ICE: Information and Content Exchange Protocol

ICE Implementation Cookbook

Version 2.0

2004 08 01

This version

http://www.icestandard.org/Spec/SPEC-ICE-2.0Cookbook.pdf

Latest version

http://www.icestandard.org/Spec/SPEC-ICE2.0d.pdf

Previous version

http://www.icestandard.org/Spec/SPEC-ICE1.1.htm

Editors:

Jay Brodsky, Tribune Media Services

Marco Carrer, Oracle Corporation

Bruce Hunt, Adobe Systems, Inc.

Dianne Kennedy, IDEAlliance

Daniel Koger, Independent Consultant

Richard Martin, Active Data Exchange

Laird Popkin, Warner Music Group

Adam Souzis, Independent Consultant

Status of this Document

This document is an approved IDEAlliance Specification. It represents a significant step towards a stable specification suitable for widespread dissemination and implementation. It has been reviewed and approved by the ICE Authoring Group of IDEAlliance.

ICE 2.0 is the first major revision of the ICE Specification. As such, ICE 2.0 is not a compatible update to the ICE 1.0 specification. This update is a response to the implementation experience that has been gained over the past four years as well as the advancement in technology and W3C Recommendations. It differs from the ICE 1.0 and ICE 1.1 specifications in that it is specifically designed to support a Web Services model for syndication, has been modularized, incorporates XML Namespaces, and moves from an XML DTD to XML Schema.

As of this publication, the ICE Specification has been organized into a set of documents. This is one document in a set of documents (ICE Primer: Introduction and Overview, ICE Cookbook, Basic ICE Specification , Full ICE Specification, ICE Schemas and Scripts, and Guidelines to Extending the ICE Protocol) intended to jointly replace ICE 1.1. It has been developed by the IDEAlliance ICE Authoring Group. New documents may be added to this set over time.

The ICE Authoring Group and IDEAlliance recommend that implementations be updated to conform to the new ICE 2.0 Specification. The new specification embraces the latest Web technologies and W3C Recommendations. It provides added functionality that greatly enhances the usability of the protocol in a very wide range of syndication applications and can provide a substantial foundation for delivering syndication solutions in a Web Services environment.

Abstract

This ICE Cookbook was written by the authors of the ICE (Information and Content Exchange) Specification to assist in implementing ICE in a wide variety of situations. Every system designer attempts to balance the near-term need for an immediate solution with the long-term desire for a well-structured and complete solution. The ICE Authoring Group recognizes the need for solving simple problems with simple solutions, and more complex problems with more sophisticated answers. The recipes included in this cookbook, start with simple Basic ICE solutions and add capabilities with each succeeding recipe. The objective of this recipe book is to start with the Basic ICE protocol and add capabilities until a Full ICE compliant implementation with push delivery, subscription management and delivery confirmation is achieved. This implementation cookbook is a way to provide the Web community with a set of practical implementation steps that both get the job done quickly and efficiently as well as providing a path so that your near-term investment in building an easy solution contributes to the long-term goal of Full ICE syndication capabilities.

Table of Contents

1. Introduction

The ICE specification consists of over one hundred pages describing the ICE protocol. When you read the specification it is difficult to know where to begin an ICE implementation. This cookbook provides an easy step-by-step set of recipes for implementing ICE. It starts with easy-to-implement Basic ICE solutions and adds capabilities of the Full ICE protocol in a step-by-step manner to support more sophisticated syndication scenarios.

1.1 Basic Terms

Although this cookbook is designed to provide simple ICE implementation advice, some basic nomenclature is required to understand the text:

ICE:

Information and Content Exchange protocol

Syndicator:

A content aggregator and distributor.

Subscriber:

A content consumer.

Catalog

A package of subscription offers.

Subscription:

An agreement between a subscriber and a Syndicator for the delivery of content according to the delivery policy and other parameters in the agreement.

Package:

A single delivery instance of a group of items. For example, a package is a single issue of a parts manual or a single set of headlines. A package is the atomic unit of information distribution in ICE. A package is also used to distribute ICE offers.

