June 2007

Building Distributed Applications

Integrating the Trading Value Chain via .NET Framework

Applies to: Financial Services Architecture .NET Framework

Steven A. Smith
4th Story, LLC.

John T. Powers
Digipede Technologies, LLC.

Brian Sentance
Xenomorph Software Ltd.

Stevan D. Vidich
Microsoft Corporation

Summary

This paper describes how .NET Framework 2.0 was used to integrate a strategy execution management system with grid-enabled trade order analytics and financial markets data management platform. Integration details and functional overviews of respective solutions are provided.

Contents

Introduction

Many customers in the Securities and Capital Markets industry rely on products from Independent Software Vendors (ISVs) to obtain solutions for their business needs. These products usually come from different ISVs and they are based on different technologies. Very often, customers are confronted with significant post-purchase deployment and integration costs if they need various ISV products to communicate and exchange data with each other.

This paper describes how Microsoft partners 4th Story, Digipede Technologies and Xenomorph Software paired and pre-integrated their product offerings to eliminate post-purchase integration costs for customers who use Microsoft's platform in their business. Seamless integration that includes a Strategy Execution Management System with grid-enabled trading analytics and enterprise-grade data management was realized using .NET Framework 2.0. Details of this integration effort are described herein, together with functional overviews of relevant partner products.

Integration Objectives

The following objectives were established at the beginning of the project:

- Minimize the time and effort required for customers in Securities and Capital Markets to identify profitable new trading strategies.
- Support development in an open-ended, state-of-the-art component-based software development environment, with the most advanced tools available. Support component re-usability.
- Achieve ease of programming equal to or greater than that available with common desktop applications.
- Provide access to common desktop-based tools and applications.
- Provide for pluggable value-added service providers, especially for market and fundamental data.
- Support open-ended configurability. Provide adapter interfaces to supporting external services such as a security master, market data access, order management and position management.
- Support multi-channel client access.
- Provide convenient component and application scalability to very high volume trading and highly complex trading rules.

4S Design Overview

Trading strategies are developed on the 4th Story product platform by using and extending existing components. Once developed, a strategy is hosted by being plugged at runtime into a service-based event container. The container provides a managed runtime environment with access to all the necessary services and events to achieve its objectives. The container provides an event management structure that synchronizes and drives the logic deployed with strategy. Strategies and their constituent parts respond to events managed by the environment.

The container's service adapters are pluggable, which allows for different implementations of the same behavior. A service adapter may include a service provider or an interface to one. The key point here is that there is a separation of service contract, or interface specification, from the implementation of the contract. Consumers of the service understand the service contract but not the implementation.

A graphical overview of business functionality is provided in Figure 1.

