

STARMOUNT APPLICATION FRAMEWORK

Revision 3.0 -- April 2009



This White Paper, prepared by Starmount, offers insight into end-to-end solutions for the integrated digital signage and retail services industry providing software solutions and services focused on digital signage, interactive kiosks, and other point-of-service products.

Table of Contents

Overview	4
Framework Logical Components	4
Process Flow Controller	5
Service Controller	5
View Controller	5
Software Components	5
Code Security	6
Internationalization	6
Development Tools	6
Application Services	6
Messaging and Monitoring Service	6
Control Messages	7
Event Notification Messages	7
Location Hierarchy	8
Location Groups	8
Location Categories	8
Data Distribution Service	9
Digital Asset Service	9
Asset Input	9
Software Deployment	9
Security Service	10
Encryption Service	11

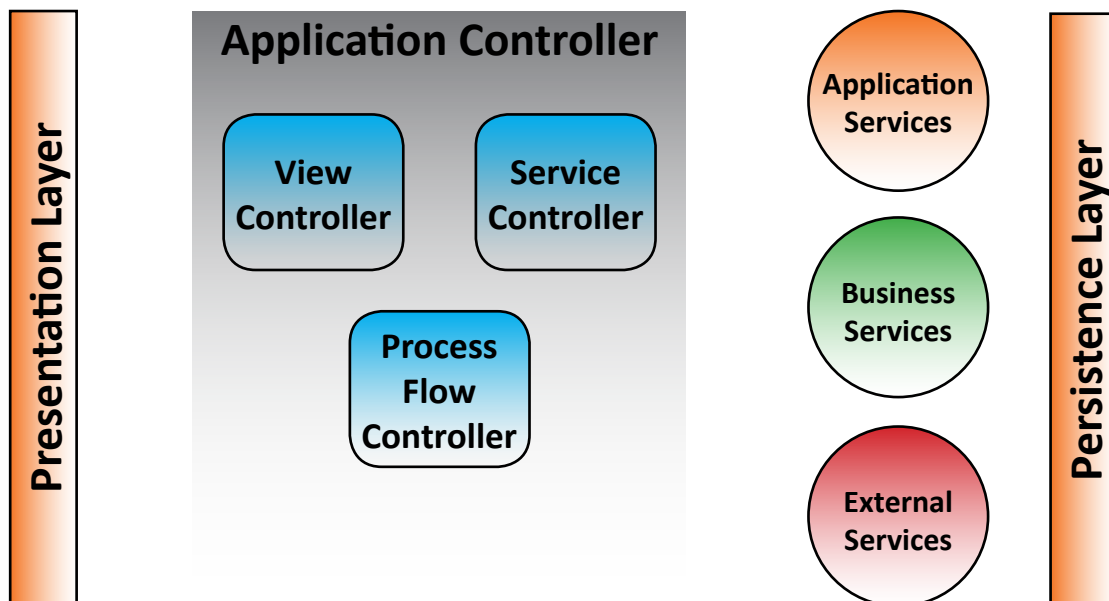
Employee Service	11
Device Service	11
Logging Service	11
System Topology	12
Starmount Application Suite	15
Starmount Dashboard	15
Starmount Media Player	15
Starmount Kiosk	15
Starmount Kitchen Monitor	15
Starmount Digital Menu	16
Starmount Scheduler	16
Starmount Directories	16
Conclusion	16

STARMOUNT APPLICATION FRAMEWORK

Overview

The Starmount Product Framework is a multi-tiered functional structure for deploying retail applications into various heterogeneous environments, integrating with new and existing business systems, and presenting a unified view to the user across multiple form factors. Its component-based architecture provides a clean separation of functional layers allowing for simple modification, expansion, and reconfiguration. The Starmount design philosophy is based on the premise that business environments change rapidly and software should be engineered with the assumption that change will need to be incorporated efficiently, without major rework or redeployment.

The Starmount Framework achieves this flexibility by extracting and isolating business functionality into data-driven formats, such as XML and databases, allowing changes to be incorporated without modification to the code base. This occurs at every level of the Framework—data persistence, service integration, business logic navigation, and screen presentation. Changes from simple interface look and feel modifications to the incorporation of entire new applications can be accomplished by distributing new configuration data.



Logical Component Model

Framework Logical Components

The core of the Starmount Application Framework is a set of loosely coupled components called the Application Controller. These controller objects manage the flow of data, the execution of business functionality, and the presentation of information to the user.

Process Flow Controller

The Process Flow Controller is based on a finite state machine, driven by XML-based node maps, or scripts. Each script describes a series of steps that execute a set of business functionality. At each step, the Process Flow Controller can trigger a call to the Service Controller or request a screen to be displayed to the user. The Flow Controller moves through the node map to the next appropriate step based on the results of the service call or the user interaction.

Service Controller

The Service Controller works as an interface to local and remote business services, existing external services, and application services such as logging, messaging, or device interaction. The pluggable nature of service definition allows new service modules to easily be deployed and integrated into an existing framework application. Standard interface protocols such as SOAP, XML-RPC, JPOS, and JMS are supported, and custom connectors can be quickly implemented for interaction with legacy systems.

