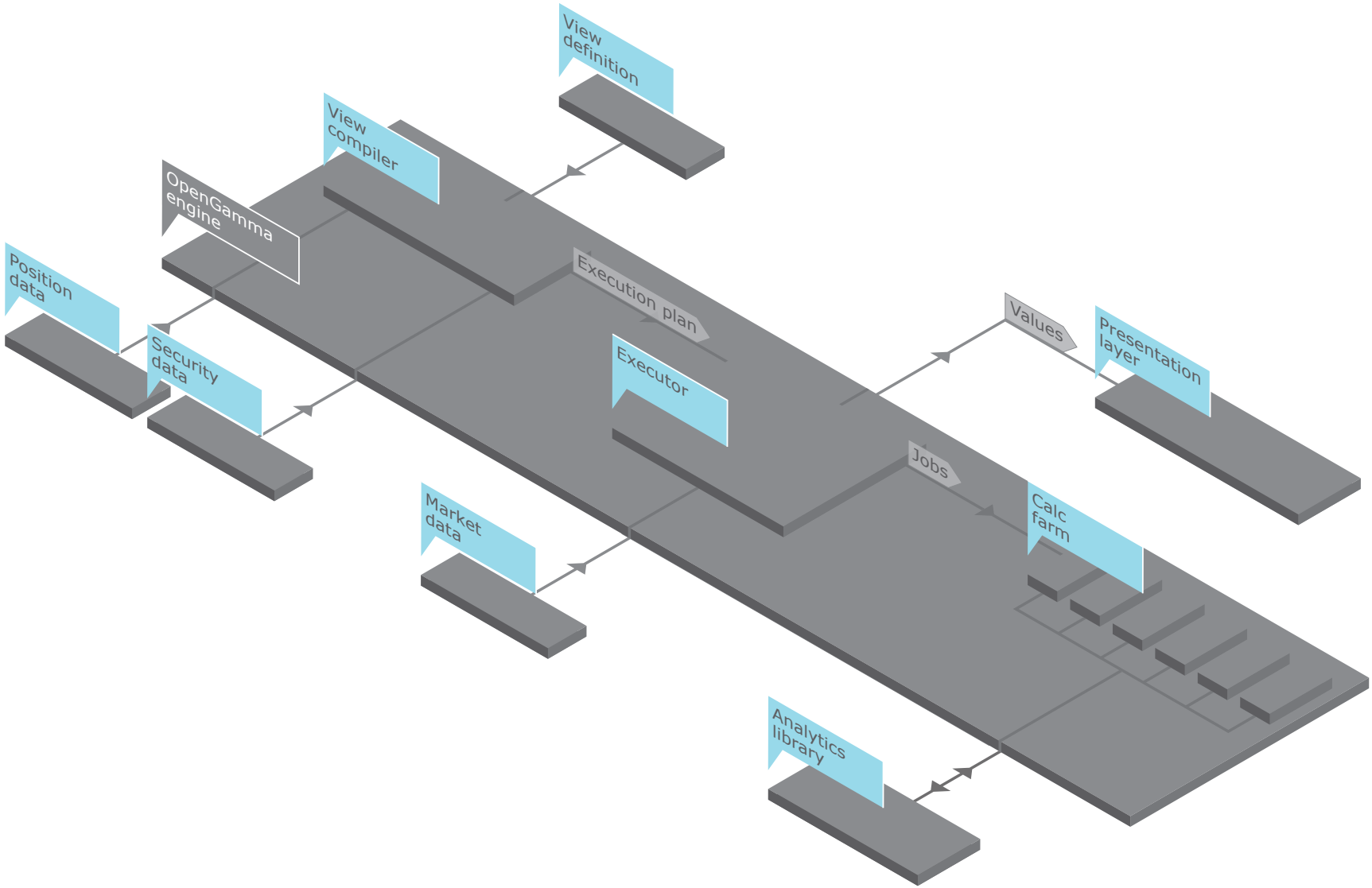
“How Will You Make Money?”

- **Support Contract**
  - Make your procurement department happy
  - Make your corporate info-sec department happy
- **Commercial Components**
  - Integrations with proprietary systems with trade secret APIs
  - Currently supported: Bloomberg (SAPI/Terminal), Reuters (RMDS), ACTIV, Excel
  - You’ll still get the source code for these
- **Consulting Services**
  - Access to the original authors
  - Proactive management and support
- **Pre-Packaged Versions**
  - Incredibly tight integration with other vendor systems

OpenGamma Platform



View Processing Engine



Key Features

- **Unified Analytics Calculation Infrastructure**
  - Ad-Hoc, Near-Real-Time/Streaming and Batch in one architecture
  - Re-use all integration with proprietary modules across all projects
- **Radically Open Architecture**
  - Every component can be replaced at customer site
  - Every component can be used independently
  - Built with the needs of Tier-1 Institutions in mind
- **Modern, Distributed Architecture**
  - RESTful endpoints to all services
  - MOM-based data distribution possible for all connections
  - Web-Scale techniques used throughout system
  - Components configurable through Dependency Injection
- **Source Code For All Modules At Your Fingertips**

Key Platform Components

- **OpenGamma-Live Data**
  - Market Data Management solution
- **OpenGamma Calculation Engine**
  - Dependency Graph approach to calculations
  - Whole system operates in metadata
- **Rich Data Management**
  - Time-Variant Fact Data
  - Data Composition
- **Client Management Facilities**