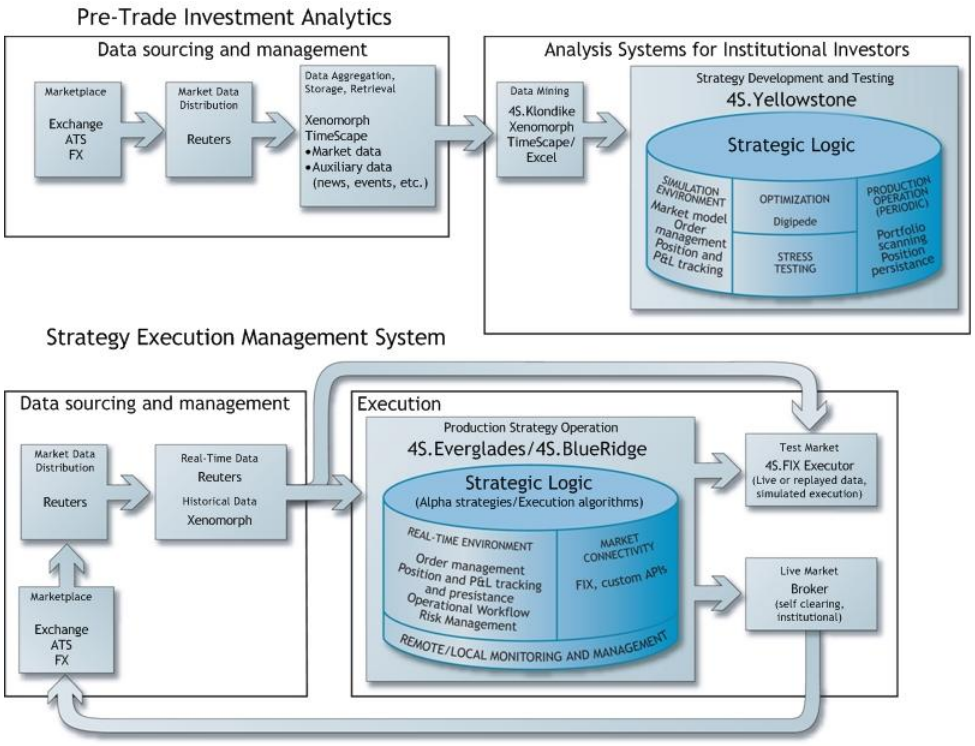

Figure 1. 4S Integrated Business Functions

Xenomorph TimeScape Design Overview

TimeScape is Xenomorph's real-time analytics and data management suite for financial markets. TimeScape combines data capture, integration, cleansing, storage, viewing and analysis, based upon high performance database technology and a highly flexible cross-asset data model. Unlike many data management and database systems, it is uniquely suited to financial markets because it understands concepts such as baskets, volatility, correlation, filling missing data (so long as the data used is not too stale) and proxying of instrument data when market data is unavailable.

TimeScape's design focuses on empowering business users (trading, portfolio and risk managers) to do more with the data they need to manage and analyze, enabling faster time to market with new trading ideas and strategies. The system also has both business- and IT-friendly interfaces, from Excel to .NET, plus server-side software development kits (SDKs) that a client can use to quickly and easily integrate proprietary statistical analytics, pricing models and external data sources.

Figure 2 describes TimeScape's technical architecture.

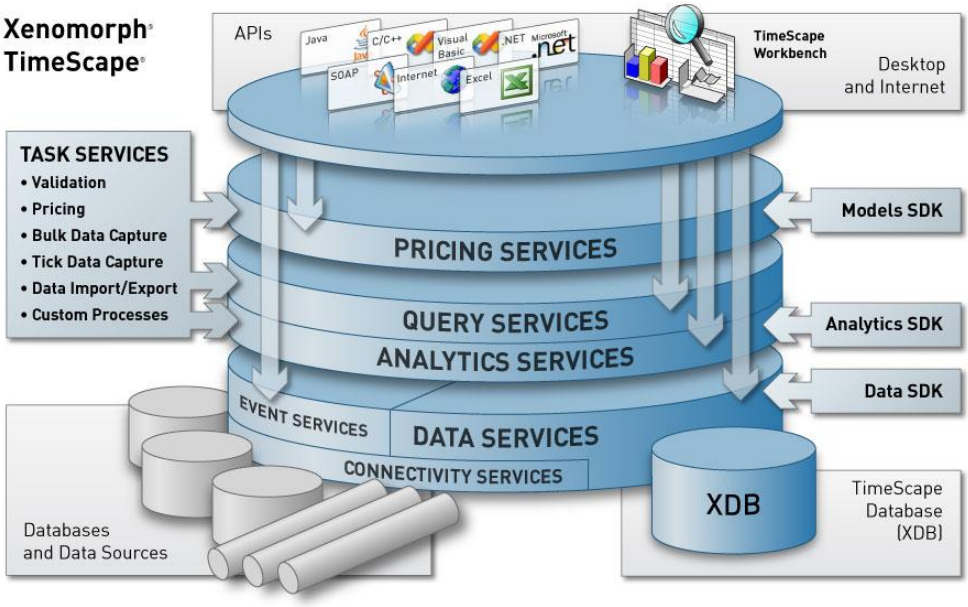

Figure 2. Xenomorph TimeScape technical architecture

Digipede Network Overview

The Digipede Network is distributed computing software that provides increased performance and scalability for Windows applications. Built entirely using the .NET Framework, the Digipede Network is integrated with many Microsoft technologies. This integration helps to make the Digipede Network radically easier to deploy and use than other grid computing solutions, making it a natural choice for scaling out complex algorithms in the 4th Story product platform.