View Controller

The View Controller manages data entry and presentation to the user across multiple technologies and form factors. The abstraction of screen definition allows the same view to display in common formats such as Adobe Flex, Java Swing, and Java Server Pages. This gives the Starmount Framework the ability to deploy as a digital signage player, a self-service kiosk, a web-based application, a stand-alone client application, or as a mobile application on a myriad of mobile communication devices.

Software Components

The Starmount Product Framework is built from a standards-based approach that provides a clear integration structure for other systems and avoids dependencies on proprietary technology. The framework uses recognized best practices and design patterns to implement a modular, pluggable architecture, making it easy to replace specific functional components and incorporate new technologies as they emerge.

The use of the Java programming language allows framework applications to be deployed on any major operating system, and this technology-neutral stance is prevalent throughout the framework. Along with the multiple UI technologies mentioned earlier, the framework also can be configured with different persistence technologies to support any standard database system. Also, along with the standard Java Messaging Service (JMS), the framework

can easily be adapted to utilize a different messaging infrastructure. The integration points of the framework present a simple API for developers, and most of the framework components can be replaced and/or reconfigured at runtime without redeployment.

Code Security

Another inherent benefit of the Java environment and of the Adobe Flex client is the standard user interface (UI) deployment has strong security features. Both Java and Flex adopt a similar “sandbox” strategy that only allows code to be executed from specific recognized locations. Additionally, both Java and Flex prevent remote code from accessing the local file system. The Flex code that executes on the client must be packaged in digitally signed files, and the Flex client can only access remote code within its assigned domain.

Internationalization

The framework provides internationalization features that are available to all applications. The base UI components and messaging structures are all internationalized and ready for localization. Static textual data is stored in standard Java resource bundles that are readily understood and accessible to vendors that provide translation services. The framework data repository, as well as the base database schemas, are multi-byte character capable, and user generated content is localizable through the UI interfaces.

Development Tools

The framework provides a number of development tools that enhance the development, building, and packaging of applications. The Developer Toolkit is based on the open source Eclipse IDE that contains a strong set of Java development features, and the framework adds a set of custom plug-ins for UI screen design -- UI Look and Feel design, graphical creation of application flow, system configuration, as well as tools for building and packaging the application for deployment.

Application Services

The framework includes a set of base services that provide important features and functionality to all applications. These services are described in detail below.

Messaging and Monitoring Service

The Messaging and Monitoring Service is a set of components that allow deployed applications to communicate with each other. By default, the service utilizes Apache ActiveMQ as an open-source JMS backbone to implement a guaranteed message delivery mechanism. The JMS standard provides fast and reliable event and update notifications, but the service can be configured to use message delivery implementations other than JMS.

The service uses small conversational messages to communicate between deployed components as it delegates complex or long running actions to asynchronous handlers that operate in the background. These delegate handlers can perform tasks such as file transfer requests via HTTP or utilize available bandwidth to efficiently retrieve digital assets from the proper repository.

The messaging sub-system defines two basic types of messaging:

