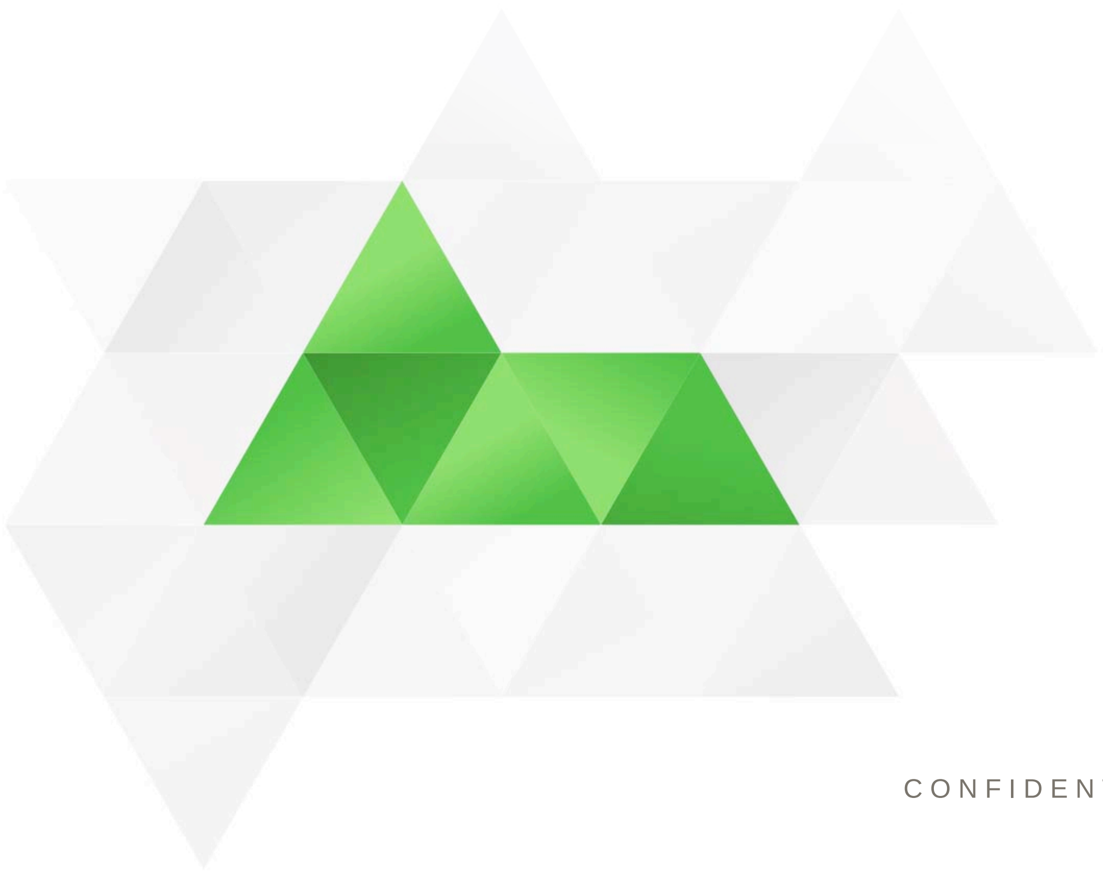
CAPLIN

Caplin Platform 6.0

Overview

November 2012



Contents

1 Preface

1.1 What this document contains

This document gives a technical overview of the Caplin Platform.

It explains:

- What the Caplin Platform is and what you can use it for.
- The Platform's features
- The Platform's architecture
- The key aspects of using the Platform to develop web trading platforms

About Caplin document formats

This document is supplied in Portable document format (*.PDF* file), which you can read on-line using a suitable PDF reader such as Adobe Reader®. The document is formatted as a printable manual; you can print it from the PDF reader.

1.2 Who should read this document

This document is intended for:

- Technical Managers
- Enterprise Architects and System Architects
- Software Developers

1.3 Related documents

- **Caplin Trader: http://www.caplin.com/caplin-trader**

  An overview of Caplin Trader, the complete development suite for creating HTML5 apps for trading. It includes a link to the detailed documentation on the Caplin Trader 3 developer's site.
- **Caplin Platform: Deployment Framework Overview**

  Gives an overview of the Caplin Platform Deployment Framework,
  and explains the concept of Caplin Platform blades.
- **Caplin Trading: Integrating The Caplin Platform With A Trading System**

  Describes how a Caplin Trading Adapter allows you to integrate the Caplin Platform with your existing trading system. It explains trading concepts (such as Trade Models), how to implement a Trading Adapter, and how to configure Caplin Liberator for trading.

- **Caplin Permissioning: Permissioning Overview And Concepts**

  Introduces permissioning concepts and terms, and shows the permissioning components of the the Caplin Platform architecture.

- **Caplin Integration Suite for Java API Documentation**

  The reference documentation for the Caplin Integration Suite Java APIs.

- **Caplin StreamLink Overview**

  Gives a technical overview of Caplin StreamLink.

1.4 Feedback

Customer feedback can only improve the quality of our product documentation, and we would welcome any comments, criticisms or suggestions you may have regarding this document.

Please email your feedback to documentation@caplin.com.

1.5 Acknowledgments

2 Introduction

Caplin provides technology to help you build a web trading platform. You can use Caplin's applications, tools, and APIs to build and run web trading apps and installed applications that integrate with your existing systems, such as pricing and trading across multiple asset classes.

Caplin's technology includes:

- **Caplin Platform**

  An integrated suite of software that supports the services and distribution capabilities needed for web trading.

- **Caplin Integration Suite**

  A set of APIs and tools for creating adapters that integrate the Caplin Platform with external systems.

- **StreamLink**

  An API that allows a client application to communicate with the Caplin Platform.

This document describes the Caplin Platform and how you can use it, along with other Caplin products and tools, to create a web trading platform.

Caplin also offers **Caplin Trader**, a complete development suite for creating HTML5 apps for trading. Caplin Trader uses StreamLink to connect to the Caplin Platform. For more information about Caplin Trader, see the Caplin Systems web site at http://www.caplin.com/caplin-trader.

2.1 What can I use the Caplin Platform for?

You can use the Caplin Platform in conjunction with other Caplin APIs and tools to build and run a web trading platform. The Platform provides everything you need to connect a custom client application, such as a web trading app, to your existing systems, such as pricing, trading, and permissioning.

2.2 What are web trading platforms and web trading apps?

A **web trading platform** is a layer of software that provides users with real-time access to trading services via the web. A good web trading platform should include all the software needed to support this, including:

- connectivity to information, pricing, and trading systems,
- normalization of data,
- permissioning,
- data management and enhancement,
- latency control,
- end-user display.

Web trading platforms are applicable to any form of trading in which real-time information and action is essential. Common examples are single dealer platforms (SDPs) operated by banks for institutional clients, and mass-market online trading offerings from retail brokers.

A typical web trading platform consists of a number of layers, as shown in the following diagram:

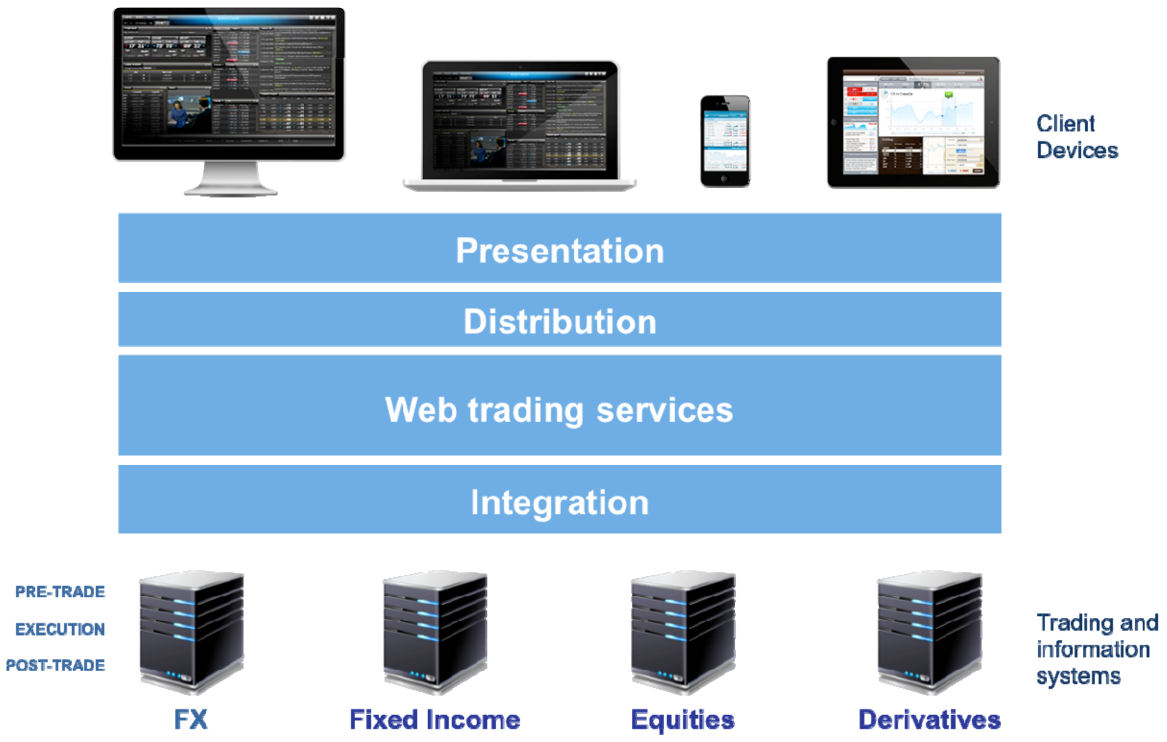


Figure 2–1 Web trading stack

- The *Presentation* layer displays information to the end-user and handles the user interactions with the Platform.
- The *Distribution* layer manages the transmission of real-time data to and from the client application.
- The *Web trading services* layer implements the core and application-specific trading functions needed by the client application. This is a key part of a web trading platform as it is important to limit the bandwidth usage and client side processing.
- The *Integration* layer brings together all the external trading and information systems, managing the connections to them and normalizing data as required.

The following diagram shows how the Caplin products fit into this stack.