The Digipede Network is composed of two main components: the Digipede Agent, which runs on each compute node and the Digipede Server (see Figure 3). The Server publishes a list of jobs and tasks, and the Agents select jobs to work on based on their own capabilities and the priorities of the jobs. A *task* is a single instance of a program or program component that runs once on a single compute node; a *job* is a collection of tasks. The Digipede Server monitors all work performed on the grid, and provides administrative and management functionality. Digipede Agents can run on any 32-bit or 64-bit desktop or server running Windows 2000 or later.

Figure 3 shows a configuration running on a set of dedicated cluster nodes (running the new 64-bit Windows Server 2003 Compute Cluster Edition), shared departmental servers (running 32-bit Windows Server 2003) and shared desktops (running Windows Vista).

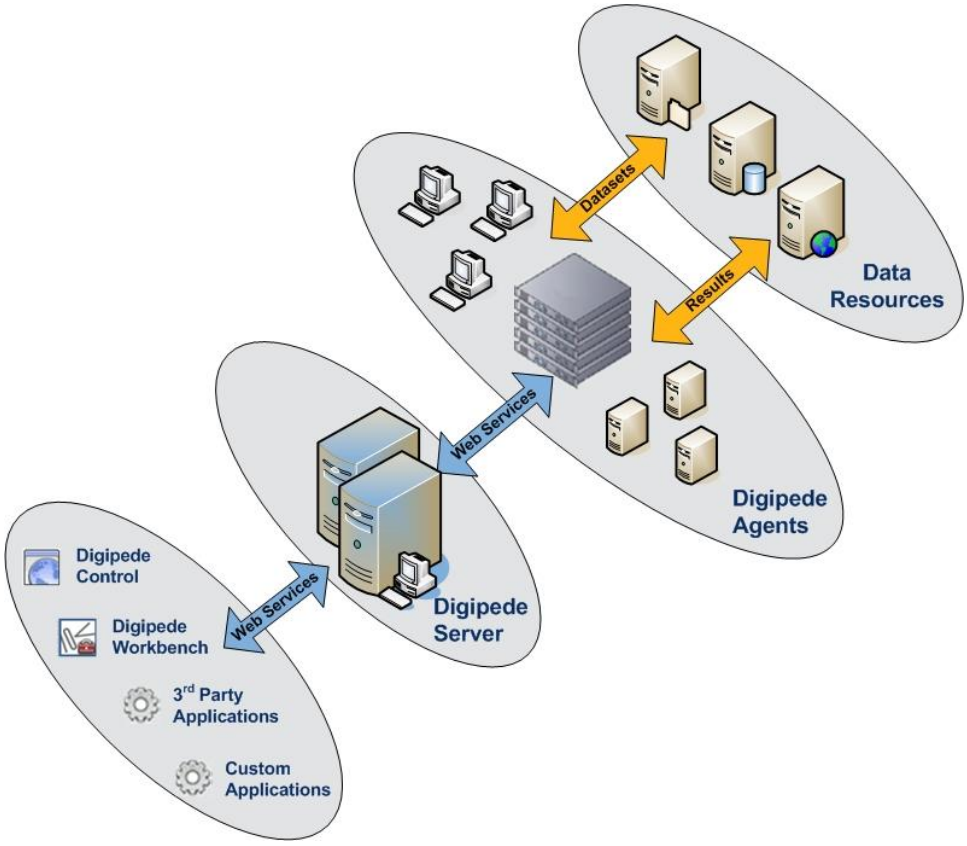

Figure 3. The Digipede Network

The Digipede Network provides special benefits for .NET applications, allowing convenient distribution and execution of .NET objects. The developer simply designates a class for distributed execution; the Digipede Network handles the process of serializing the corresponding objects, streaming these objects to available compute nodes, executing the objects on those nodes and bringing the results back together in the developer's master application. This unique capability allows .NET developers to work in the familiar Visual Studio environment without learning complex new programming patterns to develop high-performance distributed applications.

Integrating Market Data

Xemomorph offers a very flexible data structure, which is a distinct advantage in the data storage and manipulation. 4th Story strategies expect a specific data construct, so the 4th Story/Xenomorph market data service adapter is implemented to perform this mapping. Strategy developers use the adapter but do not have to be concerned with the implementation.