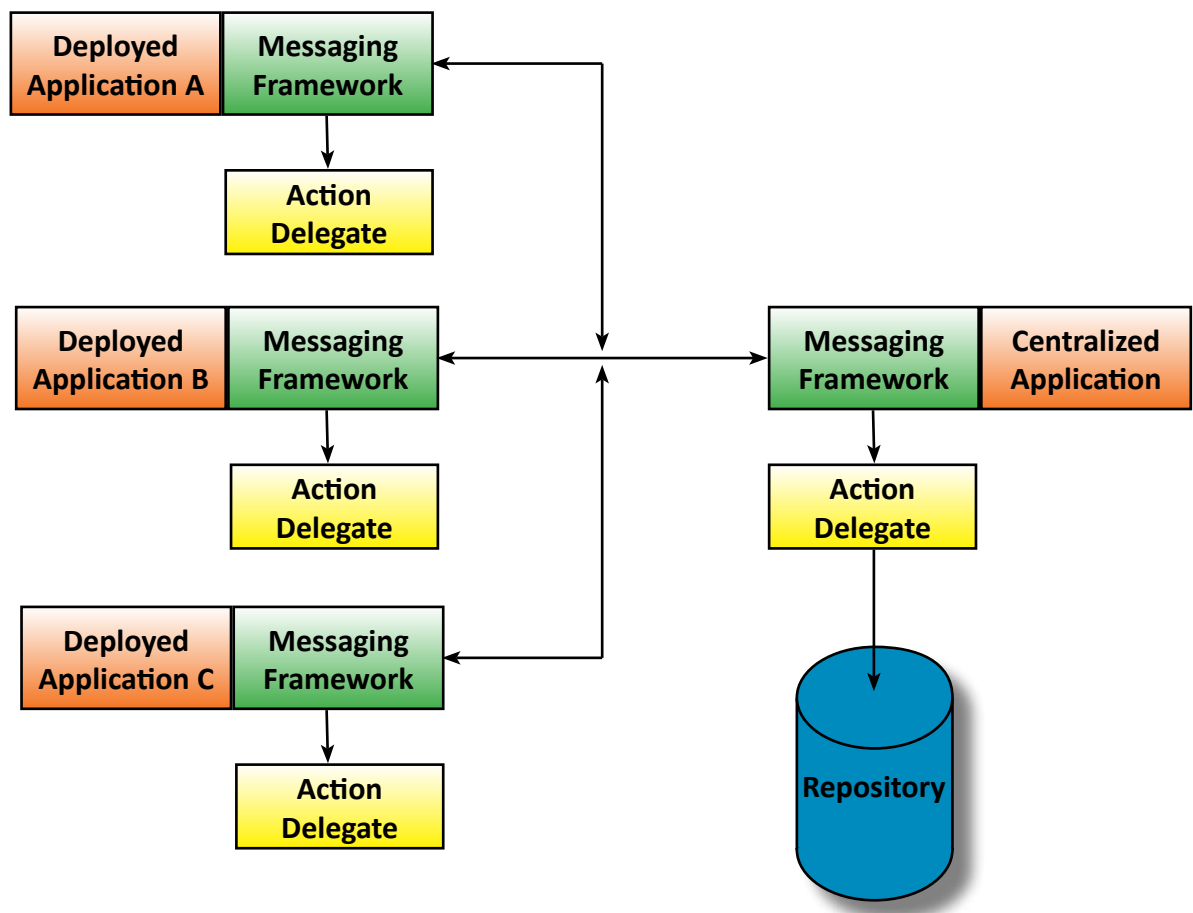
Control Messages

Control messages affect changes to configuration, or operation of remote applications or devices. Control messages provide the capability to remotely monitor and manage applications and devices deployed throughout the enterprise.

Control messages are distributed from a centralized application to endpoint entities. A control message could communicate a notification to endpoints that new or updated assets are available and should be retrieved from the central asset repository. A different control message could be used to instruct an application to increase the level of logging that is producing more easily diagnose issues.

Event Notification Messages

Event notifications are published from endpoint applications to a centralized message consuming application and represent change in state, or behavior of remote applications or devices.



Example Message Flow

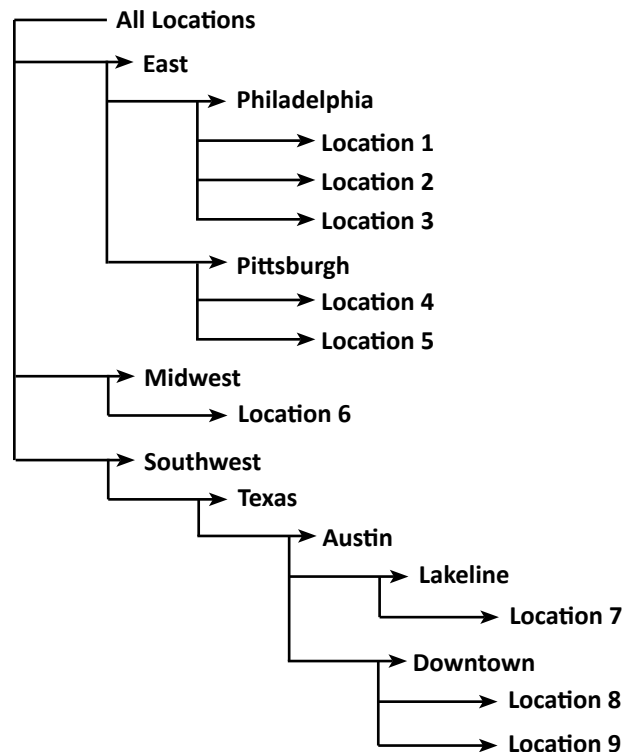
Deployed applications can be configured to publish event notifications to a centralized application. The centralized consumer application can be configured to handle the notifications in a number of ways. The event can be logged to an application log file, persisted in the dashboard application database, forwarded to an existing NMS, or published via email.

Location Hierarchy

The location hierarchy is a tree based mapping of all physical locations in a deployment. The Messaging Service utilizes this hierarchy to route messages to the appropriate destination(s). The hierarchy can be easily configured to represent complicated national, regional, area, and sub-area types of organizational structure or a simpler flat representation of locations.

Location Groups

Location groups can be configured to relate locations that share common characteristics, without regard to their placement in the location hierarchy. For example, a group could contain the ten largest locations in an organization, regardless of geographic proximity, or a group could represent all locations with a specific type of entity such as digital players, kiosks, or asset servers.



