

Marlin Architecture Overview

THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY MAKE NO REPRESENTATION OR WARRANTY, EXPRESS OR IMPLIED, CONCERNING THE COMPLETENESS, ACCURACY, OR APPLICABILITY OF ANY INFORMATION CONTAINED IN THIS DOCUMENT. THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY DISCLAIM ALL LIABILITY OF ANY KIND WHATSOEVER, EXPRESS OR IMPLIED, ARISING OR RESULTING FROM THE RELIANCE OR USE BY ANY PARTY OF THIS DOCUMENT. THIS DOCUMENT IS PROVIDED TO YOU "AS IS".

THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY MAKE NO REPRESENTATIONS CONCERNING THE APPLICABILITY OF ANY PATENT, COPYRIGHT OR OTHER PROPRIETARY RIGHT OF A THIRD PARTY TO THE INFORMATION CONTAINED IN THIS DOCUMENT OR ITS USE. THE RECEIPT OR ANY USE OF THIS DOCUMENT OR ITS CONTENTS DOES NOT IN ANY WAY CREATE BY IMPLICATION, ESTOPPEL OR OTHERWISE, ANY LICENSE OR RIGHT TO ANY PATENT, COPYRIGHT, TRADEMARK OR TRADE SECRET OF THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY, WHICH ARE OR MAY BE ASSOCIATED WITH THE IDEAS, TECHNIQUES, CONCEPTS OR EXPRESSIONS CONTAINED HEREIN.

Table of Contents

1	Introduction.....	3
1.1	Scope	3
2	Terminology and Conventions.....	3
2.1	Classification	3
2.2	Abbreviations	3
2.3	Terms and Definitions	3
2.4	Conventions	5
3	Marlin Specifications.....	6
3.1	Incorporated Technologies.....	7
3.2	Octopus DRM.....	7
3.3	NEMO Framework	7
4	Marlin Entities and Objects.....	7
4.1	Actors	8
4.1.1	Users.....	8
4.1.2	Service Providers.....	8
4.1.3	Content Providers	8
4.2	Marlin Objects	8
4.2.1	Content Objects and License Objects	8
4.2.2	Node and Link Objects	9
4.2.3	A Note about “Nodes”	10
4.3	Targeting and Binding.....	11
4.4	Functional Components	11
4.4.1	Marlin Client.....	11
4.4.2	Domain Manager	11
4.4.3	Registration Service.....	12
4.4.4	License Service.....	12
5	Content Acquisition Use Cases.....	13
5.1	Simple Purchase or Rental	13
5.2	Subscription	14
5.3	Subscription Renewal	16
6	Domains	17
6.1	User-based Domains	17
6.2	Marlin Common Domain Model	19
7	Portability Use Cases	20
7.1	Simple Portability Example	20
7.2	Multi-user / Multiple Domain Portability Example.....	20
7.3	Sharing and Superdistribution.....	21
7.4	Superdistribution	22
7.5	Sharing.....	23
8	Alternate Delivery Scenarios.....	24
8.1	Mobile Delivery using OMArlin	24
9	Summary	26
10	References	26

Marlin Architecture Overview

1 Introduction

The Marlin architecture specifies technologies for building copy protection and digital rights management (DRM) into consumer devices and services in a manner that is friendly to end users. Marlin technology allows users to acquire content through multiple distribution channels and to access it on any device that is part of their home domain. In order to accomplish this, Marlin defines both client capabilities and a service architecture so that the capabilities of consumer electronics devices can be powerfully enhanced by services provided over both local and wide-area networks.

1.1 Scope

The purpose of this document is to provide an introductory-level overview to the Marlin DRM architecture and its application to common implementation use cases. This document is best suited to technical implementers or developers who are beginning to read the Marlin specifications and who seek a high-level context with which to understand the detailed specifications. Other readers who may benefit from this document are technically-oriented business planners or project managers who wish to gain a deeper technical understanding of how Marlin can be used to execute common business scenarios.

2 Terminology and Conventions

2.1 Classification

This document is classified as INFORMATIVE, and therefore is not intended to provide testable Marlin requirements. If any differences or discrepancies exist between the information contained in this document and that included in the normative Marlin specifications, the Marlin specifications shall take precedence in all instances.

2.2 Abbreviations

DRM	Digital Rights Management
HTTP	Hypertext Transfer Protocol
MCD	Marlin Common Domain
MCS	Marlin Core System
NEMO	Networked Environment for Media Orchestration
OMA	Open Mobile Alliance
PC	Personal Computer
PVR	Portable Video Recorder
TBD	To Be Defined
URL	Uniform Resource Locator
XML	Extensible Markup Language

2.3 Terms and Definitions

The following table lists a number of frequently used terms and their definitions as used in this document. When defined terms appear in the definitions of other terms, they appear capitalized.

Term	Definition
Action Token	An XML-encoded document that directs a Marlin Client to perform a sequence of actions, such as obtaining a Node from a Registration Service or acquiring a License from a License Service. An Action Token specifies how to get service location information, and includes information necessary to make Marlin protocol requests for communicating with the specified services. An Action Token is issued by a Web Store or an e-commerce system, for example, after a User requests purchase of a particular piece of Content.
Binding a License to a Node	Making Content cryptographically accessible for a system entity (such as a User) by encrypting the Content Key with either the public key or the symmetric secret key associated with the Node representing the entity.
Content (Marlin Content)	Digital media (such as a music or video file) encrypted with a Content Key and packaged into a Marlin file format.
Content Key	A symmetric cryptographic key used to encrypt and decrypt an instance of Content. The encrypted Content Key is stored in the License corresponding to the Content.
ContentKey object	An Octopus object that is part of a License and that includes an encrypted Content Key.
Deregistration	A protocol used to invalidate a Link, e.g., a Link between a Personality Node (representing a Device) and a User Node.
Device	A self-contained hardware component or software application capable of hosting a Marlin Node with specific Marlin Roles.
Domain	A collection of Devices.
Domain Manager	A Role for managing the Registration and Deregistration of Devices with a Domain, according to a Domain Policy.
Domain Policy	The set of rules and conditions for forming Domains, for registering and deregistering Devices to and from Domains, and for managing relationships between a Domain and its member Devices (Personality Nodes).
License (License Bundle)	A set of Octopus objects that govern the use of Content and convey the conditions necessary for allowing access to the Content Key used to encrypt the Content.
License Owner	A User who has acquired the rights to use the Content associated with a License.
License Service	A Marlin service handling the creation of Licenses.
Link (Link Object)	An Octopus object that expresses a relationship between two Octopus Nodes.
Marlin Client (Marlin DRM Client)	A Marlin-compliant Device or application that is able to communicate with Marlin services, e.g., to acquire and evaluate a License that governs access to an instance of Content. A Marlin Client is represented by a Personality Node.
Membership	A relationship between a Device and a Domain. A Device becomes a member of a Domain by registering with it.
Node (Octopus Node)	In the context of this document, a Node refers to an "Octopus Node." An Octopus Node is an object representing a Marlin DRM System entity . For example, a Personality Node represents a Device, a User Node represents a User, and a Subscription Node represents a Subscription. An Octopus Node includes a public/private key pair or a symmetric key that can be used to bind Licenses to the Node.

Term	Definition
Octopus	A general-purpose DRM architecture.
Principal	A system entity whose identity can be authenticated.
Proximity	A measure of distance between Devices.
Registration	A protocol by which a Device establishes a membership relationship with a Domain. Membership in a Domain is represented by a Link from the Personality Node representing the Device to the Domain Node (which, currently, in Marlin Broadband, is a User Node).
Registration Service	A Marlin Service responsible for issuing Octopus Nodes and Links, and for disassociating the relationship between two Nodes.
Role	A combination of client or service functions supported by a Device or by a services implementation.
Service Provider	Entity that provides services, such as a License Service and a Registration Service. Also a generic term used to describe the entity or organization responsible for selling or distribution of digital media Content and associated Licenses.
Targeting a License to a Node	A process by which an application running on a Device is allowed to access Content only if the associated Marlin Client contains a valid set of Links to form a path from the Personality Node (the Node representing the accessing Device) to the Node(s) to which the License is targeted. A License may be targeted, for example, to a User Node or a Subscription Node.
User	An individual or a group of people (such as a family) represented by a User Node.
Web Store or e-commerce system	An entity that is the front end for all the operations that interact with the User. As a result of such an interaction, a Marlin Client can be provisioned with an Action Token that triggers the Marlin Client to contact a Marlin service. Note that this entity is only used for illustration, as the same tokens could be delivered via another mechanism without affecting the specifications.

2.4 Conventions

To be consistent with the objective as an overview, the UML sequence diagrams contained in this document incorporate a necessary degree of abstraction. In particular, the term “Marlin Client” is used in the sequence diagrams to refer to the combination of *both* the Marlin DRM Client and the host application responsible for playback, communications, and user interaction. Where possible, a distinction is made between steps that are included in the Marlin specifications (and are performed by the Marlin DRM Client) and steps that are out of scope of the Marlin specifications. As a convention, the sequence elements that are *not* part of the Marlin specifications are labeled in *italics*, as shown in the sample diagram below.