Full Update

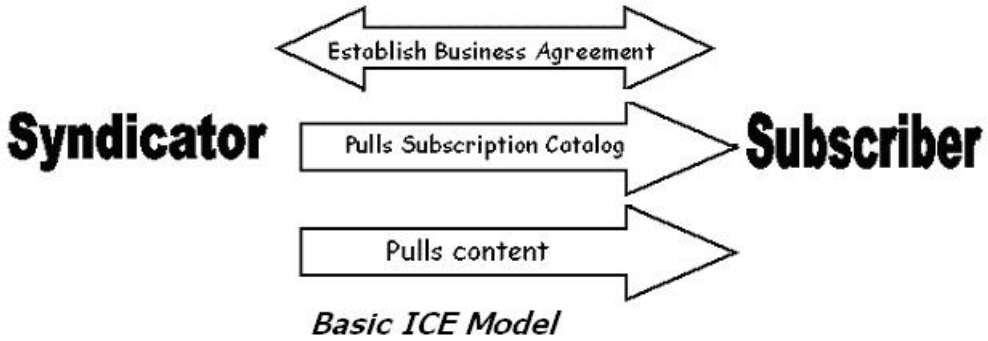
A complete set of all items within a subscription are delivered with each update. Basic ICE only allows for this update method.

Incremental Update

A set of only the items that have changed within a subscription are delivered with each update. Basic ICE does not allow for this update method.

Basic ICE:

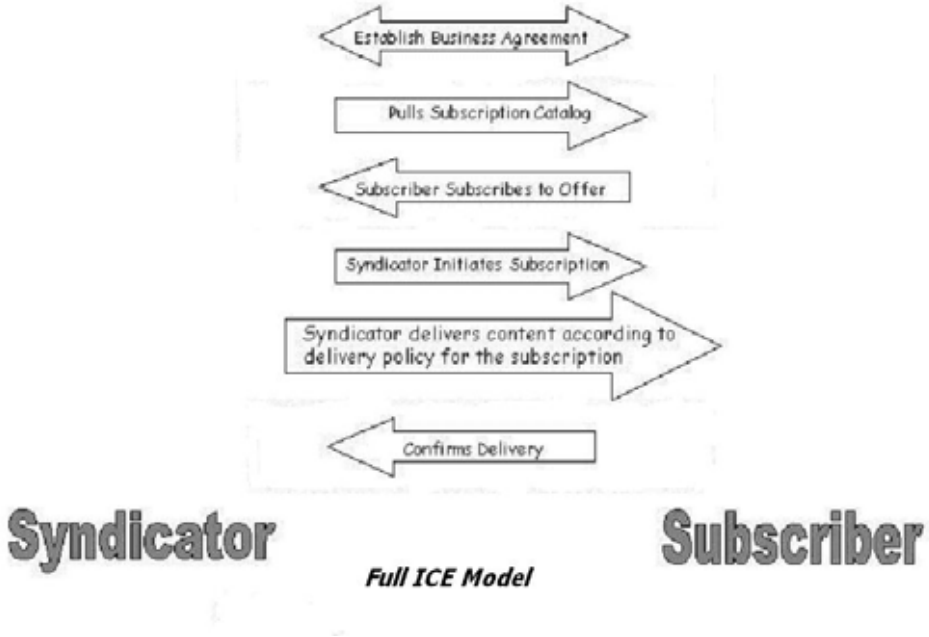
The Basic ICE level of conformance provides for very simple syndication functionality. Basic ICE enables the Syndicator to post messages to a URL where the Subscriber can "get" them.



Basic ICE Model

Full ICE:

Full ICE implements SOAP transport bindings for push or pull delivery, supports subscription management and delivery verification.