Example Location Hierarchy

Location Categories

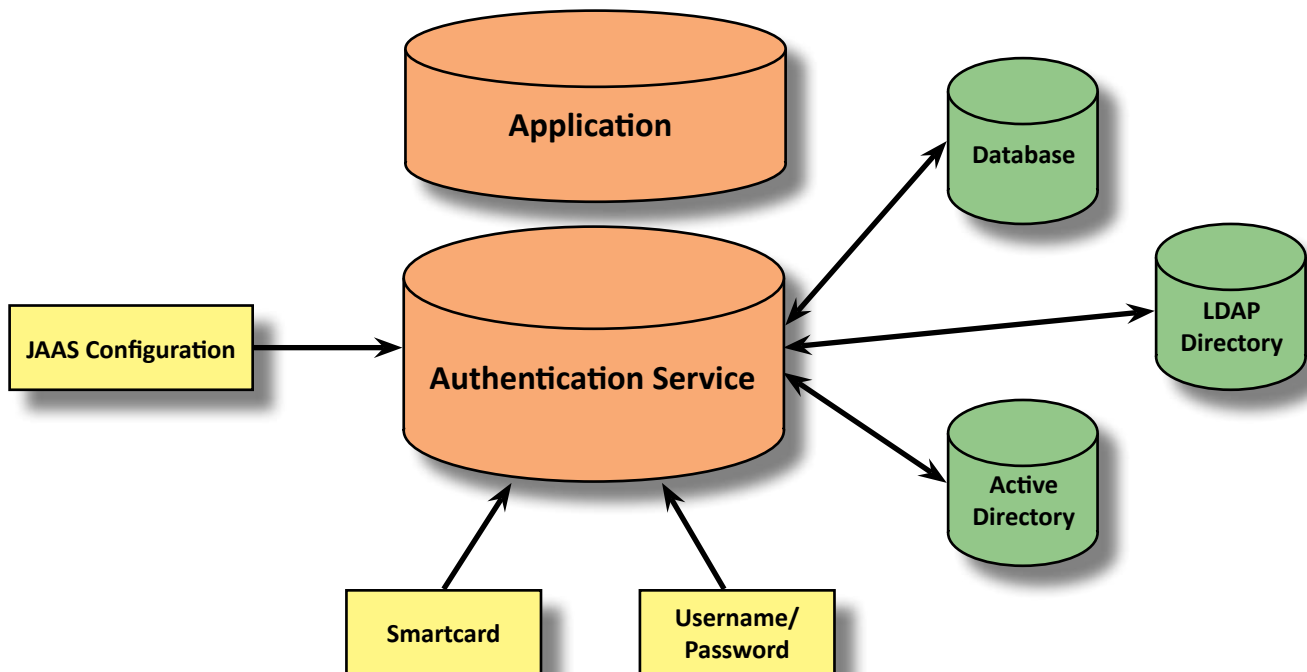
Each deployed entity can be assigned to a location category, which identifies some aspect of the device. The categories can represent things such as device type (digital player, kiosk, etc.) or any other arbitrary grouping such as front line registers, service desk kiosks, or regional asset servers. This provides the ability to only publish assets to the entities in a given category across the selected distribution list.

Data Distribution Service

The Data Distribution Service relies on the same location structures as the Messaging Service to distribute various data throughout a deployment. This data is typically larger and more specialized than the control messages, and it typically requires downloading and storage functionality beyond the simple messaging structures. Examples of this data include software updates, database seed data, or large configuration files.

Digital Asset Service

The Digital Asset Service is a specialized component of the Data Distribution Service designed to manage the transfer of large media files. While digital assets can be maintained in a central location, the real time streamed nature of these types of files often requires them to be stored in a more distributed fashion. This service allows the publication of digital assets to all endpoints for a given hierarchy base point, to all locations in a node in the location hierarchy, to a location group or even to a specific entity in the hierarchy.



Example Security Service

Asset Import

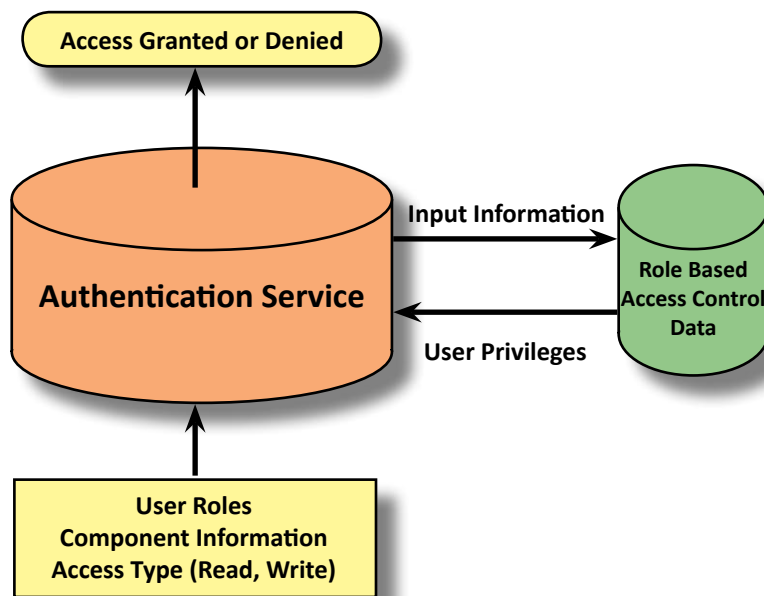
The Asset Import module allows existing content to be easily imported into the framework repository. An XML representation of the content is used to define the asset to be imported, and this representation is published to the deployed asset import module. The asset import module transforms the XML into the appropriate data attributes and then stores the content in a defined repository.