OpenGamma-Live Data

- **Market Data Abstraction**
  - Write applications (or plug into OpenGamma) and have a consistent view no matter what underlying data source
- **Market Data Aggregation**
  - Combine Reuters, Bloomberg, IDC, ACTIV, quote-based, bespoke feeds in one consistent infrastructure
- **Market Data Transformation**
  - Field name/identifier normalization (e.g. bid vs. BID_PRICE, RIC vs. BUID)
  - Value transformation (price/rate, pounds/pence)
- **Shared Services**
  - Last Known Value Caching
  - Entitlement Checking & Integration
  - Tick storage/replay

Declarative Dependency Graph

- **End-Users Specify Desired Results**
  - "Fair Value", "Delta", "Yield Curve Sensitivities", "hVaR"
  - Scenarios to modify results: flat-at-market, fixed/% bumps, curve shift
- **OpenGamma Builds Dependency Graph**
  - Each sub-calculation is a node in the graph
  - Share interim calculations between nodes
- **Dependency Graph Used For Execution**
  - Automatic job parallelism and distribution
  - Minimal recalculation on streaming results
- **Dependency Graph Allows "Explain Value" Functionality**
- **Same system for ad-hoc, live greeks/risk, and batch risk**

Metadata Basis

- **OpenGamma Engine doesn't interpret security or analytic definitions**
  - Can support new analytic measures and securities without vendor support
  - Can add support for new security types and analytic models at runtime
- **Analytic functions have control over inputs/outputs**
  - Can operate on new data types and structures without platform support
- **OpenGamma analytics are implemented as a plugin**
  - High level of confidence any customer's analytics library can be integrated

```
public interface Security extends UniqueIdentifiable {

  UniqueIdentifier getUniqueId();

  String getName();

  IdentifierBundle getIdentifiers();

  String getSecurityType();
}
```

```
public class ComputedValue implements Serializable {

  private final ValueSpecification _specification;

  private final Object _value;

  public ValueSpecification getSpecification() {
    return _specification;
  }

  public Object getValue() {
    return _value;
  }
  // SNIP -- Constructors, .equals(), .hashCode(), etc.
}
```

Time-Variant Fact Data

- **Applies only to data in OpenGamma's database schemas**
  - Fact-based data
  - Security Definitions, Positions, Portfolios, Time Series Points
- **Store all data on two time dimensions:**
  - Effective Timestamp: "At what point does this data apply"
  - Correction Timestamp: "At what point did I observe/change that value"
- **Designed for batch risk restatement**
  - Able to reproduce any metric as of any time in the past
- **Example**
  - Monday book a $100MM swap trade
  - Tuesday correct to €100MM
  - Wednesday correct to €200MM

Data Composition

- All data able to come from multiple sources
  - Single namespace and identifier resolution rules
  - RDBMS, NoSQL, Files, In-Memory
  - Example: Trading system API, OpenGamma RDBMS, and In-Memory all at once
  - Predictable, easy to implement new source

```
public interface SecuritySource {
  Security getSecurity(UniqueIdentifier uid);

  Collection<Security> getSecurities(IdentifierBundle bundle);

  Security getSecurity(IdentifierBundle bundle);
}
```

```
public interface PositionSource {

  Portfolio getPortfolio(UniqueIdentifier uid);

  PortfolioNode getPortfolioNode(UniqueIdentifier uid);

  Position getPosition(UniqueIdentifier uid);

  Trade getTrade(UniqueIdentifier uid);
}
```

```
public interface RegionSource {

  Region getRegion(UniqueIdentifier uid);

  Region getHighestLevelRegion(Identifier regionId);

  Region getHighestLevelRegion(IdentifierBundle regionIdentifiers);
}
```

```
public interface ExchangeSource {
  Exchange getExchange(UniqueIdentifier uid);

  Exchange getSingleExchange(Identifier identifier);

  Exchange getSingleExchange(IdentifierBundle identifierBundle);

}
```

Client Management Facilities

- **View calculation proceeds based on rules**
  - One-off as results requested (for ad-hoc calculations)
  - As fast as data available (near-real-time streaming)
  - At-most-as-fast, at-least-as-fast
  - On set schedules
- **Clients separated from actual calculations**
  - Each actual/remote client separate from each other
  - Allows advanced functionality like client pause/restart, separate delivery schedules, different resolution of results
- **Results can be delivered to automatic systems**
  - View Processor chaining
  - MOM-based broadcast

Technology Specifics

- **Distributed architecture**
  - Fudge for meta-object definitions, JSON, XML
  - RESTful HTTP, JMS, AMQP, pure sockets
- **Pure Open-Source Reference Platform**
  - Java 6, PostgreSQL, LucidDB, ActiveMQ, RabbitMQ, HornetQ, MongoDB
- **Designed for infrastructure portability**
  - Vertica, Oracle, Sybase, SonicMQ, Solace, Tervela
- **Comprehensive UI possibilities**
  - HTML5/CSS3 Web GUI, tight Excel integration
  - R integration Q2/Q3 2011
  - Java/C# client libraries for custom GUI work
- **Deployment Options**
  - 100% on-site, 100% off-site, split architecture all possible

Demonstration