Full ICE Model

1.2 About the Recipes

The recipes are as follows:

1. **Pull public content reference:** Syndicator posts ICE package with content references on a publicly known URL. Subscriber pulls the ICE package using `HTTP:GET`. This recipe is for Basic ICE functionality.
2. **Pull public content directly.** Syndicator posts ICE package containing *inline content* on a publicly known URL. Subscriber pulls the ICE package using `HTTP:GET`. This recipe is for Basic ICE functionality.
3. **Pull public content from a catalog listing.** Syndicator posts a public ICE *"catalog" of offers* at a known URL from which Subscribers can pull public content. Subscriber pulls or links to content from URL listed in the Catalog using `HTTP:GET`. This recipe is for Basic ICE functionality.
4. **Add subscription management.** In this recipe, the Syndicator posts a catalog for valued content at a known URL. This catalog includes a more detailed delivery policy than we have seen before. The Subscriber *selects an offer* from the catalog and requests a subscription using ICE messages. Syndicator *confirms the subscription, initiates subscription and begins posting complete content updates* for the Subscriber to pull according to delivery rules in the offer. This recipe adds subscription management capabilities and is a Full ICE recipe.

5. **Add incremental updates.** Syndicator posts a catalog for valued content at a known URL. Subscriber selects an offer and subscribes to that offer using ICE messages. Syndicator initiates subscription and begins providing incremental (partial) content updates for the Subscriber to pull according to delivery rules in the offer. This recipe adds incremental update capabilities and is a Full ICE recipe.
6. **Add push delivery.** Syndicator pushes a catalog for valued content to the Subscriber as per business agreement. Subscriber subscribes to an offer for push delivery, selects delivery rules and provides delivery endpoint using ICE messages. Syndicator initiates subscription and begins pushing content to the Subscriber. This recipe adds push delivery capabilities and is a Full ICE recipe.
7. **Add Security and Confirmation of Delivery.** Syndicator pushes a catalog for valued content to the Subscriber as per business agreement. Subscriber subscribes to an offer for push delivery that requires confirmation of delivery. The Subscriber provides a delivery endpoint and a security login and password. Syndicator initiates subscription and begins pushing content to the Subscriber requiring the subscriber to confirm receipt of the content. Subscriber verifies content delivery. This recipe adds security and confirmation of delivery receipt capabilities. It is a Full ICE recipe.

1.3 Examples

To make the recipes concrete and complete, we'll provide an example. We'll highlight the places where you will replace example specific implementation information with your own changes so that you can easily see how to apply the recipe to your environment.

The example we'll use is the following.

We'll look at both sides of content supply and consumption to illustrate how syndication works.

Consider first that you are interested in news and views about the latest in personal electronic gadgets. Let's suppose that your name is "Joe Cool" and you want to regularly obtain the latest information about the wonderful world of emerging personal electronic gadgets. This makes you an ideal subscriber (a content consumer) if you find someone that provides this information.

Consider second that you want to provide others access to all the information about personal electronic equipment (gadgets!) that you've been avidly collecting. To do this, you might put up a web site; or you could become a content Syndicator (a content supplier). As a content Syndicator, you will provide daily updates of the latest news to interested subscribers. As a content Syndicator, let's suppose your name is "Brad" and your domain name is "BradsGadgets". You share your extraordinary contacts and insight into the gadget world through various content feeds on your web site, such as newsletters, reports, columns, etc. It is well known that Brad is interested in various gadget areas and

has information and opinions on Digital Cameras, Personal Digital Assistants, Media Players, Cell Phones and Laptop Computers.

1.4 Recipe Organization

Now, a word about the way each recipe is organized. Each of the recipes has a background section that sets up the problem that the recipe solves and describes the general form of the solution. This is followed by implementation steps for Syndicators and implementation steps for Subscribers. These implementation steps are the real meat of the recipe. The steps are laid out in a step-by-step manner using parts of the example above to make it concrete. We've provided sample XML documents and sample URL references as well as descriptions of what is needed for code. These samples are given a special format so you can easily find them. In the examples, we have highlighted the lines that you will need to change to fit your own application using bold text. See the example below:

```
<icemes:Header>
  <icemes:Sender name='bradsgadgets'/>
</icemes:Header>
```

When you see a phrase in bold, you should think about how you want to replace it for your application.