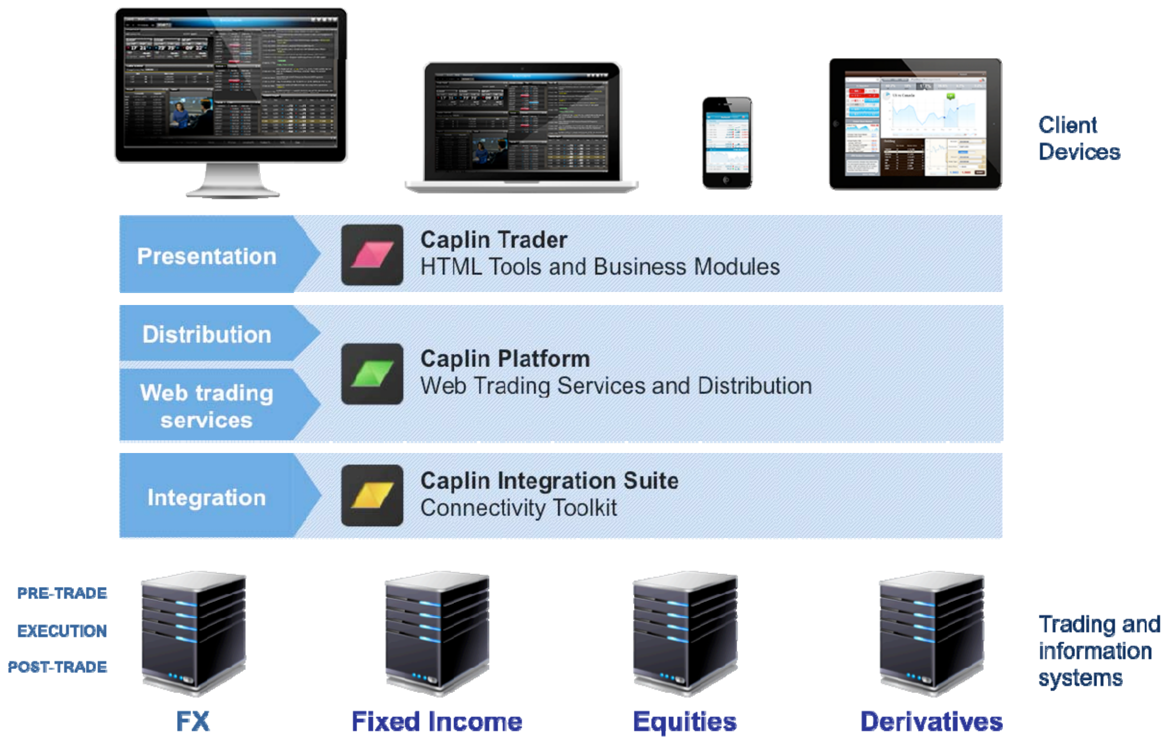


Figure 2–2 Web trading stack with Caplin products

A **web trading app** is a browser-based client application for trading over the web. In recent years such applications have become very advanced, providing functionality previously only seen in installed applications.

2.3 What is the Caplin Platform?

The Caplin Platform is a suite of server applications and auxiliary components that can be used to distribute streaming data (typically financial data, but not restricted to this), and manage trades. The Caplin Platform manages client sessions, whether connected from web trading apps or from installed applications, and provides them with real-time access to pricing and trading systems.

The following diagram shows where the Caplin Platform fits into a typical web trading stack.

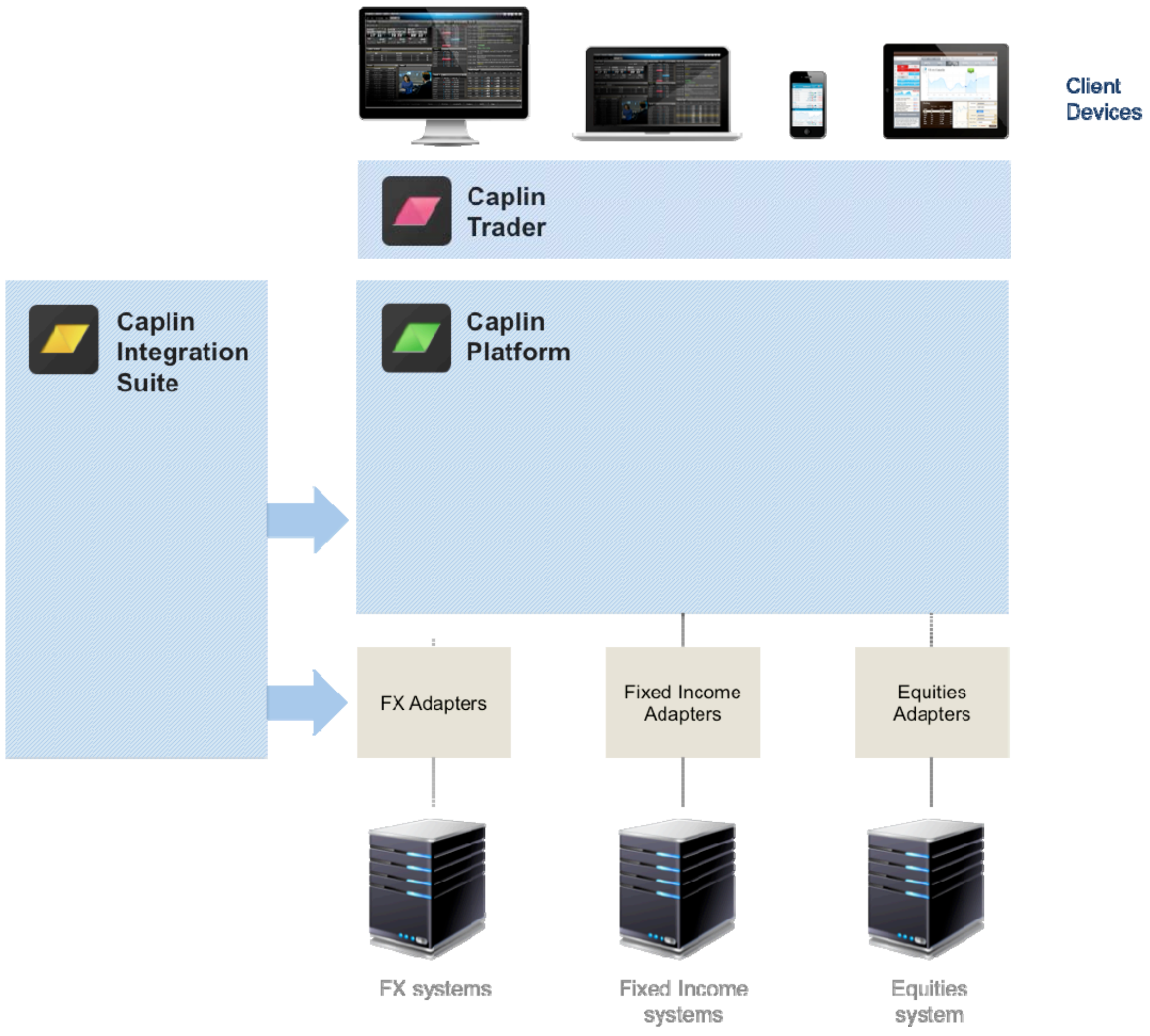


Figure 2–3 The Caplin Platform

The Platform includes two main server applications, Caplin Liberator and Caplin Transformer.

Caplin Liberator is a financial Internet hub that delivers data and messages in real time to and from subscribers over any network. **Caplin Transformer** is an event-driven, real-time data transformation engine optimized for web trading services. The services are typically implemented in Transformer modules, but Liberator also provides built in services and modules for permissioning and monitoring. Both Liberator and Transformer are designed to perform and scale to meet the needs of large web trading platforms.

Liberator and Transformer are typically connected to **Integration Adapters** that allow the Platform to access external systems and services. You develop an Integration Adapter using the **Caplin Integration Suite** (see section 5.2).

The following diagram shows the components of the Caplin Platform integrated with the systems of a single asset class.

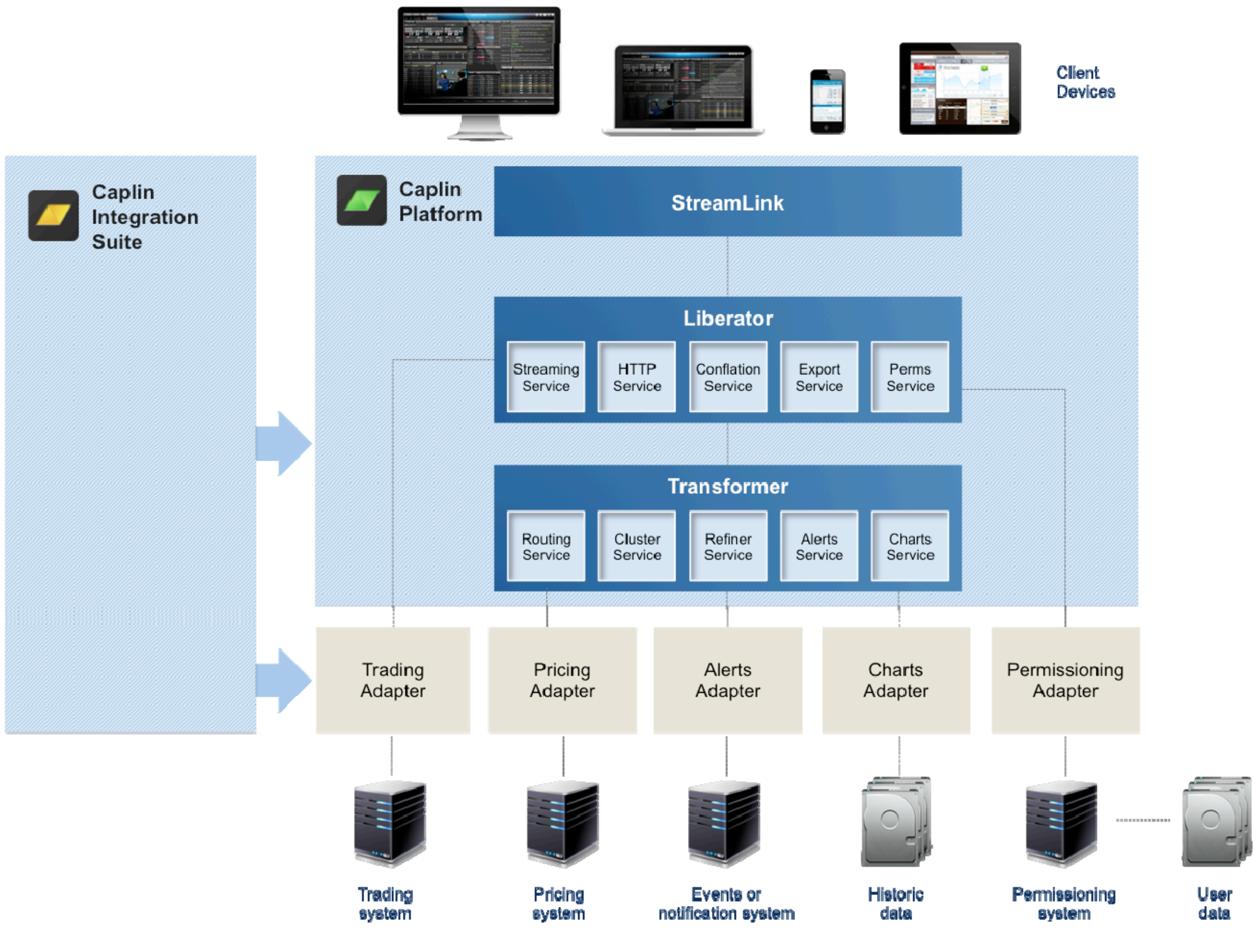


Figure 2–4 The Caplin Platform components

You can deploy the components of the Caplin Platform and the required Integration Adapters in a number of ways. For example, the **Caplin Platform Deployment Framework** makes it easy to deploy, run, and upgrade applications and Integration Adapters (see “Caplin Platform Deployment Framework” on page 25).