Note that 4th Story also has a concept of non-standard or 'Auxiliary' data (market, news, events, etc.) and a separate auxiliary data service adapter could easily be developed to interface with Xenomorph for this type of data. Due to Xenomorph's .NET API, developing a new service adapter for Auxiliary data is a very straightforward process.

The market data history manager interface specifies that an implementation must be able to supply a time series given a security identifier and a time span designator. This specification effectively

describes an adapter interface for which there are many possible implementations or specific adapters. Market data vendors have differing programming interfaces (APIs), differing connectivity and differing file organizations. An implementation of the 4th Story market data history manager interface for a specific vendor is required for a strategy to acquire data from the vendor. Adapters for different vendors plug into the container if they fulfill the interface specification. A trading strategy can know - but does not need to know - what implementation it is using.

Xenomorph provides a number of .NET libraries that can be used to retrieve data from Xenomorph's TimeScape database into 4th Story. Moreover, specialized queries/functions loaded on the Xenomorph TimeScape server can be accessed by 4th Story's adapter via the Xenomorph query functionality (see Figure 4).

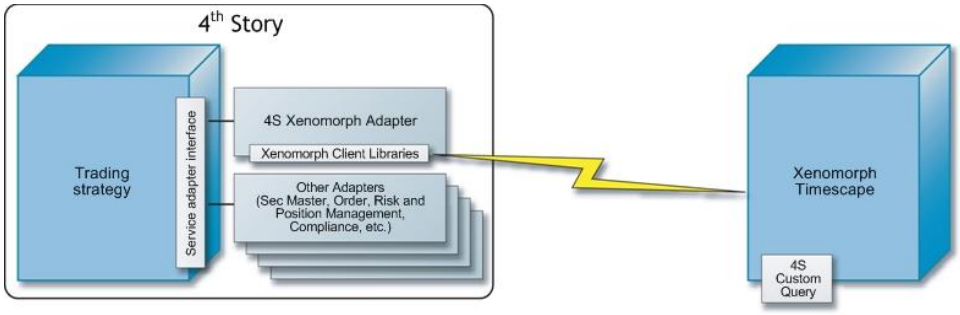

Figure 4. 4S – Xenomorph Integration

TimeScape has two primary data formats:

- Day Time Unit is for data whose date/time resolution is one day or greater. Typically, this is periodic time series data.
- Tick Time Unit is for data whose date/time resolution is less than one day, e.g., hours, minutes, milliseconds, etc. Data stored with the Tick Time Unit can be periodic (i.e., uniform time spacing between each date point) or tick data, with uneven time spacing between the data as one would see, for example, in a history of Last Trades.

4th Story uses both periodic and tick data. In addition, the periodicity of the time series data used by 4th Story is selectable depending on what the data store can provide.