2. Recipes

2.1 Pull Public Content References.

2.1.1 Background

In this recipe, a Syndicator provides content by posting an ICE package (SOAP/ICE XML document) with content references (pointers to the content) on a publicly known URL. Subscribers pull the ICE package using `HTTP:GET` and then access content from the references. This recipe is for Basic ICE functionality.

2.1.2 Syndicator Implementation Steps

1. Construct the newsletter, "Tech Tips from Brads Gadgets" using your normal production process. And put it on your website
2. Construct a standard ICE package that references the newsletter with a URL: This package is an XML-encoded document with a SOAP envelope and an ICE message header and an ICE package containing a link/reference to the newsletter content See the following example.

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2004-04-04' message-id='m1000'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
        sender-id='http://www.bradsgadgets.com'/>
   </icemes:Header>
  </env:Header>
  <env:Body>
    <icedel:package
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/delivery'
       package-id='P12' subscription-id='3'>
       <icedel:add>
          <icedel:item-ref item-id='xx203'>
           <icedel:reference
```

```
url='http://www.bradsgadgets.com/news/techtips/newsletter.htm'/>
          </icedel:item-ref>
        </icedel:add>
    </icedel:package>
  </env:Body>
<env:Envelope>
```

3. Place the ICE package at in a location that you make known to potential subscribers, such as `http://www.bradsgadgets.com/ice/techtips.ice`.

That's all that is required of the Syndicator.

2.1.3 Subscriber Implementation Steps

1. Obtain the URL for the ICE package from the Syndicator. In this example, you obtain the URL `http://www.bradsgadgets.com/ice/techtips.ice.`
2. Fetch the URL using `HTTP:GET` to pick up the ICE package containing the link to the newsletter.
3. Parse the URLs out of the `<icedel:package/<icedel:item-ref` and either download the content or reference it using the URL.
4. Periodically you will re-fetch the URL using `HTTP:GET` to determine the link to the next newsletter.

That's all there is to it! You now have the simplest possible ICE syndication.

2.2 Pull Public Content Directly.

2.2.1 Background

In this recipe, a Syndicator posts ICE package containing *inline content* on a publicly known URL. Subscribers pull the ICE package using `HTTP:GET`. This recipe is for Basic ICE functionality.

2.2.2 Syndicator Implementation Steps

1. Construct the newsletter, "Tech Tips from Brads Gadgets" using your normal production process.
2. Construct a standard ICE package that contains the newsletter. In the example the newsletter is coded in HTML 4.0. (this is not base 64)

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2004-04-04' message-id='m1000'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
        sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
    <icedel:package
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/delivery'
```

```
       package-id='P12' subscription-id='3'>
       <icedel:add>
          <icedel:item>
           <html:html xmlns:
                  html='http://www.w3.org/TR/REC-html40'>
            <html:head>
             <html:title>Brad's Gadget
Newsletter</html:title>
            </html:head>
            <html:body>
             <html:H1>BRAD'S GADGETS April 3,
2004</html:H1>
            . . . .
            </html:body>
            </html:html>
          </icedel:item>
        </icedel:add>
    </icedel:package>
  </env:Body>
<env:Envelope>
```

4. Place the ICE package at some public location on your website, such as `http://www.bradsgadgets.com/ice/techtips.ice`.

That's all that is required of the Syndicator.

2.2.3 Subscriber Implementation Steps

1. Obtain the URL for the ICE package, from the Syndicator. In this example, you obtain the URL `http://www.bradsgadgets.com/ice/techtips.ice`.
2. Fetch the URL using `HTTP:GET` to pick up the ICE package containing the newsletter.
3. Parse the content out of the `<icedel:package/<icedel:item` for download That's all there is to it! You now have the simplest possible ICE syndication.
4. Periodically you will re-fetch the URL using `HTTP:GET` to pull the next newsletter.

2.3 Pull Public Content From a Catalog Listing.

2.3.1 Background

In this recipe the Syndicator posts a public ICE "catalog" of offers at a publicly known URL. Subscribers pull the catalog and select offers. Subscribers then pull or link to content from URL listed in the Catalog using `HTTP:GET`. This recipe is for Basic ICE.