2.4 How does the Caplin Platform help me build my web trading platform?

The Caplin Platform provides the functionality and services that are required to develop and deploy a web trading platform. This helps you to achieve shorter development cycles, with fewer defects, and therefore reduce time to market at lower cost.

The Caplin Platform is an open architecture. It supports a number of different technologies both for developing client applications and for developing integration adapters on the server side. This makes it possible to integrate with mixed technology stacks across multiple departments or teams, and if you change your client technology, you do not have to redevelop the whole technology stack.

3 Platform features

The Caplin Platform provides a number of features, implemented throughout the software stack, that make the Platform ideal for web trading. The following sections describe some of the main ones.

3.1 Web streaming

At the core of the Caplin Platform is a powerful web streaming server called Liberator.

Web streaming allows data to be sent from the server to a web browser asynchronously and with low latency. There was no proper support for this capability in the early web; adding it makes full duplex client-server communication possible.

Web streaming has had many different names, from reverse AJAX to Comet, and now a number of different techniques are available that make it possible to stream data to a wide range of web browsers. Between them, StreamLink and Liberator implement the most appropriate of these techniques, including the latest HTML5 WebSocket. StreamLink hides the different implementations from developers; this allows you to concentrate on building the business logic of your web trading app, rather than implementing the communications technology.

3.2 Low latency messaging

The Caplin Platform pushes data between server components, and to and from client applications. The latency of data delivery can be critical in trading platforms, so keeping it to a minimum is a high priority. The Caplin Platform is regularly benchmarked to ensure that its message latency is as low as possible.

3.3 Normalization

The Caplin Platform is designed to integrate your existing systems and present end-users with a more consistent view into these systems. The Platform can integrate with multiple systems that handle different asset classes and use different technology. These systems are connected to the Caplin Platform via Integration Adapters, which convert data and messages into normalized formats for onward distribution to Caplin Platform components, and hence to client applications.
Caplin Transformer can also be used to normalize data according to your requirements.

3.4 Throttling and batching

The Caplin Platform can handle very high data rates. If a client application subscribes to a lot of data, or the data rate is just very high, it can be beneficial to control the amount of data being sent to a client. This can be done in two different ways.

When data is sent to the client it is the latest value of any of the fields that have changed. If the update rate exceeds a configured threshold, the data can be throttled, meaning that not all of the updates are sent to the subscribed clients. For example, if an item is updated 20 times a second, you could configure the Platform to only send out the latest values every 250 milliseconds. If an item is updating at a lower rate than this threshold, it will be sent out immediately. You can set different throttle levels for different items of data.

The other way data flow can be controlled is by batching messages together into larger packets containing multiple messages, rather than by sending each message individually. Batching makes better use of the network and the resources on the server.

Both throttling and batching add some latency to the messages, so it is important to configure these features to suit the profile of your end-users and how they use the web trading platform.

3.5 Monitoring

You can monitor the Caplin Platform components and Integration Adapters (including custom Adapters built using the Caplin Integration Suite). Each component and Adapter exposes its internal state using a JMX^{TM} interface. This information can be viewed using the Caplin Management Console, or you can use the JMX API to implement custom monitoring tools.

The monitoring interface can also be used to perform certain actions, for example ejecting a user.

3.6 Active subscriptions

In a pure publish/subscribe model, publishers are completely decoupled from subscribers, with a component between them linking them together. This means publishers have to constantly publish all their data. However, this model does not always fit in with the style of the application; for example, it does not work well if subscriptions are for data that needs to be created dynamically. Implementations often work round such limitations by arranging for publishers to subscribe to a known channel and listen for requests to start publishing data.

The Caplin Platform handles this style of subscription from the ground up. When a client application subscribes to data, if that data is not already subscribed to by another client session, Liberator sends a subscription request to the relevant Integration Adapter(s). This allows the Integration Adapter to handle the subscription dynamically and create a custom data stream based on the subscription details.

This style of subscription is called an active subscription. It is not just a benefit for custom data streams; it also reduces traffic for standard data streams, as only the streams being subscribed to by clients need to be sent between Integration Adapters and Platform components.

The Caplin Platform also supports broadcast subscriptions, which can be useful in some cases, such as when an external data-supplying component only sends out data in a broadcast manner.

3.7 Failover

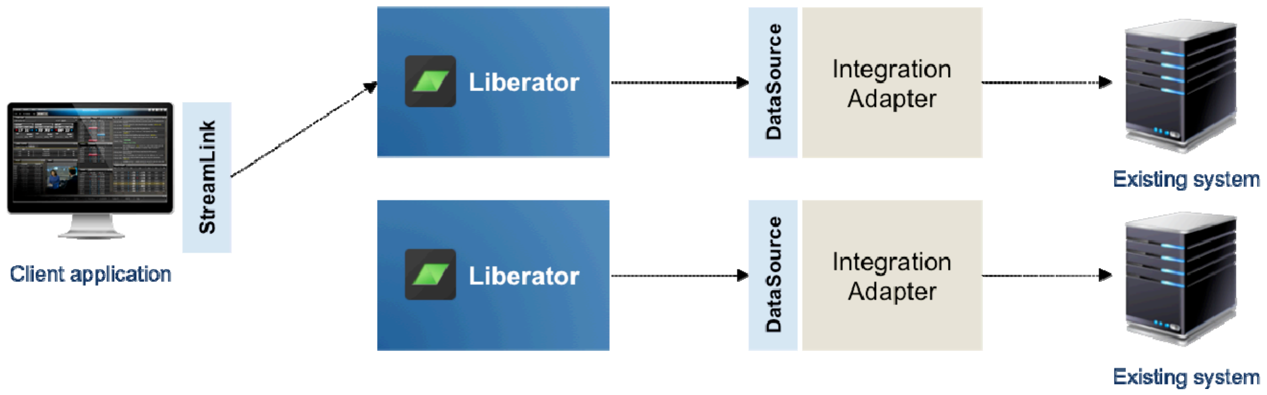
The Caplin Platform can be deployed with multiple instances of any of the components. Failover of these components is built in, and the failover strategies can be configured.

Client Failover

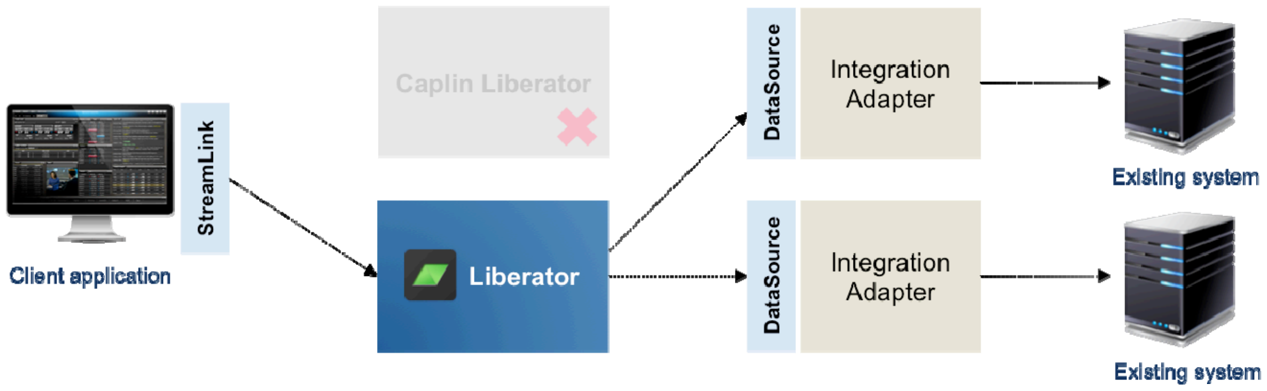
If the Caplin Platform is deployed with multiple Liberators, client applications can be configured (through StreamLink) to be aware of the Liberators available.

On first connecting, a client application chooses which Liberator to connect to, based on a configured algorithm. Typically all instances of Liberators are made available as primary nodes, sometimes referred to as 'live/live' or 'hot/hot', but other scenarios are supported too.

The following diagram shows a typical scenario with a client connected to one Liberator



If a client application's connection to its Liberator fails, StreamLink automatically attempts to reconnect to the same Liberator. If that fails, StreamLink uses the configured algorithm to choose another Liberator. The following diagram shows this.