Scenario or Use Case Title

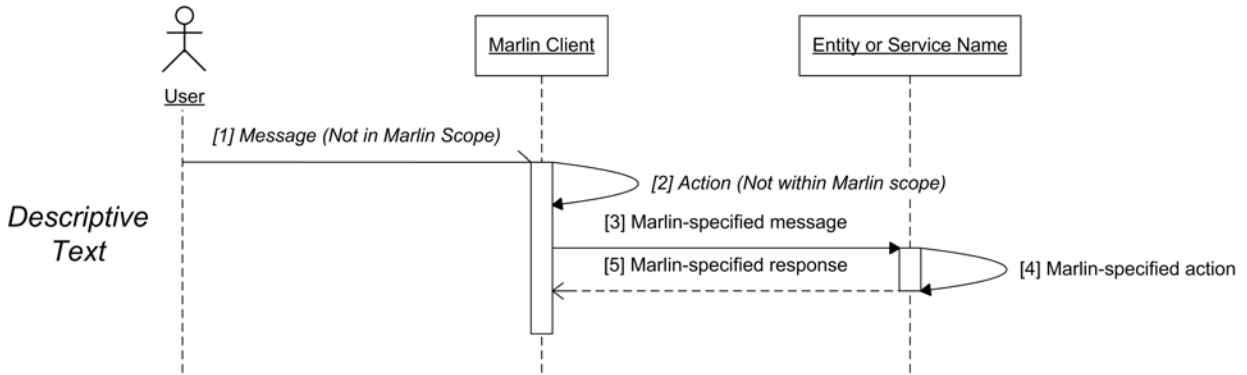


Figure 1 Sequence diagram conventions (example)

3 Marlin Specifications

The Marlin architecture consists of a set of specifications for developing a digital media distribution system. There are currently several normative Marlin specifications, including one “Core” system specification and separate delivery system specifications. The core system specification defines elements that must be present in all Marlin implementations, whereas each delivery system specification specifies additional requirements for a particular type of delivery channel, such as broadband or mobile. Other specifications include the Marlin Common Domain Specification and the Marlin IPTV End-point Service Specification. The following table lists some of the Marlin specifications and their objectives:

Specification	Purpose
Marlin Core System Specification	Defines the basic components and protocols that must be present in <i>all</i> Marlin implementations.
Marlin Broadband Delivery System Specification	Defines additional capabilities that are unique to clients that can take advantage of an interactive IP-based communications channel. Among other items, includes protocols for obtaining licenses and reporting usage data for subscription-based services.
OMArlin Specification	Specifies how to enable interoperable download, streaming, sharing and consumption of content between the OMA and Marlin DRM systems.
Marlin IPTV End-point Services Specification	Defines support for “end point” devices, such as TVs with IP capabilities.
Marlin Common Domain Specification	Defines a common policy for implementing domains (groups of devices enabling content sharing) in Marlin implementations. Establishes a common behavior for management of domains to promote end-user consistency.

3.1 Incorporated Technologies

To facilitate adoption and implementation, the Marlin DRM system uses industry standards wherever practicable. As a result, the Marlin specifications incorporate many industry standard technologies, including security standards from IETF, W3C, ISO, OASIS, and others. To support flexible consumer usage scenarios and business models, most Marlin implementations also incorporate two important platform technologies known as the Octopus DRM architecture and the NEMO framework. These frameworks were developed by Intertrust technologies and presented to Marlin as a solution for building a user-centric DRM system. These platform technologies are briefly defined in the sections that follow.

3.2 Octopus DRM

Octopus is a general-purpose DRM architecture that can be applied to a variety of applications ranging from enterprise document control to medical record privacy protection, consumer media copyright protection, or any other system requiring distributed governance and control of information. At the center of an Octopus system is an Octopus DRM Engine—a small, lightweight component responsible for evaluating controls and determining whether access should be granted in a given set of conditions. By itself, Octopus is entirely agnostic regarding content formats, hardware/software platforms, and business semantics. That is, the Octopus DRM Engine responsible for governing access to content is entirely unaware of the type of content it protects, and is not bound by a particular set of semantics (e.g., not limited by predefined meanings or hard-coded “rules languages”). The Marlin specifications are an application of this generic DRM architecture to consumer audio and video distribution.

One of the distinguishing features of an Octopus-based system (and an attribute that makes it particularly well suited for Marlin) is its ability to separate the *protection* of content from the *governance* of that content. This allows, among other benefits, the ability to issue rights to access content separately from information that governs where or when it can be used. This can be used to achieve great end-user flexibility; for example, a service provider can issue rights to a customer to use content, but can allow that customer to independently manage which devices he or she would like to use that content on at any particular moment.

Section 4 of this document discusses in more detail how the various components of Octopus are used in Marlin. For a more generic description of Octopus and its features, also refer to the “Role of Octopus in Marlin” [ROLE_OCT] whitepaper listed in the References section (§10).

3.3 NEMO Framework

The NEMO (Networked Environment for Media Orchestration) framework provides the trusted “plumbing” between the various functional components in a Marlin system. NEMO combines SOAP web services with SAML authorizations to provide end-to-end message integrity and confidentiality protection, entity authentication, and role-based service authorization. Through the use of the NEMO framework, Marlin components can leverage a consistent mechanism to ensure that messages are delivered with appropriate protection and are exchanged between entities that are properly authenticated and authorized.

For the purposes of this overview document, we assume without mention that the “NEMO layer” performs all required message security operations between Marlin Clients and services. This allows the discussion to focus on higher-level processes and protocols instead of transmission details. To gain a better understanding of the application of NEMO to Marlin, refer to the Marlin normative specifications or the “Role of NEMO in Marlin” whitepaper [ROLE_NEMO] listed in the References Section (§10).

4 Marlin Entities and Objects

This section describes many of the high-level components and objects that comprise a Marlin system. The goal of this section is to introduce general Marlin concepts, actors, and functional components that serve as “building blocks” for the use cases that follow in sections 5-8.

4.1 Actors

Actors are the individuals or entities that are responsible for performing the processes described in this document. In the use cases that follow, the primary actors are users, service providers, and in some instances, content providers.

4.1.1 Users

In this context, a user is an individual or a group of people (such as a family) who is represented by a User Node, interacts with service providers to acquire Licenses for digital content, and interacts with Marlin Clients to access or manage use of that content. In most instances, users are also “License Owners”, that is, they have acquired the required rights to use the content under given circumstances. In some use cases however, such as in the “sharing” or “superdistribution” use cases, the user does not have a License to use the content and must interact with a service provider or a valid License Owner to gain access to the content.

4.1.2 Service Providers

A service provider is an entity that provides services, such as a License Service and a Registration Service. “Service provider” is also the generic term used to describe an entity or organization that is responsible for selling or distributing digital media content and associated Licenses. Service providers are not limited to any particular type of business model for licensing or distributing content. They can choose to support “a la carte” individual content Licenses, rentals, subscription-based services, or a hybrid of these.

4.1.3 Content Providers

In most of the use cases that follow, it is assumed that the service provider also serves the role of a content provider and aggregates content from content owners and distributes it to users. However, Marlin is flexible and allows for a variety of mechanisms for content delivery. For example, content may simply be downloaded from a web service (such as in the content acquisition use cases in Section 5), or may be delivered in peer- to-peer fashion (such as in the superdistribution or sharing use cases in Section 7).

4.2 Marlin Objects

Marlin relies upon the Octopus architecture and therefore employs several types of Octopus objects to deliver protected content, Licenses, and other data required to govern the use of content on users’ devices. Although there are many Octopus objects involved in Marlin, those that are pertinent to this overview are Content and License objects, and Node and Link objects. Content and License objects are used to protect content and specify the controls that govern use of the content, whereas Node and Link objects are used to express ownership or membership—such as which devices a user owns, or which subscriptions a user subscribes to. A more detailed description of these objects and their purpose is included in the following sections.

4.2.1 Content Objects and License Objects

In an Octopus-based system such as Marlin, the term *Content object* refers to the encrypted content that the system governs. It consists of various header data and a block containing the encrypted media file. This object can be delivered in a variety of ways and can be transported within industry standard, open file formats. A symmetric key (referred to as the *content key*) is used to encrypt the media in the Content object, and this key is itself encrypted and delivered separately in a *License Bundle*, usually referred to simply as a (*Marlin*) *License*. Typically, a user will receive a License Bundle from a service provider after completing a transaction to rent, subscribe to, or purchase the content. This separation allows content and associated rights to be distributed independently and at different points in time, enabling superdistribution, “push,” and broadcast distribution models (among others).

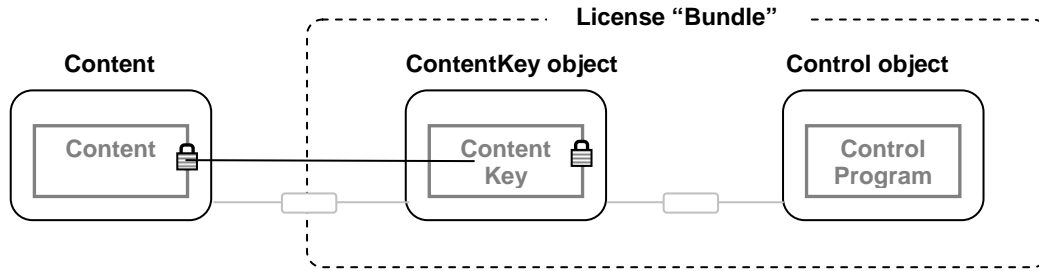


Figure 2 Content and License objects

The License Bundle consists of several objects, some of which are illustrated in Figure 2. The primary components (for purposes of this discussion) are the Control object and the ContentKey object. The Control object is used to specify the conditions under which the content key can be used to decrypt the content, and therefore governs access to the protected content. The Control object expresses the conditions or rules that control access to the content using what is known as a *control program*. A control program expresses rights programmatically, without the need for a predefined rights expression language. This allows service providers flexibility in defining rules to govern content by allowing them to combine atomic-level expressions in various ways to satisfy existing business models and adapt to new business cases that might arise over time.