2.3.2 Syndicator Implementation Steps

1. Construct the newsletter, "Tech Tips from Brads Gadgets" using your normal production process.
2. Construct a standard ICE package (SOAP/ICE/XML document) containing a catalog offer:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-03-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
<icedel:package
     xmlns:icedel='http://icestandard.org/ICE/V20/delivery'
     subscription-id='1'>
  <icedel:add>
    <icedel:item>
      <icesub:offer
    xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
    offer-id='BradsGadgets04'
    name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
```

```
       <icesub:delivery-policy>
          <icesub:delivery-rule>
           <icesub:transport>
            <icesub:delivery-endpoint
url='http://www.bradsgadgets.com/news/techtips/newsletter.htm'/>
            </icesub:transport>
          </icesub:delivery-rule>
       </icesub:delivery-policy>
      </icesub:offer>
    </icedel:item>
  </icedel:add>
</icedel:package>
</env:Body><env:Envelope>
```

Note: this example is very simple and uses the default values for a Basic ICE `<delivery-rule`:
» mode=pull
» incremental-update=false
It also uses default values for a Basic ICE `<transport`:
» Protocol=http:get
» Packaging-style=ice
Also Note: When the `subscription-id` is equal to "1" the package contains a catalog.

3. Place the ICE catalog at some public location on your web-site, such as `http://www.bradsgadgets.com/ice/techtips.ice`.

That's all that is required of the Syndicator.

2.3.3 Subscriber Implementation Steps

1. Obtain the URL for the ICE catalog from a potential Syndicator. In this example, you obtain the URL `http://www.bradsgadgets.com/ice/techtipscatalog.ice`.
2. Fetch the URL for the ICE catalog using `HTTP:GET`.
3. Parse the `<delivery-endpoint` URL out of the catalog and download the content of the newsletter.
4. Periodically you will re-fetch content from the `<delivery-endpoint` URL to get the next newsletter.

2.4 Add Subscription Management

2.4.1 Background

In this recipe, the Syndicator posts a catalog for valued content at a known URL. This catalog includes a more detailed delivery policy than we have seen before. The Subscriber selects an offer from the catalog and requests a subscription using ICE messages. Syndicator confirms the subscription, initiates subscription and begins posting complete content updates for the Subscriber to pull according to delivery rules in the offer. This recipe adds subscription management capabilities and is a Full ICE recipe.

2.4.2 Syndicator/Subscriber Business Agreement

When a syndication relationship is for valued content, a business agreement between the Syndicator and Subscriber is typically the first step of establishing a syndication relationship. In this discussion the type of content and the value of content is established.

2.4.3 Syndicator Implementation Steps

1. Construct a standard ICE package (SOAP/ICE/XML document) containing a catalog:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
<icedel:package
     xmlns:icedel='http://icestandard.org/ICE/V20/delivery'
     subscription-id='1'>
  <icedel:add>
    <icedel:item>
      <icesub:offer
    xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
    offer-id='BradsGadgets04'
    name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
             stop-date='2004-12-31T24:00:00'>
          <icesub:delivery-rule weekday='Monday'/>
       </icesub:delivery-policy>
      </icesub:offer>
    </icedel:item>
  </icedel:add>
</icedel:package>
</env:Body><env:Envelope>
```

2. Place the ICE catalog at the location on your website that was agreed upon in the business agreement, such as `http://www.bradsgadgets.com/ice/techtips.ice`.

Note: In this example we have added parameters to the delivery rule that indicate that this catalog offers content from the first of January through the end of December in 2004. It also specifies that new content will be available each Monday.

Posting the catalog is the first step for the Syndicator.

2.4.4 Subscriber Implementation Steps

1. Pull the ICE catalog from the Syndicator using `HTTP:GET`.
2. Process the catalog and evaluate the offer in terms of quantity and delivery policy. Select an offer that you wish to subscribe to.
3. Return a <subscribe message indicating your choice of offer and requesting the subscription to begin.