4th Story's Xenomorph adapter has three ways of extracting data from TimeScape:

1. For daily and weekly data, it calls a Xenomorph function which returns day time unit data (optionally rolled into weekly periods) for a specified date range or number of periods.
2. For raw tick data, it calls a Xenomorph function that returns the tick data (for a specified date range).
3. For intraday periodic (bar) data, it calls a custom 4th Story query loaded into the Xenomorph server that rolls the tick data up into periods of a user-specified duration (5 minute, 30 minute, etc.). This processing is done on the server and only the rolled up data is returned to 4th Story, speeding processing and minimizing network traffic.

Configuration settings in 4th Story allow the mapping of field names between the two systems. In addition, Xenomorph-specific symbols can be assigned to securities within 4th Story via the latter's security master.

Integrating Trading Strategies with Grid Computing

For customers in Securities and Capital Markets, finding profitable new trading strategies more quickly than competitors can provide a real business advantage. The primary focus of 4th Story's 4S.Yellowstone product is the identification of automated alpha-generating trading strategies. 4S.Yellowstone is designed to allow trading strategists to quickly explore, optimize and stress test new strategy ideas, and then operate the high potential ideas in production (with real time strategies running in 4S.Everglades).

Strategies are composed of indicators, which are computational units used and reused by the strategies, and business logic tying them together. The indicators are used in making decisions about when to enter a position (entry model) and when to exit one (exit model). When combined, the indicators, entry models, exit models and the business logic that ties them all together constitute a strategy. All of these components can also include parameterized values and they usually do.

Strategies are optimized either via an exhaustive search algorithm, which runs through all possible scenario combinations within a range to come up with the 'best' ones, or via a genetic algorithm that searches based on specific fitness tests.

Search and optimization can be very compute intensive. To expedite these steps, 4th Story turned to grid computing. The search and optimization utility for genetic algorithms can spread its work out over a grid managed by the Digipede Network. The utility works with the Digipede API to assign work to the nodes of the grid and then collects the results centrally. This approach works well and the scaling is fairly linear.

The optimization running in 4S.Yellowstone calls Digipede client libraries to create *jobs* for the Digipede network in the form of .NET objects that are serialized over to the Server. The Server manages the distribution of this work out to the grid nodes which are running the Digipede Agent software. These nodes are also pre-loaded with 4th Story software, including data adapters (Xenomorph).

Upon receipt of some work to be performed from the Server, the Digipede Agent instantiates 4th Story on the node and passes it the serialized object with the job information. The 4th Story software on the node then performs the job, calling external data (i.e., from Xenomorph) as required. The Agent passes the results back to the Digipede Server, which then passes them back to the 4th Story optimization (Figure 5).

The results are that, with the help of the grid, genetic optimization techniques can be used reasonably quickly and the throughput of optimization analysis can be increased simply by adding more compute nodes to the grid. With the computing power of a grid behind 4S Yellowstone, users can screen and stress-test a wide variety of potential strategies quickly. Users who identify profitable new trading strategies more quickly than their competitors can gain a trading advantage.