The other primary object in the License Bundle is the ContentKey object. The purpose of this object is to distribute the key that was used to protect the content. To ensure confidentiality, the content key must be carried in encrypted form within the ContentKey object. The key that is used to encrypt the content key corresponds to the principal that the content (technically, the content License) is “bound” to. In Marlin, this is typically the principal associated with the user who has purchased, rented, or subscribed to the content. The concepts of “binding” and “targeting” are described in further detail in section 4.3, following a brief description of Nodes and Links, below.

4.2.2 Node and Link Objects

One of the unique elements of an Octopus-based DRM system such as Marlin is the use of Node and Link objects to express relationships between the principals within the system (e.g., users, devices, and subscriptions). In most traditional DRM systems, licenses are directly bound to the device that is used to obtain the rights. In Marlin, a License is typically bound to a user (more precisely, to an Octopus Node representing the user), and relationships between users and devices, or users and subscriptions, are maintained separately, through a system of *Nodes* and *Links*. This ability to separate the governance rules (e.g., the License) from the frequently-changing environment of user devices and subscription status allows for a more flexible user experience.

In Marlin, Octopus Nodes are defined that represent logical entities or “principals” within the system. For example, Personality Nodes represent devices, User Nodes represent users, and Subscription Nodes represent subscriptions.

Links are self-protected objects that represent relationships between these principals, forming a directed graph. An example of such a graph is shown in Figure 3.

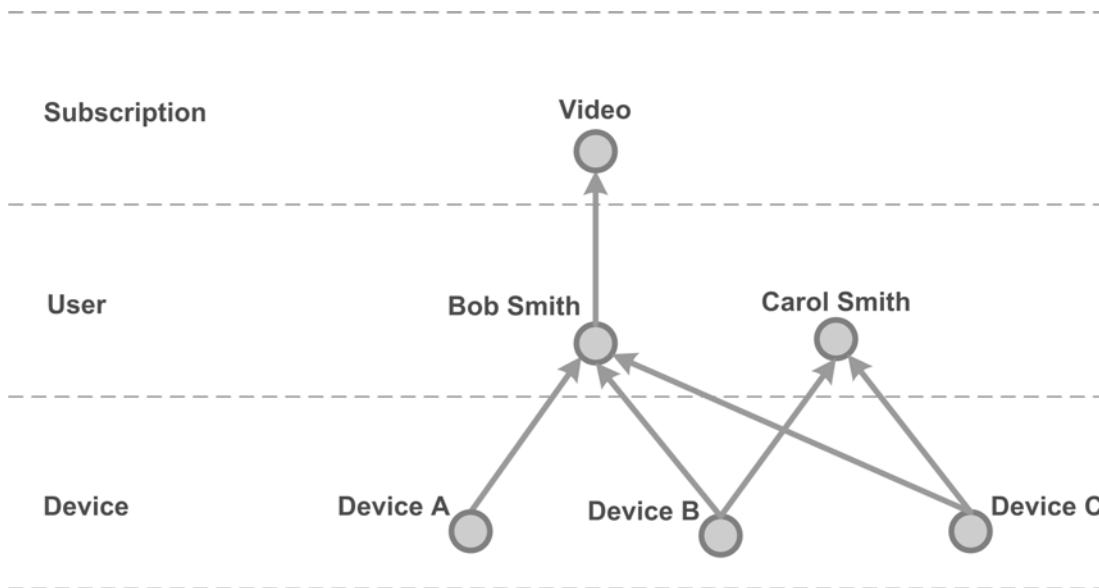


Figure 3 Node and Link directed graph

In Figure 3, Nodes are illustrated as circles, and Links between Nodes are shown as arrows. The actual semantic of a Link is a function of the business model the Link issuer is enabling. For example, for many issuers, the presence of a Link (or arrow) between two Nodes may indicate that there is a relationship between the two Nodes, and this relationship may be described as representing that a particular Node “belongs to” or “is a member of” the Node that the arrow is pointing to. Using this interpretation, the set of Nodes and Links in Figure 3 indicates that Device A “belongs to” user “Bob Smith”; and that Bob Smith “subscribes to” the “Video” subscription service. Similarly, Devices B and C “belong to” both “Bob Smith” and “Carol Smith”.

This system of Nodes and Links is integral to managing where, how, and when content can be used in a Marlin system. When a License is created, the control program within it will typically include a requirement that a certain Node, such as a User Node (e.g., one representing “Bob Smith”) be *reachable* by the device attempting to access the content. Being *reachable* means that the Marlin Client running on the device in question contains a valid set of Links from the Personality Node representing the device to the specified Node. This can be a direct Link from the Personality Node to the specified Node (such as from Device A to user “Bob Smith” in Figure 3) or a series of Links (such as from Device B to “Bob Smith” to “Video”). This process of indicating in the License’s control program that a particular Node must be *reachable* is referred to as *targeting* to that Node, and is described more fully in Section 4.3.

As another example of what the figure represents, note that Device A will be allowed to play content targeted to Bob Smith and/or the “Video” subscription, but it will not be able to play content targeted to Carol Smith, since there is no Link between Device A and Carol Smith.

4.2.3 A Note about “Nodes”

The term *Node* is used in two separate contexts within Marlin and therefore warrants further discussion to avoid confusion. In Marlin, two types of Nodes are discussed: Octopus Nodes and NEMO Nodes. In this document, all references to Nodes (e.g., User Node, Personality Node, etc.) refer to Octopus Nodes. An *Octopus Node* represents a principal (e.g., a device, a user, or a subscription) to which relationships can be described. Links between Nodes indicate that a relationship exists, such as a device belongs to a user, and a user belongs to a subscription. When evaluating a License on a particular device, an Octopus engine will check to see if the required relationships exist between the device’s Personality Node and the target Node(s) referenced in the License.

The other type of “node” discussed in Marlin is a *NEMO Node*. NEMO Nodes are active, in contrast to Octopus Nodes, which are passive. A NEMO Node is analogous to the trusted “host” of a particular functional component. This host acts as a trusted entity with which to send and receive messages. For example, a License Service would use its NEMO Node to send and receive messages to another NEMO Node representing a Marlin Client.

Note that both of these definitions are entirely separate from the notion of a node as “an addressable location on a network.” for example, a device that functions as both a Marlin Client and a Domain Manager would have a NEMO Node for each role, yet still have only one address on the network.

4.3 Targeting and Binding

Targeting and binding represent two different, yet closely related processes. In Marlin, *binding* is primarily a cryptographic process, concerned with protecting a content key, a key that was used to encrypt content. When a License is *bound* to a User Node, it means that the content key is encrypted with the public key (or secret key) associated with the User Node. Therefore, only devices that have access to the private (or secret) key of that User Node will have the necessary key to decrypt the content. Note, however, that simply having the correct key only indicates that the device has the *capability* to decrypt the content—if it is permitted to do so.

Whether or not a device is *permitted* to access the content is determined by the control program in the License, and specifically, how it is *targeted*. Within the control program, the service provider will typically specify that a particular Node or Nodes—referred to as the *target Node(s)*—must be *reachable* via a Link or chain of Links from the device attempting to play the content to the target Node(s). In most instances, the control program will specify that a particular User Node must be reachable by the consuming device. In some instances, however, a service provider may wish to indicate that other Nodes must be reachable as well. An example of this would be the case of a subscription, where the License would most likely be targeted to the subscription *and* the user. In other words, the device must not only possess a valid Link (or path of Links) to the User Node, but must also possess a path of Links to a particular Subscription Node, including a valid (e.g., unexpired) Link from a User Node to that Subscription Node.

4.4 Functional Components

Marlin defines several modular components that are designed to serve a particular purpose or role within the system. Several of these components (those that are referenced in the use cases that follow) are briefly described below.

4.4.1 Marlin Client

A Marlin DRM Client is the functional component responsible for requesting Licenses and Links, and for controlling access to protected content. A Marlin Client may be implemented in a hardware device (such as a portable media player) or as a client application (such as a PC software media player application). When the host device (or player application) requests access to content, the Marlin Client executes the control program in the License and checks for the presence of any required Links. Then, if permitted by the License, the Marlin Client allows the content key to be decrypted and used to access the protected content.

4.4.2 Domain Manager

A Domain Manager serves the function of creating a domain and managing membership of devices in the domain. The Core specification allows for a Domain Manager to either be operated by a service provider at a remote location accessible via the internet, or as a service implemented on a Marlin device in the user’s local network. In Marlin Broadband, for example, a Domain Manager is operated by the Registration Service. The rules that a Domain Manager must use to administer domains are defined via a Domain Policy.

4.4.3 Registration Service

A Registration Service is typically operated by a service provider, and serves the roles of issuing Nodes, issuing Links, and invalidating Links. During the purchase process (or a subscription renewal process), the Marlin Client receives a trigger to contact a Registration Server to request the appropriate Nodes and/or Links to support the transaction. For example, when purchasing content a Marlin Client would be instructed to request a User Node corresponding to the user, and a Link from the Client (that is, from the Personality Node representing the device the Client runs on) to that User Node. Alternatively, in a subscription renewal, the Marlin Client might be automatically triggered to request a new User Node-Subscription Node Link from the Registration Server (e.g., one with an extended expiration date).