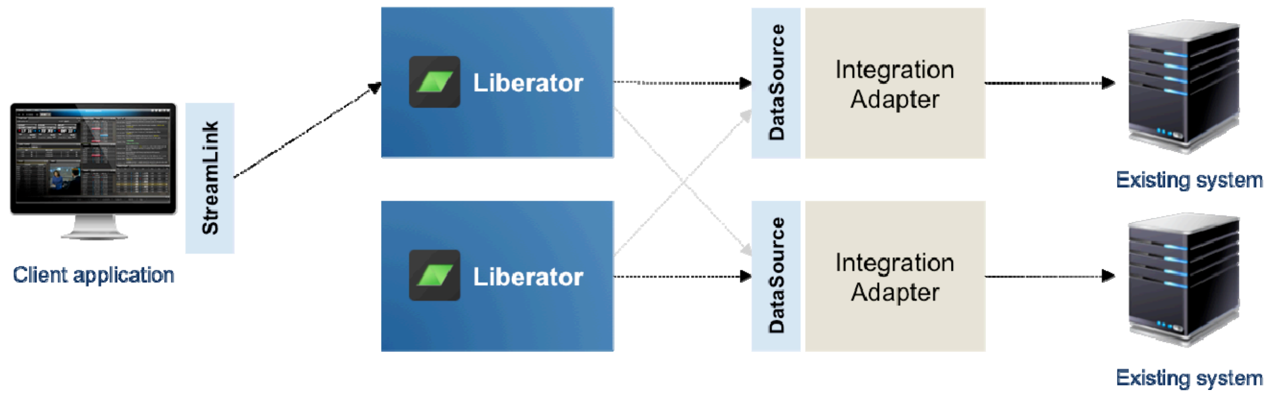
StreamLink handles the reconnection logic. When reconnection is in progress, it makes callbacks on the client application to indicate the status of the connection. The application can then indicate this status to the end-user, or take some other appropriate action.

Server Failover

A typical Caplin Platform deployment has a number of instances of Liberator, Transformer, and Integration Adapters. There may be multiple types of Integration Adapter, each type handling different data and/or functionality, and multiple instances of each type.

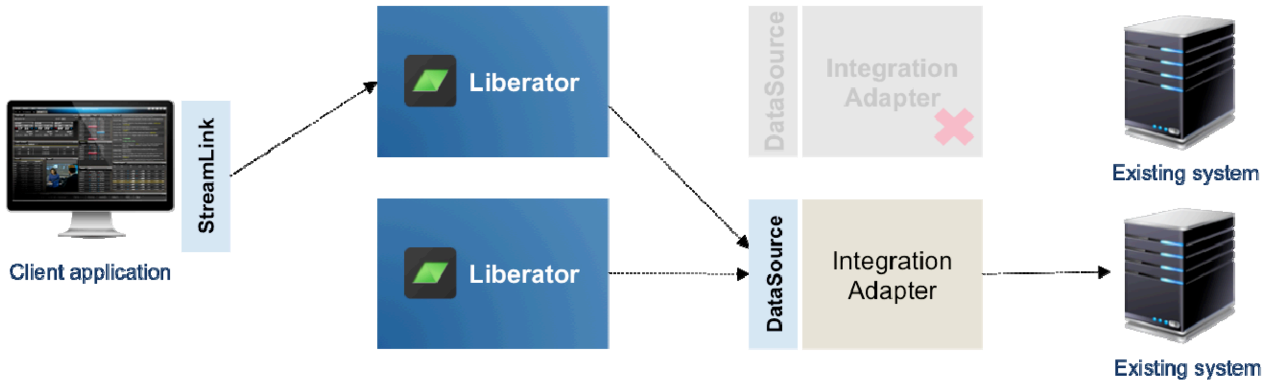
Liberator, Transformer, and all Integration Adapters are built on the core integration API of the Caplin Integration Suite (the DataSource API), which handles the connections and communication between these components. The connection topology and parameters are set up through configuration, and the configuration also defines how failover between these components should work. A number of failover configurations are supported; for example, subscriptions can be balanced across a number of Integration Adapters, or the failover strategy can be determined by the priority of each Adapter.

The following diagram shows a typical setup with both Liberator instances connected to both Integration Adapter instances.



When a connection between components fails, it is re-established whenever possible. If failover is required, existing subscriptions to data automatically failover between components. At all times, changes in the status of subscribed data are passed through to client applications so they can be indicated to the end-user (for example “data has become unavailable”).

The following diagram shows Liberators using the remaining Integration Adapter instance when the other instance fails.



3.8 Caching

The Caplin Platform acts as a real time cache of the data provided by the Integration Adapters.

Liberator holds an in-memory cache of the latest values of any data currently subscribed to. When a new client subscribes to the same data, the latest values of that data can be immediately returned from the cache to that client without needing any further interaction with the sources of the data. This can improve performance, and in some cases it can compensate for the system that supplies the data not having a current value cache of its own.

Transformer can also cache any data that is routed through it. You can make use of this cache when using the Transformer API to implement custom behavior or services.

3.9 Single Sign On

A web trading app using the Caplin Platform can use Caplin KeyMaster to easily integrate with a Single Sign On (SSO) system. This allows a web app that interacts with a traditional application server or web server to use existing sign on functionality when authenticating end-users. Subsequently, the end-users can be seamlessly logged in to Liberator without them having to enter any further user credentials.

KeyMaster creates secure authentication tokens (user credentials tokens) that Liberator uses to authenticate login requests. This method works with any single sign on system.

3.10 Permissioning

A comprehensive permissioning capability is built in to the Caplin Platform and its APIs.
This allows Liberator to control access to the system and to data in the system.

There are three main aspects to the permissioning system.

♦ Authenticating user logins.

♦ Authorizing subscriptions to data.

♦ Authorizing publishing of messages.

These aspects can all have custom implementations, and you can integrate them with an external system, such as a user permissions database.

3.11 Lists, filtering, and sorting

The Caplin Platform and Caplin APIs provide easy and efficient ways to manage and display lists of data. Such lists are known as containers.

A client application can make a single subscription to a whole container of real time data. Items can be added to, and removed from, the container in real time, and these changes are immediately sent to the client application.

If the container is large (contains a long list), a client application can subscribe to just a portion of the elements in it. This feature can be used to ensure that only the data that will fit on an end-users screen is sent, so the client application does not have to manage data that the end-user does not see. This technique significantly reduces both bandwidth and client processing load.

The Refiner module in Transformer provides a service to filter and sort containers. A client application can subscribe to a container, specifying filter and sort criteria. The Refiner module dynamically creates a custom container that matches the criteria, keeping the container up to date as the data within it changes.

3.12 Datatypes

The Caplin Platform supports a number of datatypes that can be published and subscribed to.

♦ The majority of data is represented by the *Record* datatype, which is a set of key/value pairs, like a hashtable.

♦ The *Container* datatype is described in 3.11 “Lists, filtering, and sorting”.

♦ There are datatypes that represent other forms of data in an efficient way for streaming over the Internet. They include *Level 2 Records*, *Pages*, *News headlines, News stories,* and *Permissions*.

Using the datatypes available in the Caplin Platform has the benefit that many of the built in features of the Platform are designed to work with them. For example, throttling of data works with the *Record* datatype, and the Platform understands that record fields can update independently of each other. This means the throttling feature can be more efficient than when dealing with a datatype whose internal structure is unknown to the Platform.

3.13 Services

The Caplin Platform can be used to provide services to client applications. These may be core services provided by the Platform itself, additional services implemented by Caplin, or custom services you have built to run in the Platform.

Core services include the ability to subscribe to a subset of a container. Additional services offered by Caplin include the Refiner Service blade, which allows filtering and sorting of list data (see section 3.11),

Caplin Transformer provides an API to implement services as Transformer modules. For example, The Refiner Service blade is implemented as a Transformer module. A module has access to the data in the Platform, and can receive requests from client applications via Liberator.

4 Caplin Platform architecture

The Caplin Platform is an integrated suite of software. This section describes the architecture of the Platform, how it works, and the components that comprise it.

4.1 Overview

A web trading platform built using Caplin's software and tools consists of a number of components.

The following diagram shows the full software stack in a typical deployment, including the APIs from the Caplin Integration Suite used to build the Integration Adapters. Note that a production deployment would have multiple instances of many of the components, to provide scalability and fault tolerance.

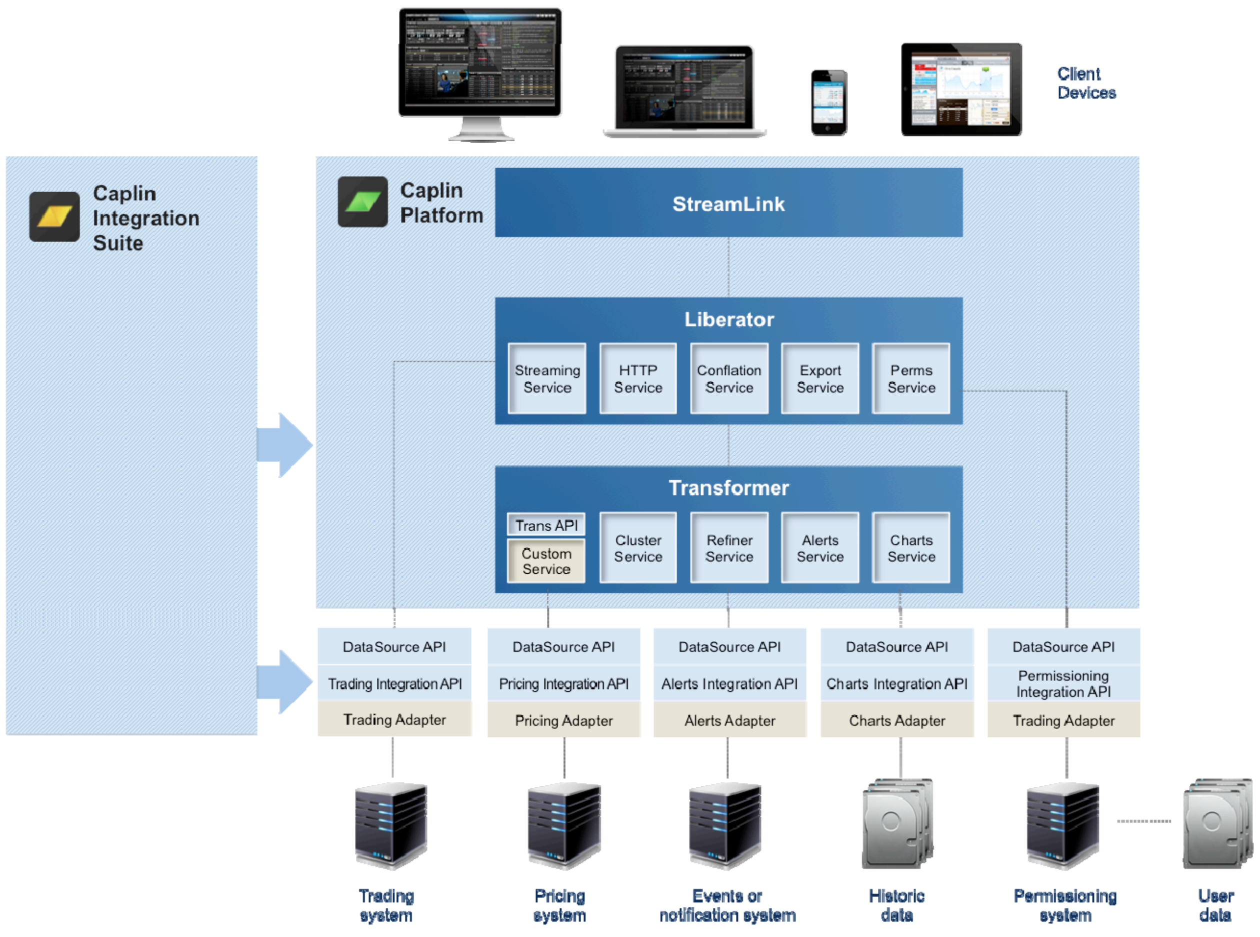
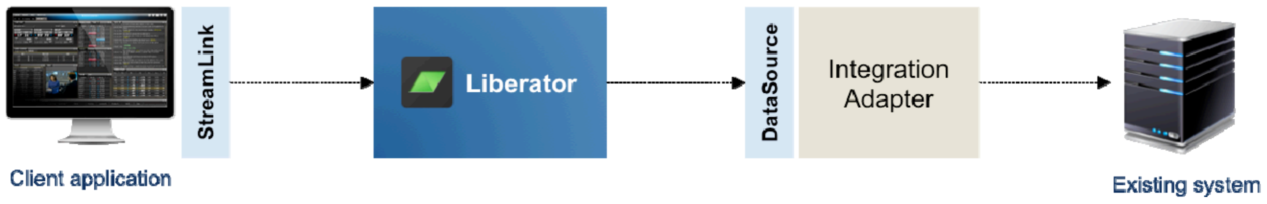