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
    <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
            sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
     <icesub:subscribe
xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            offer-id=' BradsGadgets04'/>
  </env:Body>
</env:Envelope>
```

Note: In this recipe, the Subscriber is accepting the offer exactly as it was given. Because the delivery mode is pull, the Subscriber need not return a delivery endpoint. So, in this case, the Subscriber can simply echo the `offer-id` to identify the offer and need not repeat the offer inside the `<subscribe` message.

2.4.5 Syndicator Implementation Steps

Following the receipt of a <subscribe message from the Subscriber, the ball is back in the Syndicator's court. The Syndicator must process the subscribe message and decide whether to accept the subscription. In this example the subscription is accepted and the Syndicator must return a <subscription message. This message echoes the offer to which the Subscriber is subscribing to and provides a unique subscription identifier that will be used to track the subscription throughout its term.

1. Construct an ICE package containing the <subscription message:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
     <icesub:subscription

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            subscription-name='BradsGadgetsNewsletter04'
            subscription-id='BradsGadgets04kk3'/>
      <icesub:offer

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
         offer-id='BradsGadgets04'
         name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
          <icesub:delivery-rule weekday='Monday'>
           <icesub:transport>
            <icesub:delivery-endpoint

url='http://www.bradsgadgets.com/news/techtips/newsletter.h
```

```
tm'/>
            </icesub:transport>
          </icesub:delivery-rule>
        </icesub:delivery-policy>
      </icesub:offer>
</icedel:subscription>
</env:Body><env:Envelope>
```

Note: In this message the Syndicator provides the `subscription-id` as well as the `delivery endpoint`. This recipe is for the default pull delivery, so the endpoint lets the Subscriber know where the content can be pulled from.

2.5.6 Subscriber Implementation Steps

1. Pull an ICE package from the Syndicator according to the <delivery-policy in the subscription message and using `HTTP:GET`.
2. Re-Fetch complete new content for the subscription according to the `<delivery policy`.

2.6 Add Push Delivery

2.6.1 Background

In this recipe, the Syndicator pushes a catalog for valued content to the Subscriber as per business agreement. Subscriber subscribes to an offer for push delivery, selects delivery rules and provides delivery endpoint using ICE messages. Syndicator initiates subscription and begins pushing content to the Subscriber. This recipe adds push delivery capabilities and is a Full ICE recipe.

2.6.2 Syndicator/Subscriber Business Agreement

When a syndication relationship is for valued content, a business agreement between the Syndicator and Subscriber is typically the first step of establishing a syndication relationship. In this discussion the type of content and the value of content is established.

2.6.3 Syndicator Implementation Steps

Following the establishment of a business agreement between the syndication partners, the catalog is constructed. In this recipe, the Syndicator will push the catalog to the Subscriber.

1. Construct a standard ICE package (SOAP/ICE/XML document) containing a catalog:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
<icedel:package
     xmlns:icedel='http://icestandard.org/ICE/V20/delivery'
     subscription-id='1'>
  <icedel:add>
    <icedel:item>
      <icesub:offer
    xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
    offer-id='BradsGadgets04'
    name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
             weekday='Monday'
             starttime='00:01:00'>
          <icesub:transport protocol='soap'
              packaging-style='ice'/>
         </icesub:delivery-rule>
       </icesub:delivery-policy>
```