4.4.4 License Service

A service provider also operates a License Service that is responsible for issuing Licenses to Marlin Clients. After a service provider's e-commerce system processes a transaction for a purchase, the e-commerce system triggers the Marlin Client to contact a License Service with a request for a License. The License Service generates the necessary License objects and responds with a License Bundle that the Marlin Client will use to govern access to the content.

Note: Although the License Service and the Registration Service are distinct functional components, for purposes of simplicity the sequence diagrams that appear later in this document group both of these components under a common heading labeled "Service Provider". Even though the diagrams depict these components together, the description that accompanies each diagram clearly indicates which functional component is responsible for addressing each request.

Additional functional components are also defined in Marlin, but are not explicitly referenced in the use cases that follow. These other components serve functions such as ensuring that Clients have the most current versions of security metadata, trusted time values, etc. For more information about these other components, please refer to the detailed Marlin Specifications listed in the References Section (§10).

5 Content Acquisition Use Cases

This section describes the typical processes that a user undertakes to purchase content and the rights to access and use that content. For illustration, we describe two use cases that reflect currently popular business models. First, we describe a “simple content purchase” use case, where a user selects and purchases the rights to access a particular content item. Next, we describe a “subscription” use case, where a user purchases time-limited rights to access a large catalog of content. Marlin is extremely flexible and can support a wide variety of business models and implementation variations in addition to the two described here. However, for the purpose of brevity, they are not included in this document.

In the examples that follow, we assume that the Marlin Client can be either hardware-based (such as a dedicated device) or software-based (such as a user-installed PC application). In the former case, we assume that the Client has already received a unique Personality Node at the time of manufacture. In the case of a software-based client, we assume that the application has undergone an additional provisioning step in which the Client has requested and received a unique Personality Node.

5.1 Simple Purchase or Rental

The “simple purchase” use case describes a scenario where a user purchases rights to play a single content item that is (or has been) downloaded to their device. This use case is also commonly referred to as an “a la carte” business model, since the user selects individual content items to purchase and acquire Licenses. The case described below applies as well to a rental as it does to a purchase, the only difference being that the rental License would specify an expiration date in its control program, whereas the License for a purchase would allow for unlimited use.

Simple Purchase or Rental

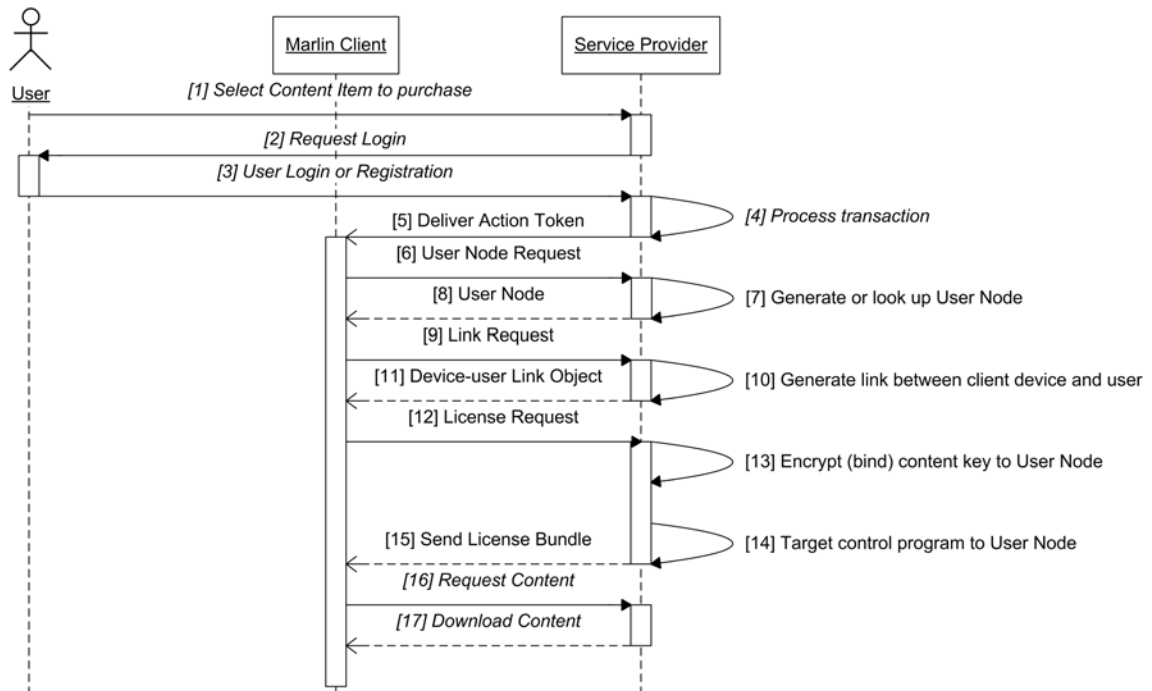


Figure 4 Simple purchase use case

Figure 4 illustrates a typical Marlin implementation for the simple purchase (or rental) use case. Several steps (1-5, 16 and 17) are out of scope of Marlin, but represent a typical e-commerce buying experience and are included here for context. In this example, the process begins when a user browses an e-

commerce portal and [1] selects content item(s) to purchase. If the user has not already done so, the service provider [2] requests the user to sign in to their account (or create a new account if they are a first-time user). The user responds in [3] by supplying appropriate user identification and payment information to complete the purchase. The service provider's e-commerce system then processes the transaction, and, when satisfied that the payment has been arranged, proceeds by issuing an Action Token [5] to the user's Marlin Client. This is essentially the beginning of the Marlin process.

The Action Token directs the Marlin Client to perform a sequence of activities, one of which is to contact the service provider's Registration Service to [6] request a copy of the customer's User Node. If this is a new customer, the Registration Service will [7] generate a new User Node for the customer; otherwise, it retrieves (or regenerates) a User Node from the customer's record. After the User Node is delivered [8], the Client sends a request to the service provider's Registration Service for a Link object. The Registration Service generates [10], and responds [11] with a Link object that associates the User Node with the Client's Personality Node. In addition to the Link and Node objects, the client also requires a License to use the content. The client is directed to contact the service provider's License Service to request a License. The License request [12] includes public portions of the User Node that are used by the License Service to [13] *bind* (encrypt) the content key to the User Node. The License Service then [14] targets the License to the user by specifying in the Control object (in the License) that the User Node must be reachable. Any playback device that can construct valid Links to this user's User Node *and* satisfies any other constraints within the License will then be permitted to access the content. In the case of a rental, these additional constraints may include restrictions involving either the expiration date or the maximum elapsed time allowed by the rental term.

Delivery of content may occur at any time and via a variety of mechanisms. In the figure above, content delivery is illustrated as a simple request [16] and download [17]. However, since a Marlin Content object is self-protected (encrypted), it can be delivered via superdistribution or via physical media, broadcast, or push—at any time during, before, or after the process of obtaining the rights to use the content.

Note: The illustration in Figure 4 depicts a sequence of steps required for the first time a device acquires content and a License from a service. Subsequent purchases would not require reregistration or requests for a User Node. The actual sequence of steps is determined by the order of actions listed within the Action Token, and can vary by implementation or by situation.

5.2 Subscription

In a digital media context, a subscription business model typically refers to customers purchasing rights to access a large collection of content for a limited period of time. During the validity period of a subscription, a customer is permitted to play or use any content that is part of the subscription as many times as they wish. At the end of the validity period, the subscription may be renewed and the customer can continue to use the content. However, if the subscription is not renewed, the customer does not retain any rights to access the content, and any Licenses that were previously used to access the content are allowed to expire and become invalid.

By separating the functions of License management, content protection, and subscription membership, the Marlin architecture supports a very efficient model for managing digital subscriptions. This model allows a user's subscriber status to be easily updated (or renewed) without the costly and resource-consuming process of reissuing content and Licenses after each subscription period. A description of this renewal process is included in Figure 5 and the text that follows.