Software Deployment

The Data Distribution Service also provides deployment functionality for software updates and new applications. The application client automatically detects updates and downloads them from the backend server. Server applications are updated and reconfigured through the Messaging Service, and new applications or remote service interfaces are deployed in the same fashion. A version control component in the base framework ensures that the correct version of each application is running on the server and all of the clients.

Security Service

The Security Service provides user authentication as well as role based user authorization to application functionality using the standard Java Authentication and Authorization Service (JAAS) to reliably and securely determine who is currently navigating through the system. JAAS authentication is a pluggable technology that permits the framework to



Example User Authentication

remain independent from any underlying authentication technologies. In the base framework, user authentication is performed against information stored in a relational database, but in actual customer deployments it is more common to integrate the framework with an existing back-end system, using an enterprise LDAP directory or Windows Active Directory.

The user authorization component of the security service uses a Role Based Access Control (RBAC), a method of regulating access to different parts of the application based on the roles assigned to individual users within an enterprise. In this context, access is the ability of an individual user to perform a specific task, such as view, create, or modify certain information within an application. Roles are defined according to job competency, authority, or responsibility within the enterprise. The relationships between user roles, privileges and application components are stored in the application database and are accessed at runtime, enabling users to carry out a wide range of authorized tasks by dynamically regulating their actions according to their roles and constraints.

Encryption Service

The Security System also provides an encryption service that ensures user sensitive information such as passwords and customer data are properly encrypted. The base implementation uses the standard Secure Hash Algorithm (SHA -1) one way encryption algorithm to encrypt sensitive data before storage. Other encryption algorithms, such as those required for PCI compliance (Triple-DES 128-bit or AES 256-bit), can easily be plugged into the Encryption Service. External dedicated encryption hardware can also be integrated and used by the service.

Employee Service

The Employee Service provides a graphical front-end for entering and modifying employee information. Authorized users can search for employees and edit information such as user roles, address, phone numbers and passwords. Edit access can be restricted in a number of ways—single employees, all store employees, or all enterprise employees—based on the user's role.

The Employee Service also provides a number of password management features, including criteria-based validation of new passwords and expiration notification at login. The service can also serve as a front-end to any back-end systems that manage employee/user data.

Device Service

The Device Service provides an interface between framework applications and any installed peripheral devices such as scanners, printers, card readers, and pin pads. Typically, the Device Service is deployed on a client machine, providing a layer of abstraction between the client application and the low-level device drivers. This shields the applications from low level device interactions and allows device sharing among multiple applications. The Device Service supports various methods of connectivity to applications (sockets, JMS, RMI), can access devices remotely (devices connected to a remote system), and can run either stand alone or within an application JVM.

The Device Service can monitor device events and communicate to the appropriate framework applications. This enables device monitoring and management, diagnostics and self-checks, as well as device recovery upon failure. The Device Service defines a number of components to deliver its functionality. Context objects hold the desired state of a device. Multiple Contexts can be used for sharing a device between applications. Action objects communicate desired behavior for a local or remote device and update the state represented in the Context. Handler Objects control the device using whatever software mechanism required or provided. This typically would be JavaPOS, but could also be serial port communication or through a native driver (via JNI). The Handler will attempt to keep the device in a state specified by the active Context. The Handler supports device recovery and state restoration. Finally, Agent objects

apply Actions to Contexts. They also utilize Handlers to make a device reflect the Context. Agents queue requests to ensure that a single thread communicates with the device.