Figure 4–1 Caplin Platform architecture

4.2 How does the Caplin Platform work?

The Caplin Platform lies at the heart of a web trading platform. It enables web apps and installed applications to publish data, subscribe to data, and interact with real-time systems such as pricing and trading.

Here is a simplified view of the Caplin Platform's software stack.

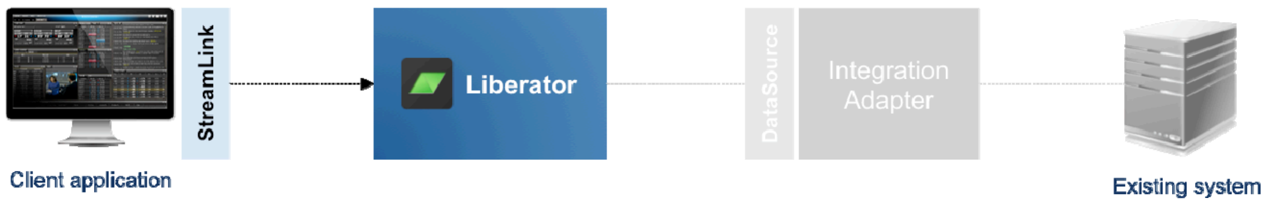


From left to right, the stack consists of:

- A client application built using StreamLink
- The Caplin Platform itself
- An Integration Adapter built using the DataSource library from the Caplin Integration Suite (see section 5.2)
- An existing system to which the Platform is integrated via the Adapter

Connections and StreamLink

Caplin Liberator (part of the Caplin Platform) accepts connections from client applications that have been developed using the StreamLink API.



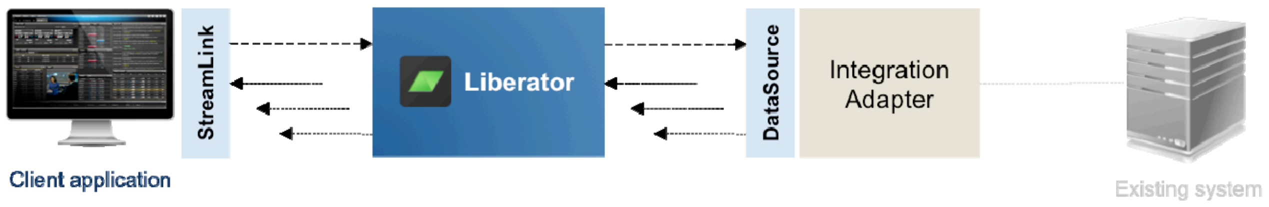
These connections can be made in a number of ways; the StreamLink API chooses the most appropriate connection type for the technology and software environment being used.

If the client is a modern browser, StreamLink uses HTML5 WebSocket connections. When StreamLink runs in older browsers it applies various different techniques to establish a streaming connection to Liberator. Other types of (non-browser) application can also use StreamLink to connect, using either direct socket or HTTP(S)-based methods.

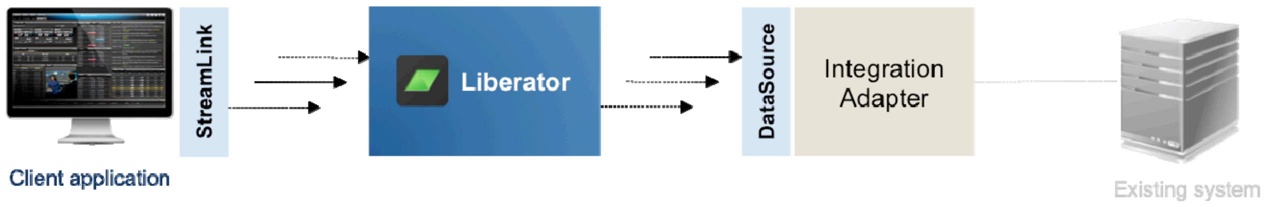
Sessions and publish/subscribe

Liberator maintains client sessions, services subscriptions to data, and handles the publishing of messages. When a client application subscribes to data, Liberator has to retrieve a stream of data to supply to the client. This subscription may be for some common data that other clients are subscribed to, or it may be for some private data specific to the user.

In both cases, if Liberator is not already subscribed to the stream of data it sends a subscription request to a Transformer or Integration Adapter that can supply the data. When the data is received from the Transformer or Integration Adapter, Liberator processes it and sends it on to all subscribed clients.



Client applications can also publish data to Liberator. Typically, the data is passed on to a suitable Transformer or Integration Adapter to be handled in a custom way. This technique is often used in trading applications to send trade messages from the client to a trading system.



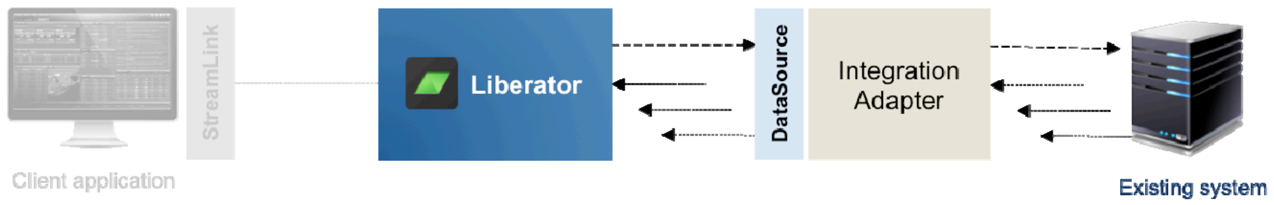
DataSource applications and Integration Adapters

DataSource applications

A DataSource application services subscriptions for data on behalf of Liberator. DataSource applications communicate with other DataSource applications (which include Liberator and Transformer) using TCP/IP sockets. DataSource applications are developed using the Caplin Integration Suite.

Integration Adapters and the Caplin Integration Suite

Typically a DataSource application is an Integration Adapter that simply acts as a bridge between the Caplin Platform and the API or protocol that an existing service or system uses. To do this, it uses one of the APIs in the Caplin Integration Suite. The Adapter processes subscription requests, passing them on to the existing system, and passes back to the subscribing Liberator the data updates returned by the system, as shown here:



The DataSource library

The Caplin Integration Suite contains a DataSource library that handles a DataSource application’s connections to other DataSource applications, and manages failover to alternative DataSource applications if connections fail.

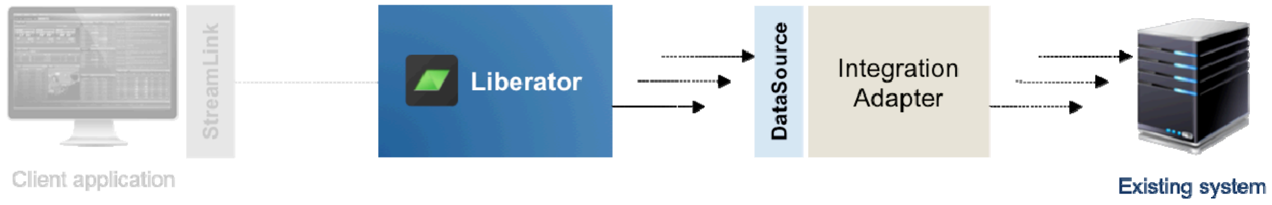
The DataSource library also converts data to and from the DataSource protocol, which is the common protocol that DataSource applications use to communicate with each other. Using the DataSource Library, an Integration Adapter can communicate with the Caplin Platform and onward to StreamLink-based client applications.

Multiple Integration Adapters

Multiple Integration Adapters can work alongside each other, providing access to different types of data (for example data about FX instruments, and data about FI instruments), or as multiple instances providing the same type of data (for example multiple connections to the same system for fault tolerance or load balancing).