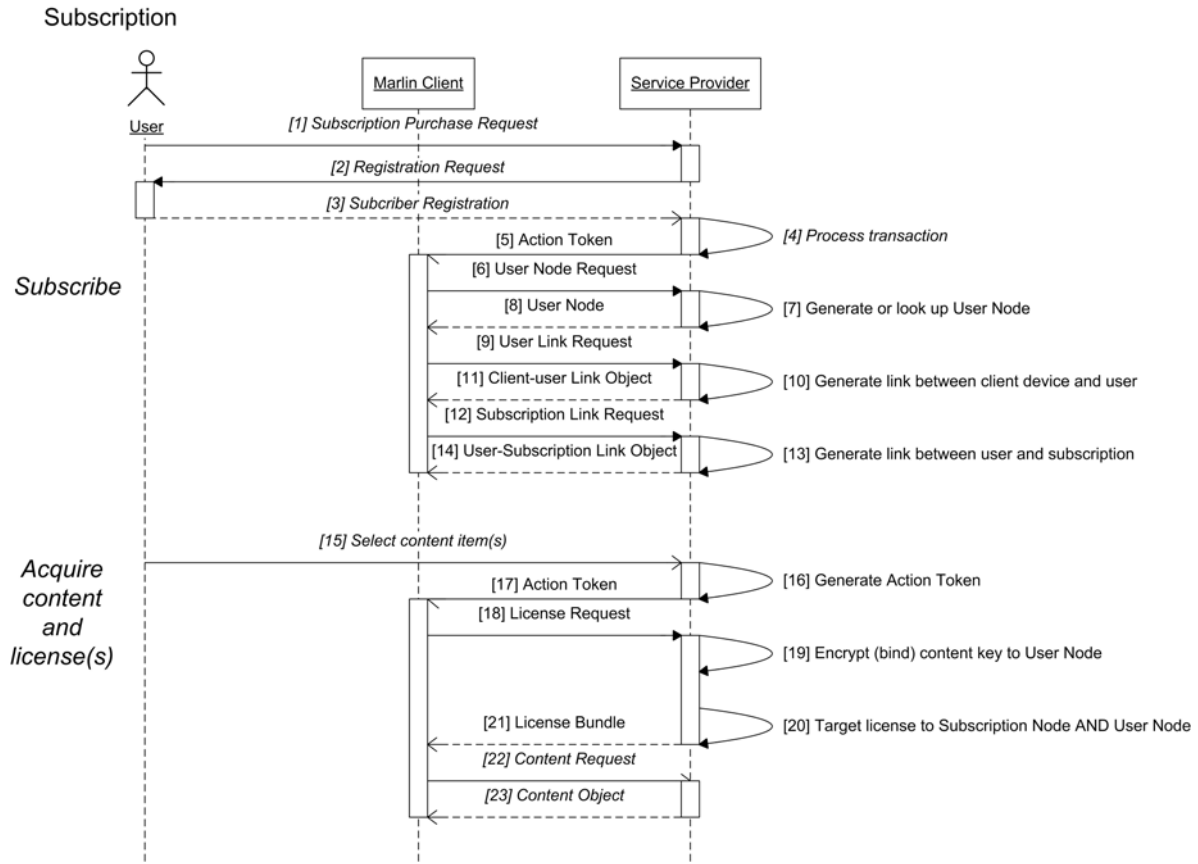


Figure 5 Subscription use case

Figure 5 illustrates a typical Marlin implementation of a subscription use case. As in the previous scenario, several steps (1-4, 15, 22, and the method of delivery for the objects specified in 5, 17 and 23) are not within Marlin's scope, but are included to provide context. The process begins with a user's request to purchase a subscription [1]. Upon receiving the user's request to subscribe, the subscription service provider's e-commerce system requests [2] and collects [3] information from the user to set up a subscriber account and complete the purchase. The service provider's e-commerce system then processes the transaction [4], and, when satisfied that the payment has been arranged, the service provider proceeds by issuing an Action Token [5] to the user's Marlin Client. The issuance of the Action Token begins the Marlin portion of the process.

The Action Token directs the Marlin Client to perform a sequence of activities, the first of which is to contact the service provider's Registration Service to [6] request a copy of the customer's User Node. If this is for a new customer, the Registration Service [7] generates a new User Node for the customer; otherwise it retrieves (or regenerates) a User Node from the customer's record. After the User Node is delivered [8], the client sends a request to the Registration Service for a Link object. The Registration Service generates [10], and responds [11] with a Link object that links the Client's Personality Node with the User Node.

Since this is a subscription purchase, the Action Token indicates that the device must also acquire a Link from the User Node to the Subscription Node. Consequently, the client submits a Subscription Link Request [12] to the Registration Service. The Registration Service generates a time-expiring Link object that links the User Node to the Subscription Node. The presence of a valid user-subscription Link object indicates that the user is a current subscriber in good standing. When this Link object expires at the end of the subscription period, it will need to be refreshed in order to continue to use it, as described in §5.3.

After the user has subscribed, he or she may browse a catalog of content associated with the subscription and can request specific content items [15]. The service provider then issues an Action Token [17] that triggers the Marlin Client to request Licenses for the chosen content item(s) [18]. The License request message includes public portions of the User Node that the License should be bound to, and business metadata about the content being acquired via subscription. The License Service then *binds* the License to the user's User Node (encrypts the content key with the User Node's public key) [19]; and then typically *targets* the License [20] to the user *and* the subscription. The License Service targets the License by specifying that both the User Node *and* the Subscription Node must be "reachable" by the playback device. These Nodes are reachable if the device possesses a chain of valid Links from its device Personality Node to the User Node *and* to the Subscription Node.¹

Similar to what was described in the simple content purchase scenario, delivery of the content may occur at any time and via a variety of mechanisms. In the figure above, content delivery is illustrated as a simple request [22] and download [23].

5.3 Subscription Renewal

Figure 6 below illustrates the process used to renew a subscription. As mentioned previously, the subscription Link object (i.e., the Link associating the user with the subscription) is issued with an expiration date coinciding with the end of the current subscription period. Since all Licenses received in conjunction with the subscription require a valid chain of Links to the Subscription Node, access to all of the subscription content will be denied if this subscription Link expires and it is not replaced with a new subscription Link object with an extended expiration date.

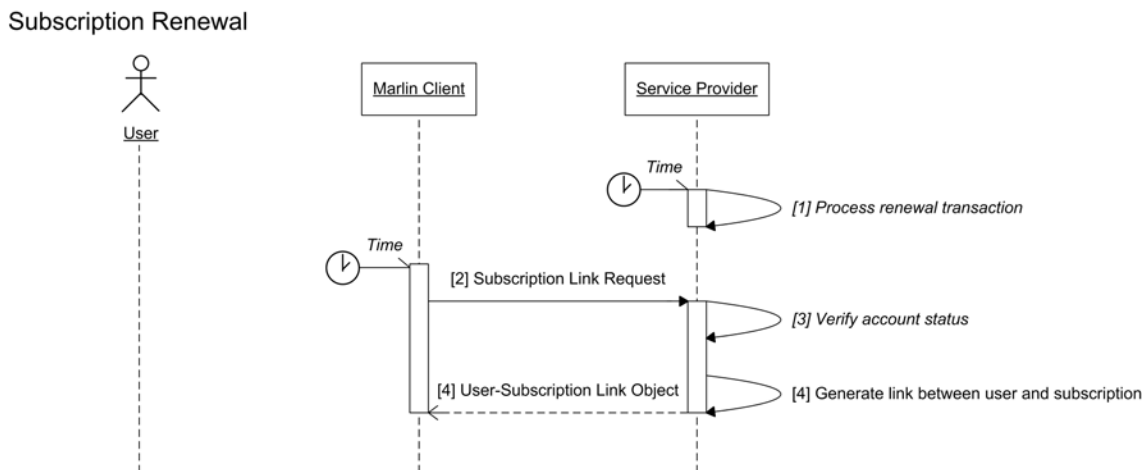


Figure 6 Subscription renewal

The renewal process illustrated in Figure 6 shows one of several methods by which a Marlin Client can request and obtain a new subscription Link. This example assumes that the service provider's e-commerce system will [1] automatically charge the user prior to the start of a new subscription period (unless previously cancelled by the user). At some point prior to the end of the current subscription period or at the start of the new subscription period (to be determined by the implementation), the Marlin Client is triggered to request a new subscription Link object. This trigger can be time-based, or the result of a user attempting access to subscription content, or another implementation-specific arrangement. In

¹ Since the License is bound to the User Node, it is recommended that the subscription Licenses be targeted to the User Node in addition to the Subscription Node. This ensures that the necessary links are available to decrypt the content key in the License.

response to the trigger, the Marlin Client submits a subscription Link request [2] to the service provider's Registration Service. If the user's account is verified to be in good standing [3], the Registration server generates a new subscription Link object [4] that links the user to the subscription and has an expiration date that coincides with the end of the next subscription period. After receiving the new, "refreshed" subscription Link [5], the Marlin Client will once again have a valid chain of Links to the Subscription Node, and the subscriber can continue to use all previously received subscription content and Licenses. As mentioned above, this is just one example of how renewals can be processed. The only requirement is that a new subscription Link must be acquired. The implementer can decide whether the specific delivery mechanism is online or offline.

6 Domains

Perhaps the most compelling, user-friendly feature of the Marlin DRM system is the flexibility it gives to users to transfer and consume content on all devices that they have access to. Unlike many other DRM systems, Marlin gives users the freedom to exchange and transfer content directly between any two devices that they own (without the need to synchronize Licenses), allows multiple family members to share their devices and content, and even allows temporary use of their content on "guest" devices (those owned by other individuals). Marlin's unique architecture permits this degree of user flexibility while still protecting content owners' interests and guarding against widespread copyright abuse and piracy.

Marlin provides this capability through its robust and flexible domain model. A Marlin domain consists of a collection of devices that can be associated with a particular user, or in the case of a family environment, all members of the family. As discussed previously, Marlin content Licenses are typically bound and targeted to the user; and thus, any device that can produce a chain of valid Links from its Personality Node to the User Node referenced in the License can use the content associated with that License. Marlin also allows the user to independently manage devices that belong to his or her "personal entertainment network" at any given time. By decoupling domain management from the License issuance process, Marlin allows consumers to configure (or reconfigure) domains as their needs or environment change. To guard against abuse, the system enforces a limit on the number of devices that may belong to a single domain. Likewise, individual devices may be limited as to the number of domains they may belong to at a given time. By setting reasonable limits on these constraints, Marlin is able to satisfy the needs of both the content consumer and the content owner. With Marlin domains, content owners can be assured that their content will not be exposed to widespread piracy, yet content consumers can also enjoy the flexibility they have grown to expect when using media content amongst all their devices.

The domain model builds upon the "Node and Link" architecture that is found in other parts of the Marlin governance model, as discussed in the following sections.