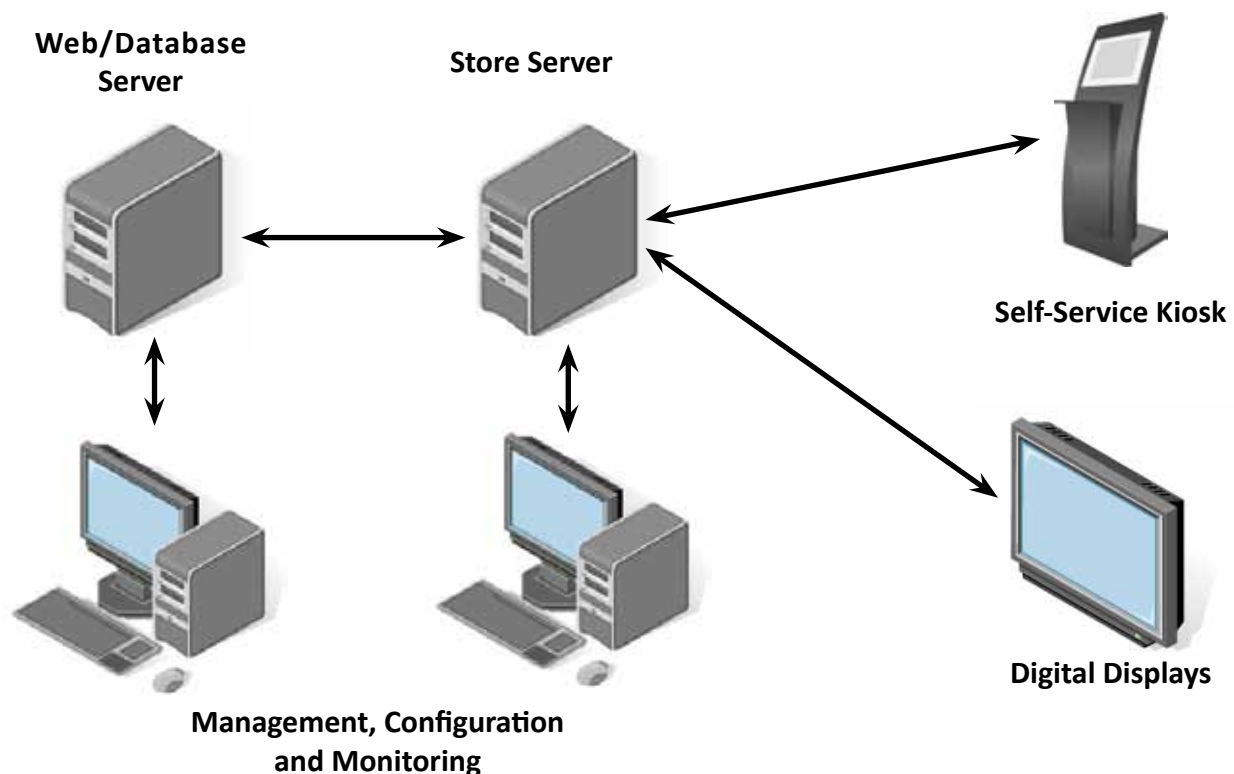
Logging Service

The logging service provides a mechanism for applications to record events occurring during runtime. The logging service records these events to a log file that can be used to debug the application. This service makes use of a standardized API known as log4j. Log4j is a popular logging tool, as it provides flexible control over logging and debugging requirements of a Java project. It is hierarchical in nature and provides runtime control over all aspects of logging without having to change the source code.

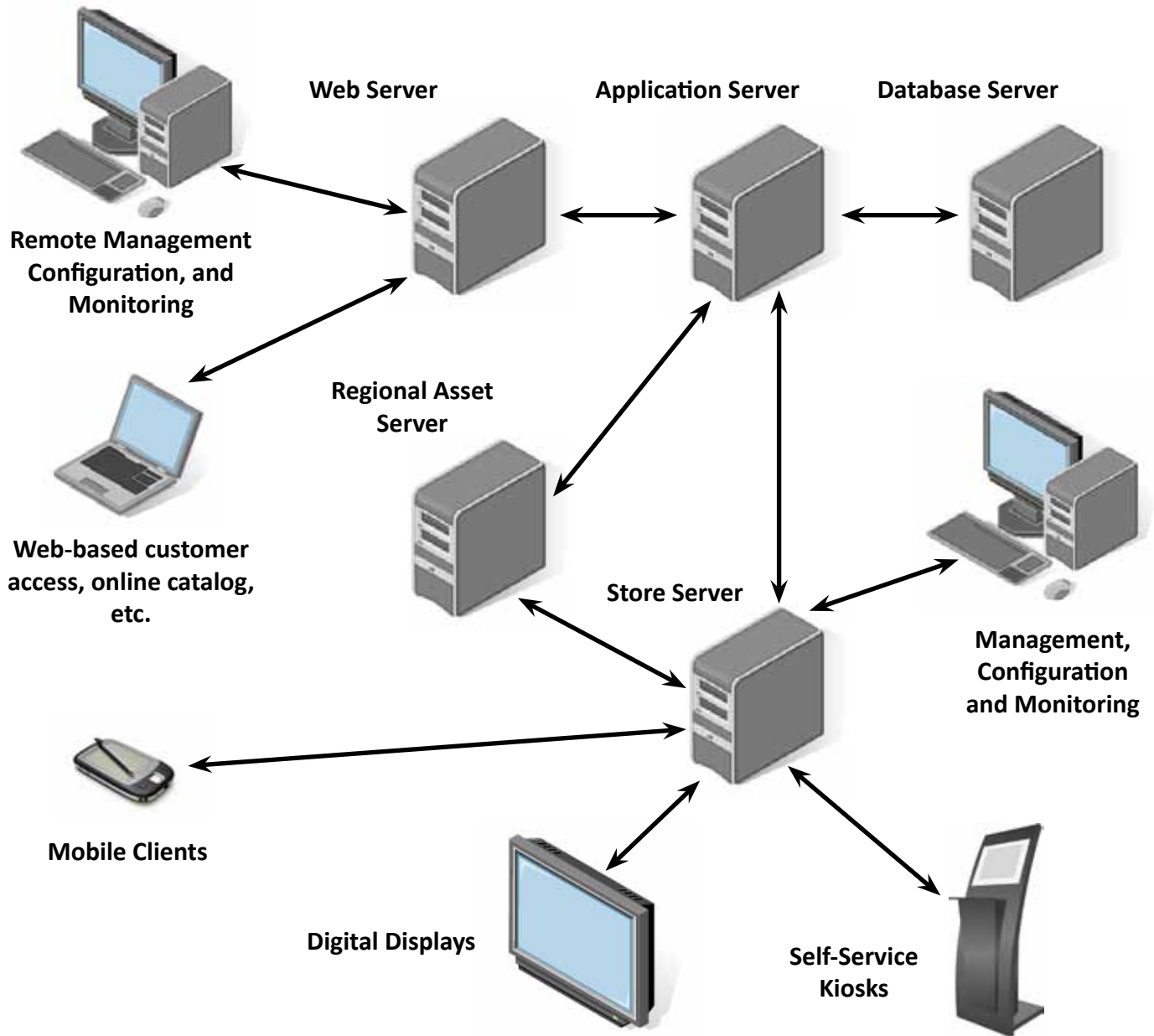
Log4j controls logging with three main properties: loggers, appenders, and layouts. A logger logs to an appender in a particular layout (style). These can be specified using an external configuration file and the configuration properties are loaded during application startup and can be changed at runtime. More than one appender can be attached to a logger. Utilizing the appenders construct within the log4j API, the application is able to log requests to multiple destinations (log file, database, e-mail,...). Currently, appenders exist for the console, files, GUI components, remote socket servers, JMS, NT Event Loggers, and remote UNIX Syslog daemons. It is also possible to log asynchronously.