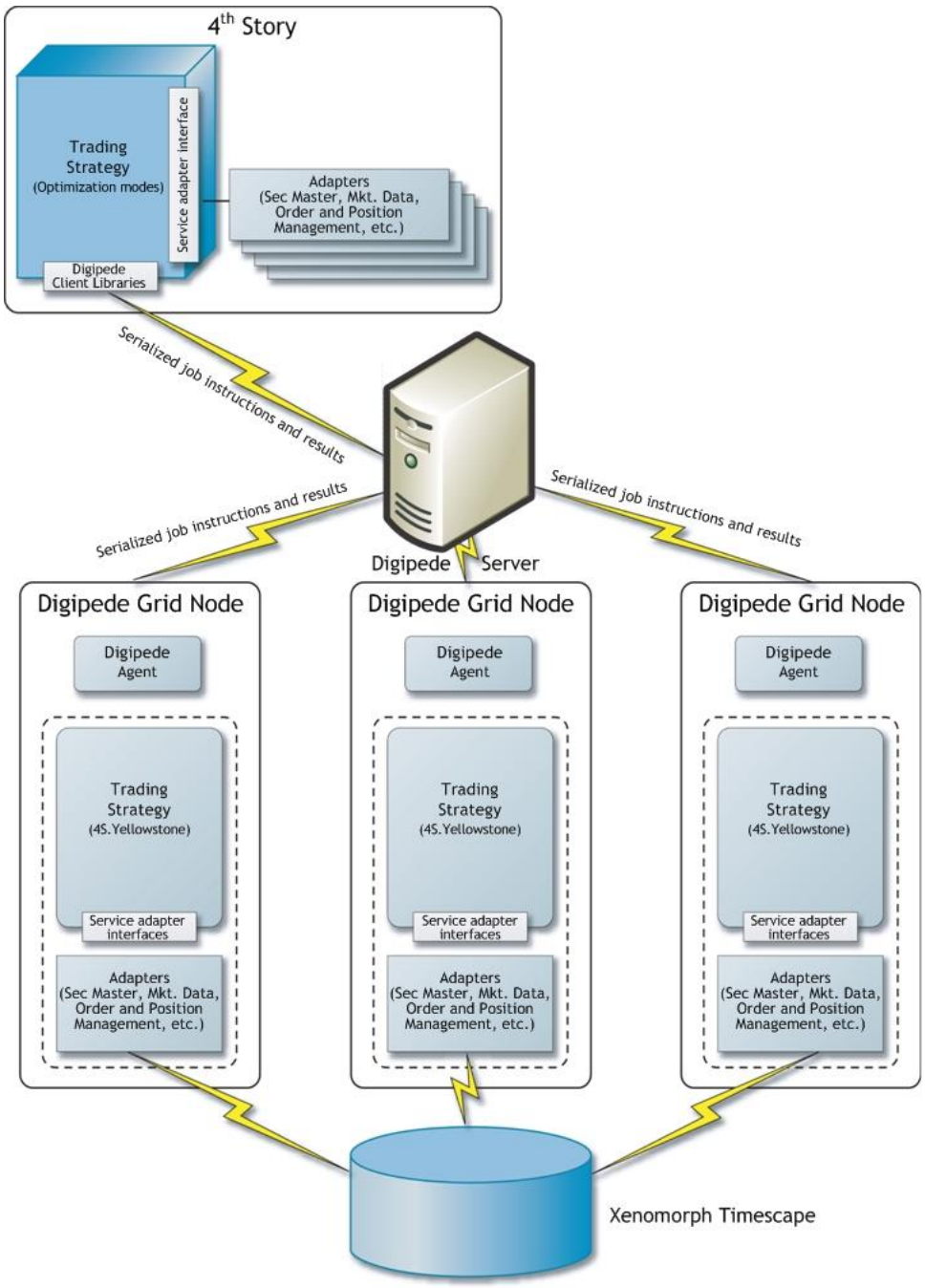

Figure 5. 4S – Digipede Integration

Conclusion

Reducing overall technology costs while increasing agility is a key deciding factor when organizations are looking to add new functionality to their application portfolio. In an effort to reduce integration costs and improve interoperability, enterprises expect vendors to adopt open standards and build products that can be easily integrated with each other. Microsoft partners 4th Story, Digipede Technologies and Xenomorph Software worked together to integrate their products using the underlying .NET Framework. In this manner, customers in the Securities and Capital Markets industry that rely on the .NET/Windows platform for their business can expect to save considerable time and resources when implementing some of the key Trading Value Chain components. The effort described herein has led to seamless integration of a strategy execution management system with grid-enabled trade order analytics and financial markets data management platform.

References

www.4thstory.com – Provides information about 4th Story, its products and available services.

www.digipede.net – Provides information about Digipede Technologies, its products and available services.

www.xenomorph.com – Provides information about Xenomorph Software, its products and available services.

msdn2.microsoft.com/en-us/architecture/aa699365.aspx - MSDN Financial Services Industry Center