Data published by clients

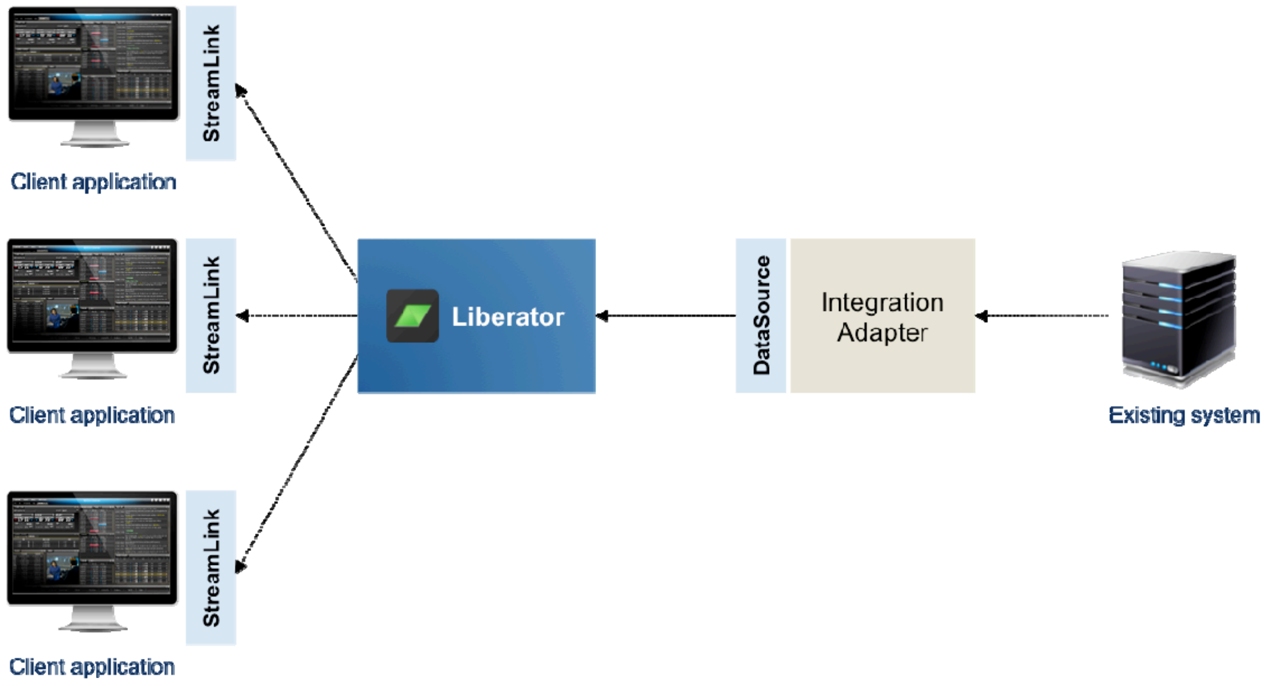
DataSource applications can also receive data published by clients, as shown here;



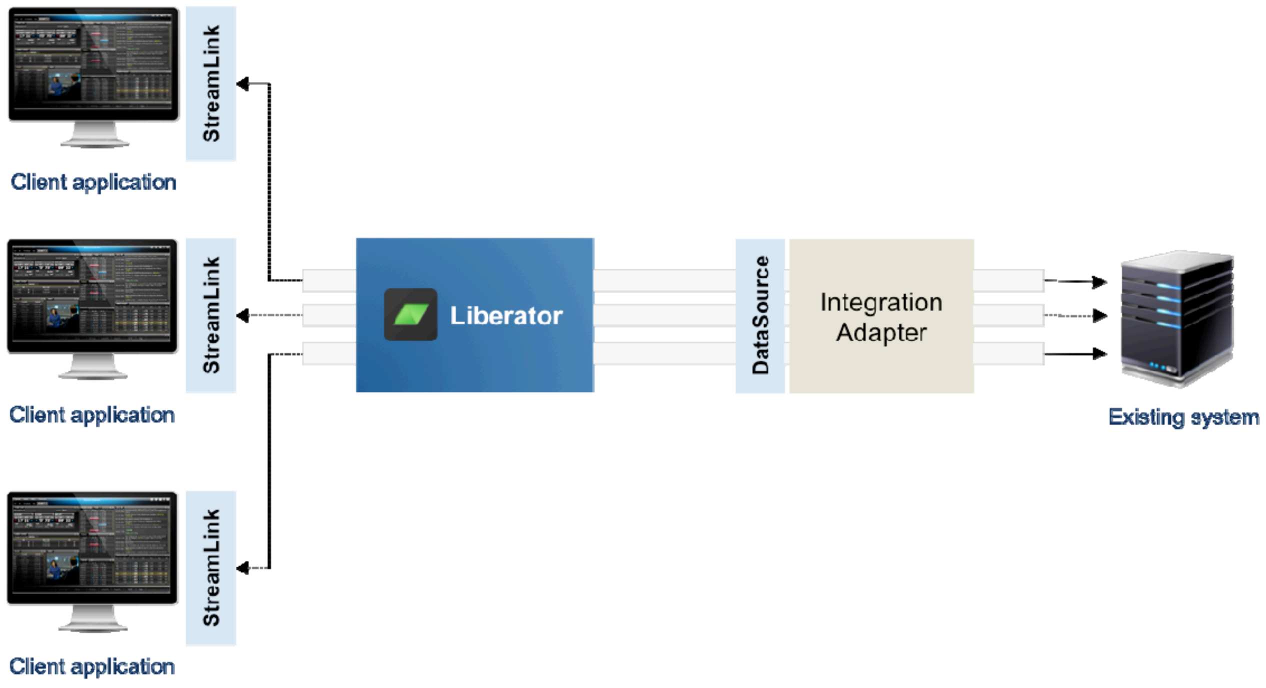
It is the job of the DataSource application to process this data. It could be private data relating to a trade, which would be passed on to a trading system to process, or it could be data to be re-published back into the Platform and thence on to other client applications.

Fan-out or channels

The Caplin Platform can be used in a number of ways. When subscribing to common data a fan-out approach is used. Subscriptions to the same data item from a number of clients are managed by Caplin Liberator allowing a single subscription to be made from Liberator to the backend systems. When updates are sent from the backend systems, Liberator sends the data to all subscribers. The following diagram shows this style of communication.



The other style of communication is using channels. This is often when the data is either bidirectional or private to that session. This is often used for trading or for clients subscribing to custom data with parameters specific to that client. The following diagram shows this.



Presentation versus Data

A client application usually consists of a presentation part and data. For a typical web app, the resources needed to run the app are typically HTML, CSS, JavaScript, and images, which make up the presentation part of the application. Both the presentation resources and the application logic are loaded from a traditional web stack that consists of web servers, application servers, and databases. On the other hand, for an installed application, the resources are usually compiled code in an executable form.

The data in the application is separate from the presentation. In a trading application, much of the data is updated in real time or is transient (there can also be historic data, but this is often updated in real time too). A trading application gets its data from the Caplin Platform by subscribing to it and receiving a stream of updates. It then uses the presentation logic to display the updated data to the end-user.

The following diagram shows the two complementary stacks providing data to the same client application

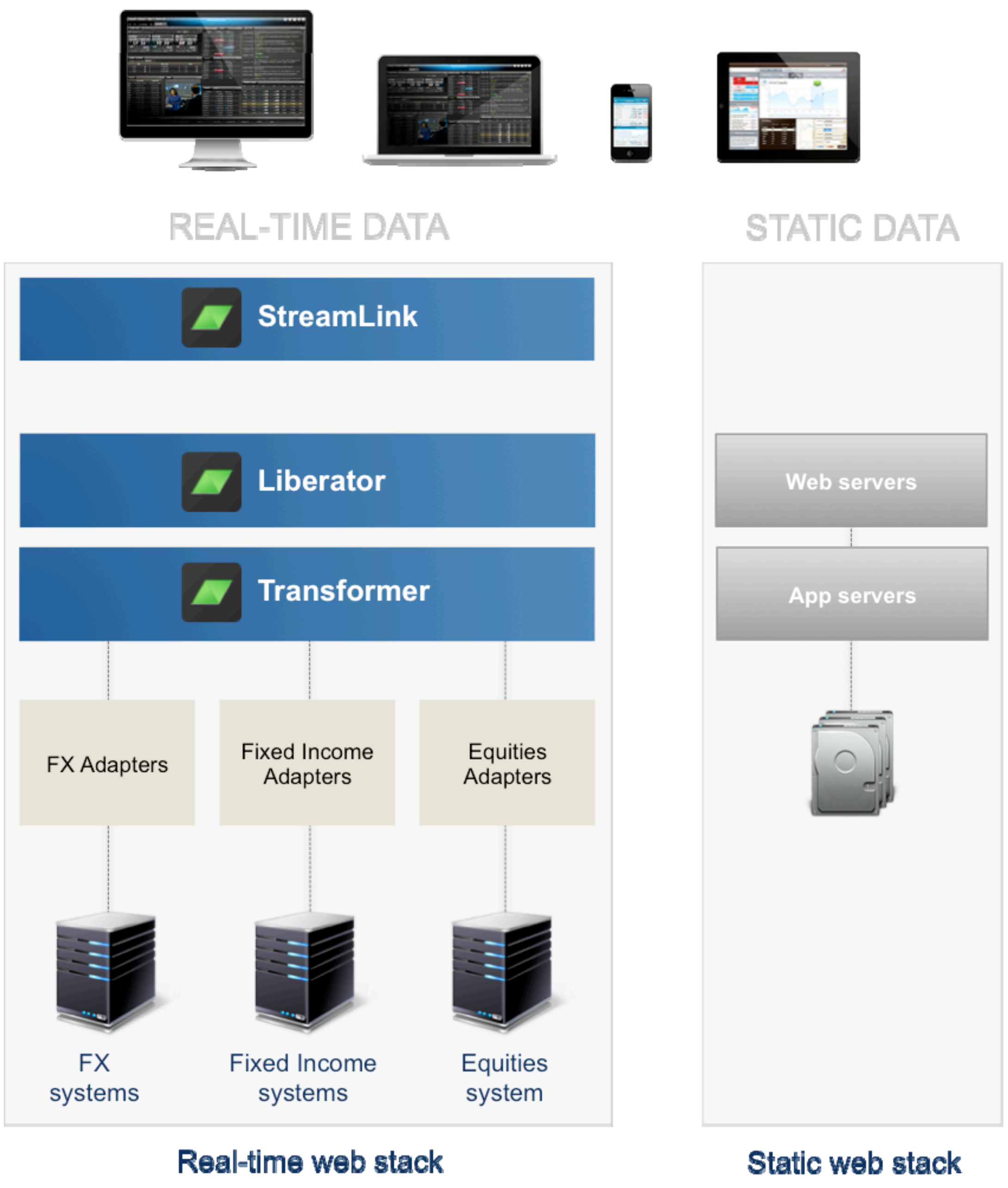


Figure 4–3 Presentation and Data separation

Because the web trading app gets its presentation logic and resources from a traditional web stack, and its data from the Caplin Platform, this decoupling of logic+resources and data gives you more flexibility in how you develop and deploy your app.

4.3 Core components

At the core of the Caplin Platform are the Liberator and Transformer server components.
Both of these components are DataSource applications.