System Topology

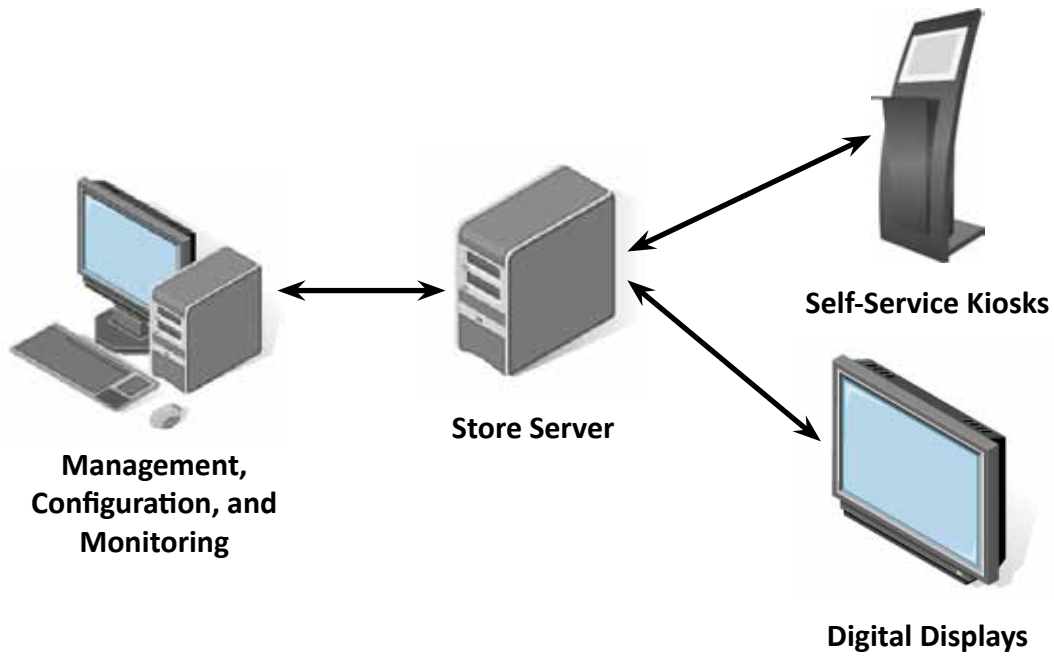
The component-based nature of the framework provides a wide variety of deployment options, including distributed internet applications, traditional client/server implementations, or complete stand-alone systems. The Application Controllers and other system components can be deployed in a single container or on separate tiers, and they can also be deployed into standard application servers when high use and robust scalability are required. Several deployment variations are depicted below.