6.1 User-based Domains

A Marlin domain is considered to have a unique Domain Node. When domains are user-based, as in Marlin Broadband [MRL_BB], all devices linked to a particular user are considered members of the user's domain, and the "Domain Node" is simply the User Node. For each device that is a member of the domain for a particular user, a Link is issued that links the device's Personality Node to the User Node. The presence of a valid Link between the device and the user indicates that the device "belongs to" the user's domain.

Figure 7 illustrates a domain containing three devices for a user named Alice. Each device's Personality Node is linked to Alice's User Node. As a result, content purchased by Alice (that is, targeted and bound to her User Node) can be played on any of the devices in her domain.

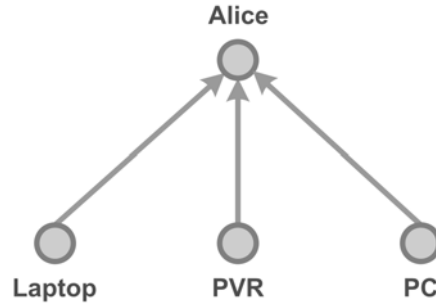


Figure 7 Alice's domain containing three devices

Figure 8 illustrates a single device that is a member of multiple users' domains. The device contains Links from its Personality Node to each of the User Nodes for family members Alice, Bob, and Carol. As a result, content purchased by any of the family members can be played on this device.

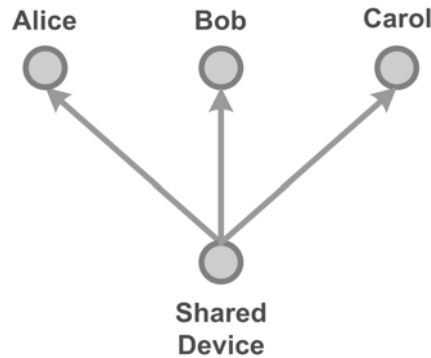


Figure 8 Single device as a member of multiple users' domains

The multi-device, multi-user domain concept is depicted in Figure 9 for a household with two shared devices in the home: a PC and a portable video recorder. Since each device is linked to each user in the example, any content purchased by a family member (in the past or future) can be used on *both* of the family's devices.

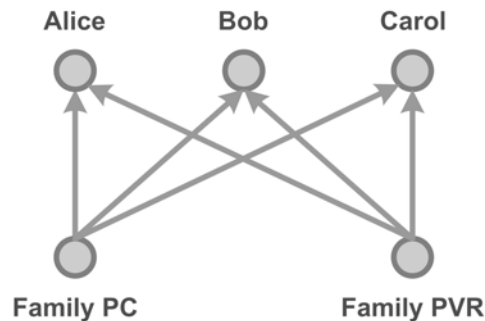


Figure 9 Multiple devices as members of multiple users' domains

Please note: In this context, the term "user" is meant to refer to either an individual or an entire group of people (such as a family) represented by a single User Node. For example, all members of a family may

share an account, which has a single User Node. Any devices linked to that User Node can play content acquired through that account. It is not necessary for each family member to have their own account and User Node, and thus it is not necessary for a device to be linked to individual User Nodes for each of the family members. Figure 10 shows an example of the Smith family's devices linked to a single "Smith Family" user domain.

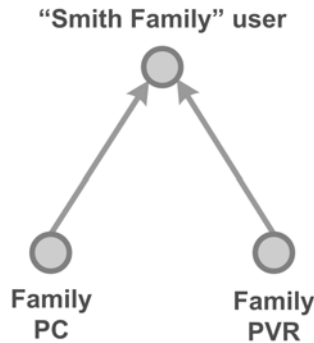


Figure 10 Multiple devices as members of “Smith Family” user domain

The domains discussed in the use case models in this document are user-based, as defined in the Marlin Broadband Specification [MRL_BB] and described in this section. First, however, the following section briefly describes a more general Marlin Common Domain model.

6.2 Marlin Common Domain Model

The user-based domain model can be considered a special case of a more general Marlin Common Domain model. In the Marlin Common Domain [MCD] model, a domain makes use of an intermediate Domain Node. That is, each device that is a member of a domain has a Link from the device Personality Node to the Domain Node, and multiple users can be associated with the domain, via Links between the Domain Node and the User Nodes.

The presence of a valid Link from the Domain Node to a particular user's User Node indicates that the domain (perhaps jointly) “belongs to” that user. In the Marlin Common Domain model, when you add a new device, you only need to register it to the intermediate Domain Node (that is, obtain a Link from the device Personality Node to the Domain Node) in order to be able to play content targeted to any user associated with the Domain Node.

The multi-user domain concept is depicted in Figure 11 for a typical household with multiple shared devices in the home. Since each device is linked to the domain (that is, the Domain Node), and the domain is linked to each user (User Node) in the example, any content purchased by a family member (in the past or future) can be used on *all* of the family's devices.

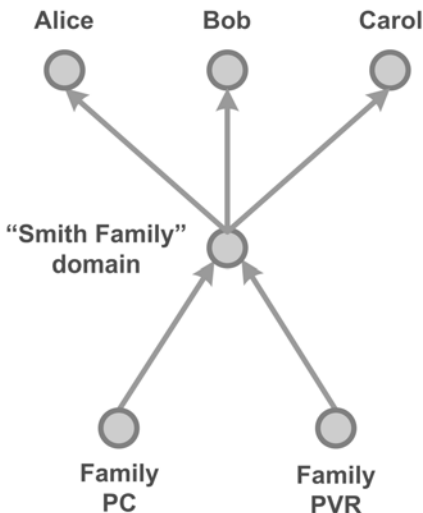


Figure 11 Multi-user Marlin Common Domain example for a typical household

7 Portability Use Cases

This section presents several examples that illustrate how the Marlin concepts described earlier are brought together to enable a seamless user experience allowing users to consume content on all Marlin-enabled devices in their environment. From a technical standpoint, this section is quite simple because no additional DRM objects need to be created, transferred, or synchronized between devices. In these scenarios, the framework for user portability has been created as a part of the Marlin architecture itself, and, as a result, the user portability scenarios “just work” without any outside interaction from the user or remote DRM services. Since there are no protocols to describe, the following examples focus on the user experience and how the Marlin architecture supports that experience.

7.1 Simple Portability Example

Ravi is an avid video fan and has several devices that he uses to purchase and play back video content. He occasionally purchases short video content with his laptop computer while on the road, but he usually purchases longer length feature content using devices in his home entertainment system that are connected to his always-on broadband connection. Regardless of which device he used to originally purchase the content, he wants the freedom to use it on all of the devices he owns, including a new portable video player he just purchased.

When Ravi brought home his new portable video player and connected it to his home network, he was prompted and asked whether he wanted to register it, that is, add it to a domain (group of devices). Ravi agreed, and pointed the device to the service that manages the domain he uses for all his devices. As part of the registration process, his new device was given a Link to the User Node representing an account (and thus the domain) Ravi has with the service. Since all of his content is locked (i.e., targeted and bound) to this User Node, and the new device now has a Link to that Node, all of Ravi’s content—regardless of what device he used to purchase it—can now be used on his new device also.

7.2 Multi-user / Multiple Domain Portability Example

Alice Smith is a college student who lives away at college for most of the year. While at college, she shares an apartment with her roommate, Jane. Both Alice and Jane each have their own laptop computer and portable media player, but they also share several devices in the apartment, such as their

PVR, a set-top box, and an old desktop computer that they have connected to the stereo to play their music playlists. Alice's laptop and portable media player are members of her domain of devices. That is, each is registered to her; each device has a Link from its Personality Node to her User Node. All content that Alice purchases can be played on either device, since such content is targeted and bound to her User Node, and each device has a Link to that Node. Similarly, Jane's laptop and portable media player are members of her domain, and all content she purchases can be played on either device. In addition, each of the devices they share is registered to both Alice and Jane (that is, has Links to both Alice's User Node and Jane's User Node). Now, regardless of what device Alice or Jane uses to purchase content, they can transfer their content to a shared device, and they can enjoy it together in their apartment (see Figure 12 below).

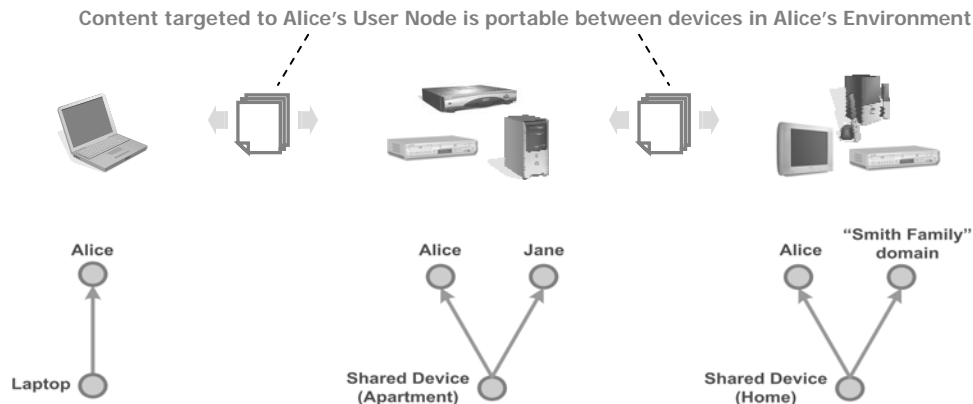


Figure 12 Multi-user multiple domain portability example

Continuing this example further, when Alice returns to her parents' home for the summer break, she brings her media devices loaded with content she has purchased throughout the year while at college. While at home, she decides to play some of her content on the family's entertainment system. The devices at home are all registered to the "Smith Family" domain, the account the family members use when they purchase content at home. That is, each device has a Link from the device Personality Node to the "Smith Family" User Node, and thus all content targeted and bound to that User Node can be played on any of the devices. Since the content Alice brings home is targeted to her User Node, she can play it on any device that can produce a chain of valid Links to her User Node. This will be the case for any device at home that is registered not only to the "Smith Family" domain but also to her domain.