Caplin Liberator

Caplin Liberator is a financial Internet hub that delivers data and messages in real time to and from subscribers over any network, tunneling automatically through proxy servers and firewalls. As the result of many years of intensive development and optimization, Caplin Liberator is the world’s best streaming server for web trading apps.

At its core, Liberator is a streaming server, supporting many types of connections to web browsers and other installed application technologies. It uses the latest web connection techniques, such as WebSocket, whenever suitable.

Liberator handles client sessions, and provides publish/subscribe capabilities to link clients to services such as pricing and trading systems. A single Liberator instance can support thousands of concurrent client sessions, managing their subscriptions to data and processing their published messages.

Liberator is more than just a generic streaming server; it provides explicit support for web trading apps. For example, it supports the concept of ‘containers’, a data type that represents a list of data in an efficient manner, and which has many uses in representing financial data (see section 3.11 “Lists, filtering, and sorting”).

Liberator is written in C and runs on Linux®. Caplin also provides a Windows® and a Mac® version of Liberator for use in development environments only.

Liberator provides two extension points: auth modules and monitoring modules.

Auth modules

An auth module allows you to implement custom code to handle authentication and permissioning (authorization). You can develop Auth modules in C or in Java^{TM}. Caplin provides a number of ready-to-use auth modules, including the Permissions Auth Module, which works in conjunction with the Permissioning Integration API (see “**Permissioning Integration API**” in section 5.2).

JMX monitoring module

The JMX monitoring module allows JMX client applications to monitor and manage the internals of Liberator that are exposed by the module.

Caplin Transformer

Caplin Transformer is an event-driven, real-time data transformation engine optimized for web trading services. These services are implemented in Transformer Modules. Transformer is a DataSource application and can therefore communicate with Liberators and Integration Adapters.

You can use the Transformer API to create your own services as custom Transformer modules. Such modules can act on subscriptions and streaming data. Transformer houses the modules and manages the data coming in and out of them. It provides a number of services to modules, so your module implementation can concentrate on manipulating data. These services include data caching, and routing of subscriptions and data.

Transformer is written in C and runs on Linux. Caplin also provides a Windows and a Mac version of Transformer for use in development environments only.

You can develop Transformer modules in C, Java, and Lua.

Caplin provides a Transformer module called Refiner that allows large lists of data (containers) to be efficiently filtered and sorted in real time on behalf of client applications (see 3.11 “Lists, filtering, and sorting”).

Caplin Platform Deployment Framework

Once you have developed Caplin Platform components and Integration Adapters using the Caplin Integration Suite, you can deploy them independently of each other and in a way that fits in with your existing conventions and policies. However, you can also use the Caplin Platform Deployment Framework, which makes the deployment process easy and flexible.

The Deployment Framework is a directory structure, into which you can deploy both applications and configuration. It provides a highly convenient way to deploy new functionality across multiple machines. The framework uses the concept of a Caplin Platform blade, which is a package containing an application and the configuration required for it to run on the Caplin Platform.

For example, an Integration Adapter could be packaged together with its own configuration and any configuration changes needed for Liberator and Transformer to support the Adapter. This Platform blade would then be deployed on each Liberator and Transformer machine, as well as the machine the Integration Adapter runs on. The following diagram shows how blades deploy into a typical setup.

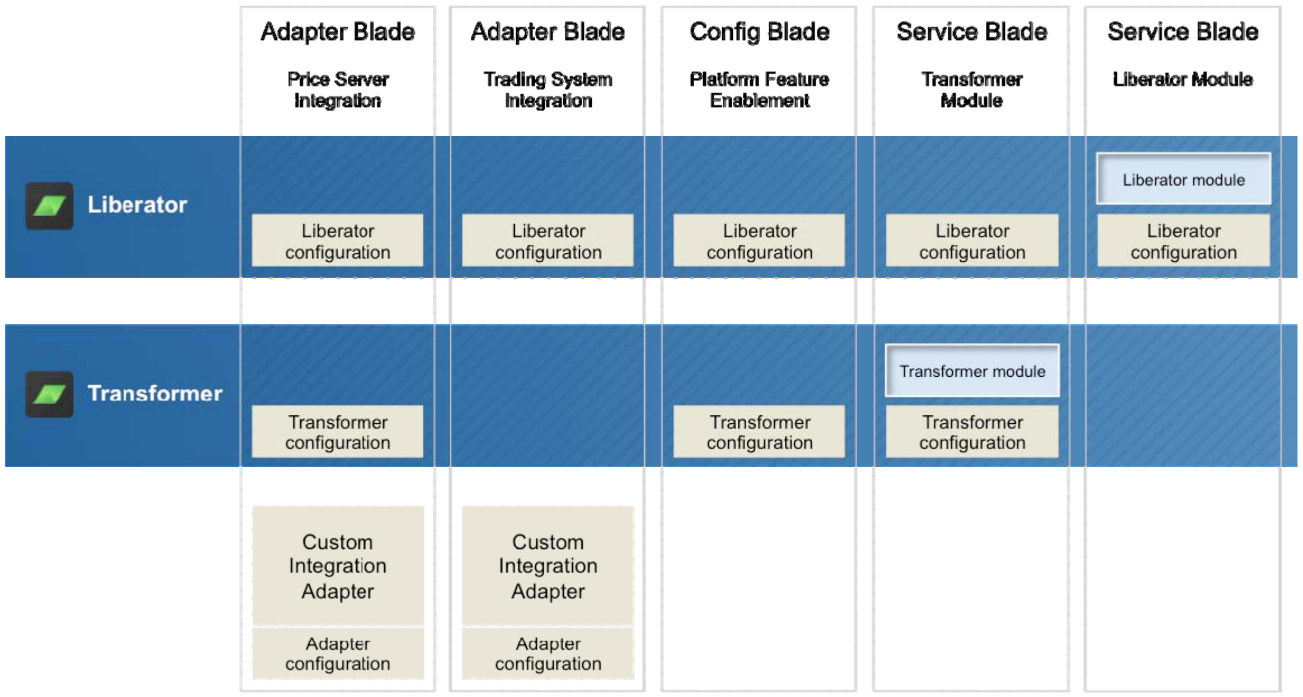


Figure 4–4 Deployment Framework and blades

For more about the Deployment Framework, see the document **Caplin Platform: Deployment Framework Overview**.

More about Caplin Platform blades

The Deployment Framework supports three types of Caplin Platform blade:

- An *Adapter blade* consists of an Integration Adapter that uses one or more of the Pricing, Trading, or Permissioning APIs from the Caplin Integration Suite. The blade includes the Adapter's configuration, and any configuration required for Liberator and Transformer.
- A *Service blade* consists of a Transformer module or a Liberator Auth module that provides a service. It also contains any configuration required for Liberator and/or Transformer.
- A *Config blade* enables a feature in the Caplin Platform entirely through configuration (it does not contain any executable components).

The Deployment Framework comes with a number of Config blades to enable particular Platform features, and Caplin can also supply a number of other blades such as the Refiner Service blade and the RMDS Adapter blade.

You can easily create your own Platform blades by using the tools in the Caplin Integration Suite. Once created, you can package a blade as a zip file and deploy it to any machine that has been set up with the Deployment Framework.

For more about Caplin Platform Blades, see the document **Caplin Platform: Deployment Framework Overview**.

4.4 Additional components

There are a number of additional components in the Caplin Platform that provide extra functionality in conjunction with Liberator and Transformer.

Caplin Management Console (CMC)

Liberator, Transformer and other DataSource applications all expose internal data via a monitoring interface, which is available via a JMX API. The Caplin Management Console (CMC) is an installed Java application that connects to this JMX API and gives you an easy to use interface for viewing the available monitored data.

Information that you can monitor with the CMC includes:

- Which users are logged in to Liberator, and what data they are subscribed to
- The state of the connections between Caplin Platform components
- The rate at which messages are send to, and received by, a component
- Statistics about the data sent over each connection

You can extend the CMC to show different views of the data available, or to create views of new data in your custom-built DataSource applications, such as data in Integration Adapters.

Caplin KeyMaster

KeyMaster allows you to integrate access to the Caplin Platform with a single sign-on (SSO) system. This allows end-users to log in just once to a website or application; when the application needs a streaming connection to the Caplin Platform (Liberator), the end-user does not need to enter credentials for a second time in order to be authorized by Liberator.

KeyMaster is a simple API used to create a Java Servlet or ASP.NET application. The servlet or application generates secure user credentials tokens for client applications. The credentials token is used by the client application to log in to Liberator without the end-user needing to provide additional credentials. The Java Servlet or ASP.NET application is configured to be protected by an existing SSO system, so it integrates with other applications that use the same login credentials.

KeyMaster also includes a tool to generate an encryption key pair. One key is used by KeyMaster to generate user credentials tokens, and the other key is used by Liberator to verify these tokens.

KeyMaster’s design allows it to be used with any SSO system.

Caplin Director

Director is a complete user administration system for the Caplin Platform that allows you to control how end-users can access the Platform and what they can do on it.

You can define permissions and assign them to users or groups of users. This dictates what data people can access, what instruments they can trade on, and what types of trading they can do. Director can also control what data people receive, for example by mapping users’ subscriptions to different price bands based on who they are, or based on how much value they are trying to trade.

Director consists of a web-based administration interface, a database, and an Integration Adapter that uses the Caplin Integration Suite’s Permissioning Integration API.

4.5 Deployment Environment

The Caplin Platform is typically deployed on a number of networked machines, to provide resilience, load balancing, and secure operation.

Liberator, Transformer, and any other DataSource applications (such as Integration Adapters) are commonly deployed as multiple instances. Where high numbers of users or high data rates are required, this increases the capacity of the system and helps to balance the load. Multiple instances also add resilience to the system, since all DataSource applications, including Liberator and Transformer, can be configured to fail over automatically.

It is also common practice to run each component on a dedicated machine, and to place different component types in different network zones for security purposes. The actual arrangement is usually dictated by security policies. Client applications connect to Liberator, which must therefore be internet facing. Transformers and DataSource applications (such as Integration Adapters) are typically deployed on a separate network behind firewalls, and the systems that the Integration Adapters communicate with are often on yet another network.

The following diagram shows a typical network deployment. In a full production system, there could be multiple instances of some of the Integration Adapters.