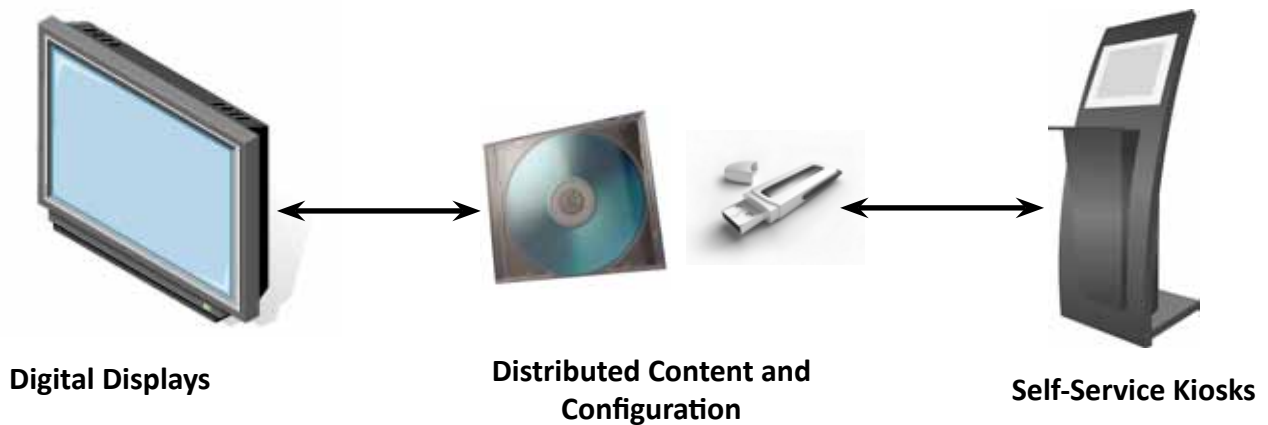
Basic Distributed Deployment



Complex Enterprise Deployment



Single Store Deployment



Standalone Deployment

Starmount Application Suite

Starmount has developed a suite of retail applications that utilize the Starmount Framework. Not only do these applications serve to demonstrate the power and flexibility of the framework, but they are also robust, ready-to-deploy applications that can immediately serve the needs of retail customers. A brief description of each application follows.

Starmount Dashboard

The Dashboard is the core of the application suite. It is a web-based application that provides monitoring, reporting, configuration, and content creation functionality for all of the other Starmount applications. It also incorporates builder tools that configure the other applications. In most configurations, the Dashboard is deployed with the server-side framework components, either at the enterprise or store level, or both.

Starmount Media Player

The Media Player is a content presentation client for digital signage applications. A typical player deployment would consist of a large digital monitor and a small factor PC such as the Apple Mac Mini. The player runs presentation files created in the Presentation Builder component of the Dashboard application. These presentations can contain digital images and video stored in almost any common format, as well as textual data. The builder tool provides screen layout, content sequencing, the addition of transitions and effects, and scheduling.

Starmount Kiosk

The Kiosk provides self-serve functionality for a variety of applications including order entry, product information, price checking, building or store directories, and any other application that wants to present highly-interactive content directly to customers. Along with the touch screen interface, the Kiosk can take input from and send output to common peripheral devices such as receipt printers and card readers. Multi-vendor configurations such as mall or airport food courts are easily supported, each with their own branded style and color scheme.

The Kiosk also has a Dashboard component for building and modifying hierarchal menus, importing item data, and monitoring orders. Kiosk order data conforms to the IXRetail schema standard for distributing transactional data to external systems, allowing integration with existing POS or back-end inventory systems. The Kiosk also integrates with multiple payment gateways for credit/debit authorization. It can also be configured to function as a web-based order taking application or as a mobile application deployed to smart phones, PDAs, and a myriad of web-enabled wireless devices.

Starmount Kitchen Monitor

The Kitchen Monitor is an order processing application that allows kitchen personnel to view orders, generated either in the Kiosk application or another existing system, process those orders, and update the status as each order is completed. It has a simple, intuitive interface that is typically deployed on an industrial grade touch screen system that can tolerate the extreme conditions found in a typical quick-service or fast-casual kitchen environment.

Starmount Digital Menu

The Digital Menu uses the same builder and menu data found in the Kiosk application, but it presents it as a dynamic menu solution on a large digital monitor. It also shares functionality with the Media Player allowing portions of the menu screen to be used for video or images, commercials, or daily specials.

Starmount Scheduler

The Scheduler provides appointment or work order scheduling, usable in markets such as auto repair, hair styling, dental, medical, restaurant reservation, and many others. It has a highly-configurable rule-based analysis component that searches for open time slots and validates entered appointments based on numerous criteria. Proposed appointments can be evaluated by task to perform, soonest available time slot, customer preferences, store location, available space (i.e., bays, chairs, tables), required equipment, available personnel, personnel qualifications or certifications.

Starmount Directory

The Directory is a kiosk application that is deployed, for example, in the lobby of an office building to allow visitors and customers to interactively locate and view business details of the building's occupants or tenants. There is a wide range of deployment possibilities including medical facilities, campuses, outdoor malls, trade shows or airports.

Conclusion

Starmount is the leading provider of end-to-end solutions for the integrated digital signage and retail managed services industry providing Oracle Retail software solutions and services for digital signage, interactive kiosks, and retail point-of-service products. Built on our next generation Application Framework, Starmount provides quality integrated products and services, enabling our customers to realize value from their digital marketing solutions. Taking a partnership approach, we help extend the lifetime value of our customers' solutions by focusing on understanding their exact business process as well as technology needs and providing a complete integration of their systems.

With a proven track record of professional software consulting services for a myriad of industry applications -- including the retail industry, Starmount has extensive experience implementing/integrating your next-generation, centralized, or in-store Oracle Retail software solutions; enhancing your existing solutions to meet changing business needs; or performing as an outsourcing partner to provide maintenance and support for your legacy solutions.

For more information on Starmount and Starmount solutions go to www.starmountsystems.com.