Note that if Alice had returned from college with content purchased by Jane, that particular content would *not* be permitted to play on the Smith family devices. This is by design, since the Smith family devices are linked to Alice, but not to Jane. This is the correct behavior because members of the Smith family have not purchased rights to Jane's content. In this instance the roommate's content would be treated as superdistributed or shared content. The details of sharing and superdistribution are further discussed in the following sections.

7.3 Sharing and Superdistribution

The Marlin DRM architecture is unique in that it allows users to *share content with* others and *superdistribute content to* others. In a digital media context, superdistribution generally refers to the situation where a user receives content from a third party, typically from another user. Since the recipient does not have rights to use the content, the recipient is directed (usually via a URL in the metadata of the content) to a service provider where they can purchase rights to use the content. Marlin supports traditional superdistribution while also supporting the controlled sharing of content. In addition to supporting the traditional method of sharing content among members of the same domain, Marlin

makes it possible to implement innovative sharing methods, such as one where individuals not part of a domain are allowed to join it temporarily. This allows a user to visit a friend and use a device in the friend's home to view or listen to content the user has purchased. Alternately, it allows two users to share the experience of watching content on their portable devices while traveling together.

In the rest of this section and in Section 7.5, the term “sharing” refers to the temporary sharing of content with others not in the same domain.

Superdistribution and sharing processes are closely related, because in each case, a Marlin Client will initially receive content with a License that is bound to a “foreign” user. (“Foreign” in this context connotes a user for which the Marlin Client does not have a valid Link to the User Node referenced in the License). When the Marlin Client encounters such an instance, it first attempts to request updates of valid user Links from any Domain Managers that it is associated with. If this is not successful, the Marlin Client and host application can either direct the user to a location where a License can be purchased (i.e., the superdistribution option, as described and illustrated in §7.4) or obtain a temporary Link, in the sharing option (§7.5). The host media application can choose an action to take relative to the context; however in most instances, some user interaction will be required to decide which action is appropriate.

7.4 Superdistribution

As shown in Figure 13 below, the superdistribution use case begins when a Marlin Client receives a bundle [1] containing a Marlin Content object and its associated License. For purposes of this case, neither the location from which the content was received nor the mechanism by which it was received is important. The Marlin Client [2] checks whether the User Node (and possibly also the Subscription Node) referenced in the License is “reachable” by a chain of Links. In this instance, the Marlin Client cannot access the content because the License requires a Link to a Node that the client does not have. We assume here that the client exhausts all possibilities for acquiring the Link(s). If the host application has the capability of connecting to a service provider, it now [4] asks the user if they would like to purchase (or subscribe) to obtain access to the content. If the user agrees [5], the host application can direct the user to a service provider URL that is embedded in the content metadata [6]. From this point, the user can renew Links, subscribe, or purchase content following the processes described previously for simple purchase (§5.1), subscription purchase (§5.2), or subscription renewal (§5.3).

Superdistribution

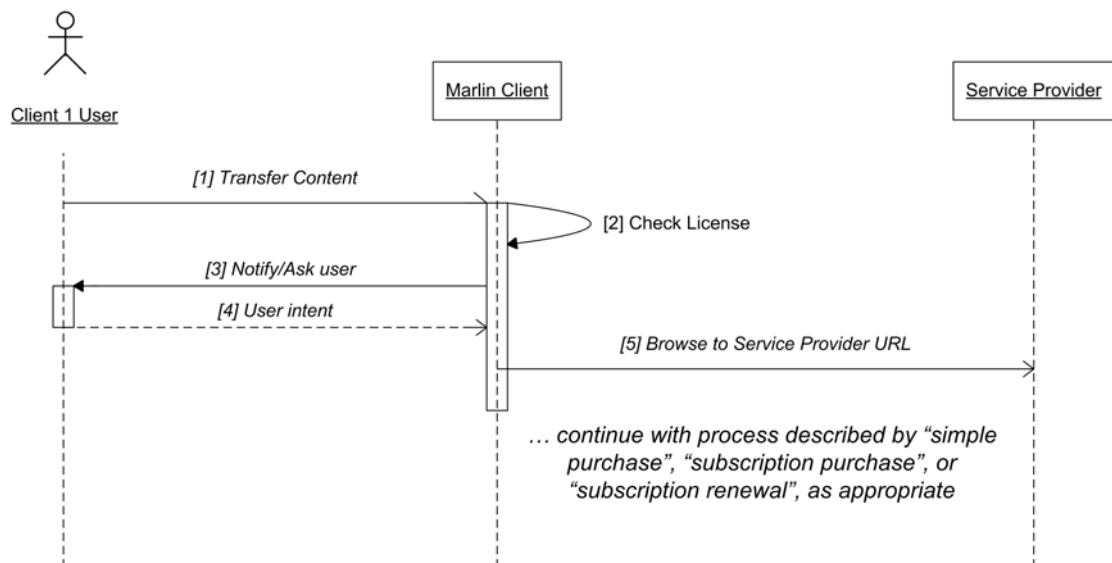


Figure 13 Superdistribution process

7.5 Sharing

To illustrate a sharing use case, we continue the example introduced in Section 7.2. In this scenario, we consider the case where Alice returns home for a college break with her roommate Jane. Jane brought along her portable video player, which is loaded with several recently released movies that she previously downloaded and purchased. During the visit, Jane wants to share a movie she purchased and allow the entire family to watch it using the DVR and the large-screen TV in the family room.

An example of this use case is illustrated in Figure 14. The initial steps [1-6] of the process are identical to that of the superdistribution use case. The process begins when Jane transfers her movie (and its embedded License) to the Smith Family DVR. After determining that the content cannot be accessed, the user is directed to the service provider that distributed the original content and License. However, in this scenario, the user (in this case Jane) is also the License Owner for the movie. Since Jane has an account with this service provider, she [7-8] logs in to the service and requests the temporary addition of the Smith family's DVR to her list of devices. To prevent abuse, the service provider's system will [9] check her account to determine if she is permitted to add a temporary device. To make this determination, the service provider may check several parameters, including the total number of devices already associated with her account or the elapsed time since other devices were temporarily added to her account. If the service provider allows the device to be added, it [10] delivers an Action Token to the DVR.

The Action Token directs the DVR's Marlin Client to perform a sequence of activities, the first of which is to contact the service provider's Registration Service to [11] request a copy of Jane's User Node. After the User Node is delivered [13], the client will send a request [14] to the service provider's Registration Service for a Link to that User Node. The Registration Service generates [15], and responds [16] with a Link that associates Jane's User Node with the DVR's Client Personality Node. The DVR now has a Link to the User Node required in the License, and the movie can be played. When the temporary Link to Jane's User Node expires, the User Node will no longer be "reachable" and the content will not be allowed to play.

Sharing

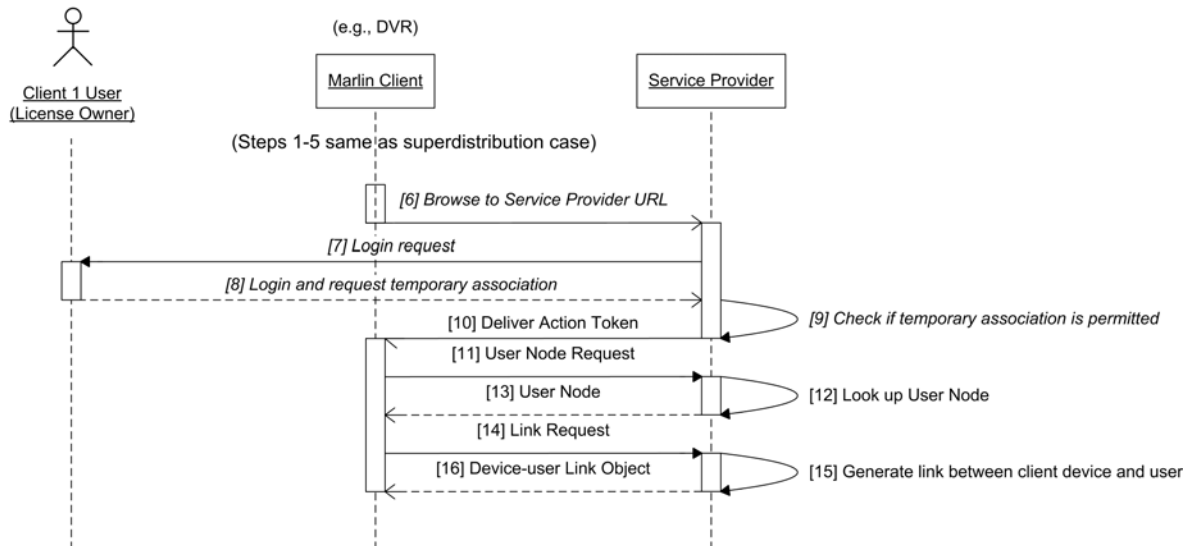


Figure 14 Sharing example

8 Alternate Delivery Scenarios

Although much of the discussion thus far has centered around broadband (IP network-based) usage models, the Marlin DRM architecture also provides comprehensive support for broadcast and mobile delivery options. As an example, the following section briefly describes one possible mobile approach and a typical scenario for its usage.