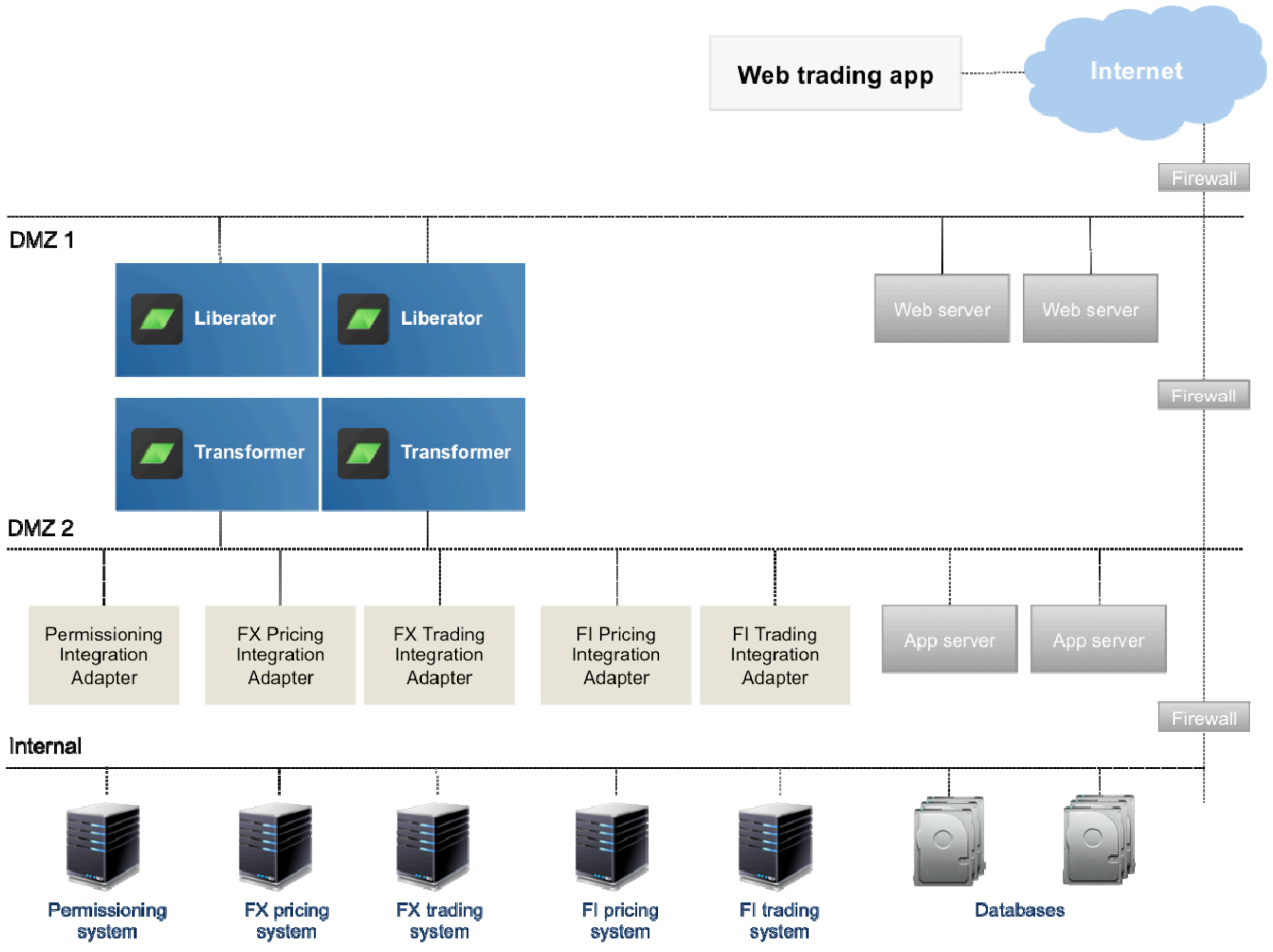


Figure 4–5 Network deployment

4.6 Operating systems supported

The core of the Caplin Platform runs on the Linux operating system. The Integration Adapters developed with the Caplin Integration Suite can run on a number of operating systems, depending on the language used.

Typically Integration Adapters are implemented in Java, in which case they can run on any operating system that supports Java. Integration Adapters can also be written in C/C++ to run on Linux and Microsoft Windows, or in .NET to run on Microsoft Windows only.

5 Developing with the Caplin Platform

5.1 Introduction

Caplin provides APIs and tools to help develop a web trading platform using the Caplin Platform. On the client, web apps and installed applications can be developed with the StreamLink API, and web apps can be developed with Caplin Trader. You can use the APIs in the Caplin Integration Suite to develop DataSource applications, such as Integration Adapters that integrate the Caplin Platform with external services such as pricing, trading, and permissioning.

5.2 Caplin Integration Suite

The Caplin Integration Suite (CIS) is a set of APIs and tools for creating adapters that integrate the Caplin Platform with external systems such as pricing, trading and permissioning.

Integration Adapters are standalone applications that can be deployed and run in whichever way suits your needs. However, they often require additional configuration to be added to Liberator and Transformer. The Caplin Integration Suite includes tools to help you package Integration Adapters and associated configuration into blades. The blades can then be deployed into the Caplin Platform Deployment Framework in a modular way (see “Caplin Platform Deployment Framework” and “More about Caplin Platform blades” in section 4.3).

The following diagram shows an Adapter blade being created using the Caplin Integration Suite and being deployed into the Caplin Platform.

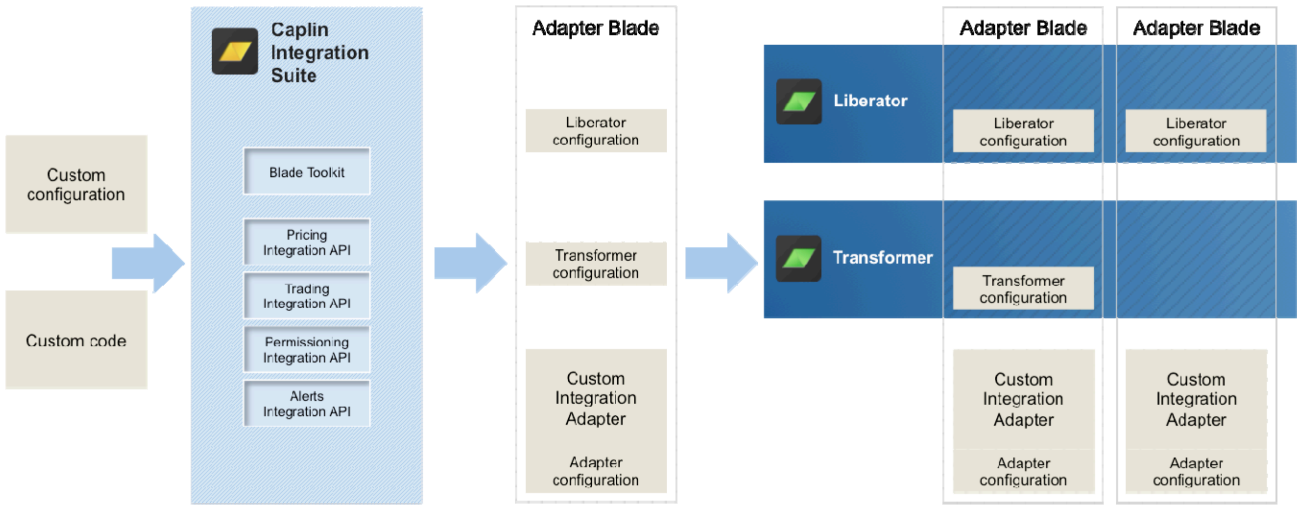


Figure 5–1 Caplin Integration Suite

The Caplin Integration Suite consists of a number of APIs that can be used in separate applications or combined into the same application. The following sections describe the main APIs.

Pricing Integration API

The Pricing Integration API is the core DataSource API used by all DataSource applications. It allows an application to connect to another DataSource application such as Liberator or Transformer and to send and receive data and subscriptions. This API provides access to all the core features of the Caplin Platform.

The Pricing Integration API is available in Java, .NET and C/C++.

For more about the Pricing Integration API, see the **Caplin Integration Suite for Java API Documentation**.

Trading Integration API

The Trading Integration API works alongside the core Pricing Integration API to add trading functionality. The API is designed to make trading integration as quick and simple as possible.

Trade workflows are modeled using a state machine, which is supplied to the Trading Integration API as XML configuration. Trades are managed by this state machine in the library to minimize the code you have to write to handle errors.

The Trading Integration API is available in Java.

For more about the Trading Integration API, see the documents **Caplin Trading: Integrating The Caplin Platform With A Trading System**, and **Caplin Integration Suite for Java API Documentation.**

Permissioning Integration API

The Permissioning Integration API works alongside the core DataSource API and allows you to create Permissioning Adapters or add permissioning capability to other type of Adapter. It sends user authentication and permissioning information to Liberator.

The Permissioning Integration API is used in conjunction with the Permissioning Auth Module that runs within Liberator. The Permissioning Auth Module controls user access to the system and to data. It also sends certain types of permissioning data on to the client application, which enables the client to control how data and functionality are presented according to the permission settings.

The Permissioning Integration API is available in Java.

For more about permissioning and the Permissioning Integration API, see the documents **Caplin Permissioning: Permissioning Overview And Concepts**, and **Caplin Integration Suite for Java API Documentation.**

Eclipse support

The Caplin Integration Suite includes an Eclipse plugin. This allows you to easily create, from within Eclipse, a new Java Integration Adapter project as a Caplin Platform blade. You can develop and run Integration Adapters within Eclipse, or deploy them to another machine. There is also an export wizard for creating blades that can be deployed in the Caplin Platform Deployment Framework.

5.3 StreamLink

StreamLink is an API for writing client applications that communicate with the Caplin Platform.

StreamLink provides a publish/subscribe style API, allowing applications to subscribe to real time data from the Platform and publish data into the Platform.

The API is available in various technologies for building applications of different types, as shown in the following table:

StreamLink API technology	Application types that can be developed
StreamLink JS	Web apps and mobile web apps
StreamLink Java	Installed applications Mobile native apps for Android and Blackberry
StreamLink.NET	Installed applications
StreamLink Silverlight	Web apps
StreamLink C/C++	Installed applications
StreamLink iOS	Mobile native apps for iPhone and iPad

For more information, see the **StreamLink Overview**.

CAPLIN

THE WEB TRADING TECHNOLOGY COMPANY

Contact Us

Caplin Systems Ltd

Cutlers Court

115 Houndsditch

London EC3A 7BR

Telephone: +44 20 7826 9600

www.caplin.com