8.1 Mobile Delivery using OMARlin

The OMARlin Specification [OMARLIN] tells how to enable interoperable download, streaming, sharing, and consumption of content between the OMA and Marlin DRM systems. OMARlin content protection uses either OMA DRM 2.0 [OMA_DRM] or Marlin Broadband [MRL_BB] to enable the distribution and consumption of digital content in a controlled manner. The basic approach of OMARlin is indicated in Figure 15.

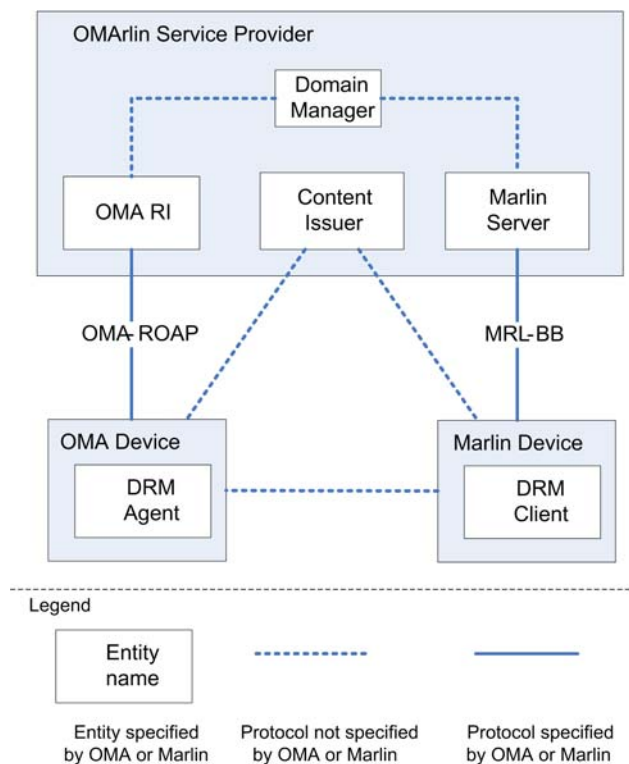


Figure 15 OMARlin configuration

In order to gain access to either an OMA Rights Object (RO) or a Marlin License, an OMA or Marlin device must authenticate and register with an OMARlin Service Provider and join the domain for the user to which the rights/License applies. If the device implements an OMA DRM Agent, it will join the domain and obtain the Rights Object from the OMA Rights Issuer (RI) via the ROAP protocol. If the device implements a Marlin DRM Client, then it will join the domain and obtain the Marlin License from the Marlin Server using the Marlin Broadband protocols. Guarding the overall limits on the size of the domain is performed by the OMARlin service through synchronization between the OMA RI and the Marlin Server via protocols that are not specified by the OMARlin guidelines.

Central in OMARlin is the choice for a content format that is supported by both OMA and Marlin devices. This content format is based on the OMA content format. An *OMARlin File*, a content file suitable for use

by both OMA and Marlin systems, uses the *OMArLin Content Format* defined in the OMArLin Specification [OMARLIN]. It can contain both OMA Rights Objects and Marlin Licenses, as well as OMA- and Marlin-specific information that allow a device to retrieve a License in the required DRM format.

In order to share the same protected file among different devices of the same user, OMArLin uses *domains*. A domain can consist of different types of devices (mobile telephones, portable players, PC players, etc.) and may contain both devices with OMA DRM Agents and devices with Marlin DRM Clients.

Devices can freely exchange OMArLin Files with embedded OMA Rights Objects and Marlin Licenses. Protected content issued for a specific domain can be consumed on any device that has joined this domain. In this way, the devices that are part of a user's domain will be able to access the embedded OMA Rights Object or Marlin License and consume the content ("in-home distribution"). An example of a use case where a downloaded OMArLin file contains both an OMA Rights Object and a Marlin License is illustrated by Figure 16.

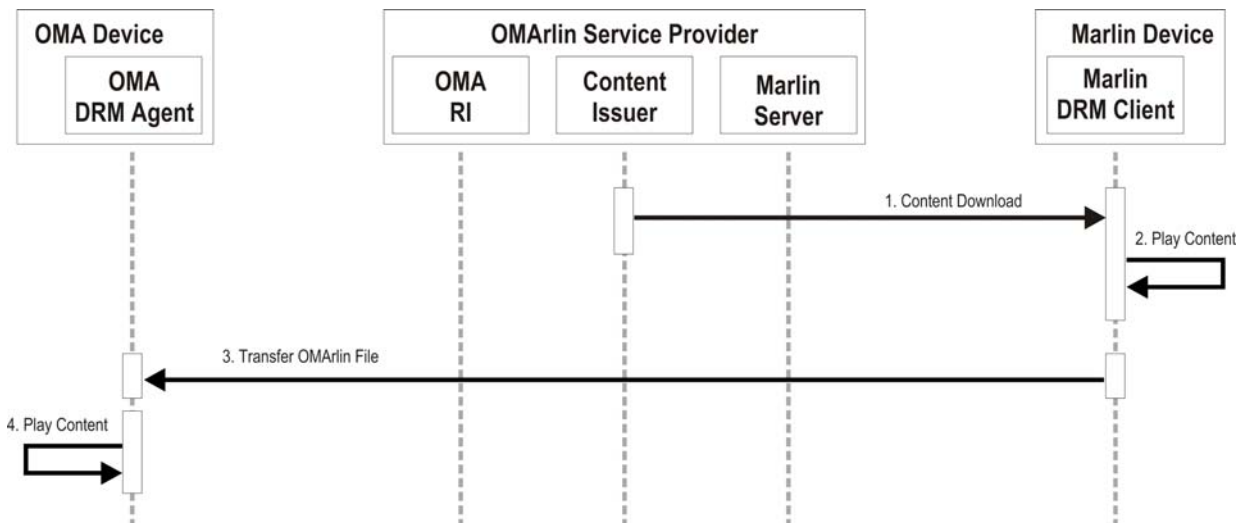


Figure 16 Sample OMArLin use case

The steps illustrated by Figure 16 are the following:

- 1: After a purchase transaction between the Marlin device and the OMArLin Service Provider (not shown), the Marlin device downloads the OMArLin File. In this example, the downloaded OMArLin File contains both an OMA Rights Object and a Marlin License.
- 2: The Marlin device may now access the OMArLin File using the embedded Marlin License (assuming the Marlin device has already joined the domain).
- 3: The OMArLin File is transferred to the OMA device via a protocol not specified by OMArLin.
- 4: The OMA device may now access the OMArLin File using the embedded OMA Rights Object (assuming the OMA device has already joined the domain).

It is up to the OMArLin Service Provider as to whether to include both an OMA Rights Object and a Marlin License in an OMArLin File when an OMA or Marlin device initially purchases content and downloads the file. If the OMArLin File just includes one or the other, and the file is subsequently

transferred to a different type of device, that device can obtain the missing information (Rights Object or Marlin License) it needs and embed it in the OMArlin File.

Devices belonging to other users may initiate a purchase transaction with the OMArlin service ("superdistribution"), using the superdistribution information contained in the OMArlin File.

9 Summary

The Marlin architecture is an extremely flexible, user-centric approach to digital rights management and content sharing. By targeting Licenses to users rather than devices, Marlin permits consumers to freely use and exchange content between all devices in their personal entertainment network. Further, Marlin allows consumers to acquire content through multiple delivery channels (e.g., broadcast, broadband, etc.) while still maintaining a consistent, user-friendly experience.

Although many diverse use cases and business models have been included in this overview, this only represents a fraction of the potential opportunities available to implementers. Due to Marlin's modular architecture, implementers may combine Marlin components in the best way to serve their particular business needs, while still preserving interoperability within the larger Marlin ecosystem. Detailed technical specifications for these modular components are defined in the Marlin Core Specification and the associated Marlin delivery system specifications.

10 References

- [IPTV] Marlin IPTV End-point Service Specification
Version 1.0
July 20, 2006
marlin-IPTVESSpecification.pdf
- [MCD] Marlin Common Domain Specification
Version 1.1.1
September 2006
marlin-CommonDomainSpecification.pdf
- [MCS] Marlin Core System Specification
Version 1.3
September 13, 2006
marlin-CoreSystemSpecifications.pdf
- [MRL_BB] Marlin Broadband Delivery System Specification
Version 1.2
September 8, 2006
marlin-BroadbandSystemSpecifications.pdf
- [OCTCTRL] Octopus Controls
Version 1.0.1
September 6, 2006
Octopus-Controls.pdf
- [OCTOBJ] Octopus Objects
Version 1.0.1
September 14, 2006
Octopus-Objects.pdf
- [OMA_DRM) Open Mobile Alliance
DRM specification
Approved Version 2.0 – 03 Mar 2006
OMA-TS-DRM-DRM-V2_0-20060303-A

[OMARLIN]	OMArlin Specification Version 1.0 April 17, 2007 marlin-OMArlinSpecifications.pdf
[ROLE_NEMO]	The Role of NEMO in Marlin RoleofNEMOinMarlin.pdf
[ROLE_OCT]	The Role of Octopus in Marlin RoleofOctopusinMarlin.pdf