```
      </icesub:offer>
    </icedel:item>
  </icedel:add>
</icedel:package>
</env:Body><env:Envelope>
```

Note: In this example we have added parameters to the delivery rule that indicate that this catalog offers content from the first of January through the end of December in 2004. We are also indicating that we will push content to the Subscriber using SOAP each Monday beginning at 12:01:00 am.

Pushing the catalog to the Subscriber is the first step for the Syndicator in this recipe.

2.6.4 Subscriber Implementation Steps

1. Receive the ICE catalog from the Syndicator.
2. Process the catalog and evaluate the offer in terms of quantity and delivery policy. Select an offer that you wish to subscribe to.
3. Return a `<subscribe` message indicating your choice of offer and requesting the subscription to begin.

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
    <icemes:Sender name='bradsgadgets'

role='http://icestandard.org/ice/2.0/role/syndicator'
            sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
     <icesub:subscribe

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            offer-id='BradsGadgets04'>
      <icesub:offer

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
         offer-id='BradsGadgets04'
         name='BradsGadgetsNewsletter04'>
      <icesub:description>
```

```
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
            weekday='Monday'
            starttime='00:01:00'>
          <icesub:transport protocol='soap'
              packaging-style='ice'
              delivery-
endpoint='http://www.mysite.com/ice/'/>
         </icesub:delivery-rule>
       </icesub:delivery-policy>
      </icesub:offer>
   </icesub:subscribe>
  </env:Body>
</env:Envelope>
```

Note: In this recipe, the Subscriber must echo the offer inside the `<subscribe` message because they must communicate the delivery endpoint for a push delivery.

2.6.5 Syndicator Implementation Steps

Following the receipt of a `<subscribe` message from the Subscriber, the ball is back in the Syndicator's court. The Syndicator must process the subscribe message and decide whether to accept the subscription. In this example the subscription is accepted and the Syndicator must return a `<subscription` message. This message echoes the offer to which the Subscriber is subscribing to and provides a unique subscription identifier that will be used to track the subscription throughout its term.

1. Construct an ICE package containing the `<subscription` message:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
```

```
  <env:Body>
     <icesub:subscription

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            subscription-name='BradsGadgetsNewsletter04'
            subscription-id='BradsGadgets04kk3'/>
       <icesub:offer

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
         offer-id='BradsGadgets04'
         name='BradsGadgetsNewsletter04'>
       <icesub:description>
        Tech Tips from Brads Gadgets
       </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
             weekday='Monday'
             starttime='00:01:00'>
          <icesub:transport protocol='soap'
              packaging-style='ice'>
          <delivery-endpoint
url='http://www.mysite.com/ice/'/>
          </icesub:transport>
         </icesub:delivery-rule>
       </icesub:delivery-policy>
      </icesub:offer>
</icedel:subscription>
</env:Body><env:Envelope>
```

Note: In this message, the Syndicator provides the `subscription-id`. The `delivery endpoint` was provided by the Subscriber because this is a push delivery.

2.6.6 Subscriber Implementation Steps

1. Begin accepting SOAP deliveries of the ICE package from the Syndicator according to the <delivery-policy in the subscription message.

2.7 Add Security and Confirmation of Delivery

2.7.1 Background

In this recipe, the Syndicator pushes a catalog for valued content to the Subscriber as per business agreement. Subscriber subscribes to an offer for push delivery that requires confirmation of delivery. The Subscriber provides a delivery endpoint and a security login and password. Syndicator initiates subscription and begins pushing content to the Subscriber. Subscriber verifies content delivery. This recipe adds security and confirmation of delivery receipt capabilities. It is a Full ICE recipe.

2.7.2 Syndicator/Subscriber Business Agreement

When a syndication relationship is for valued content, a business agreement between the Syndicator and Subscriber is typically the first step of establishing a syndication relationship. In this discussion the type of content and the value of content is established.

2.7.3 Syndicator Implementation Steps

Following the establishment of a business agreement between the syndication partners, the catalog is constructed. In this recipe, the Syndicator will push the catalog to the Subscriber.

1. Construct a standard ICE package (SOAP/ICE/XML document) containing a catalog:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
<icedel:package
     xmlns:icedel='http://icestandard.org/ICE/V20/delivery'
     subscription-id='1'>
  <icedel:add>
    <icedel:item>
      <icesub:offer
xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
        offer-id='BradsGadgets04'
        name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
            weekday='Monday'
            starttime='00:01:00'
            confirmation='yes'>
          <icesub:transport protocol='soap'
             packaging-style='ice'/>
```

```
          </icesub:delivery-rule>
        </icesub:delivery-policy>
       </icesub:offer>
     </icedel:item>
   </icedel:add>
</icedel:package>
</env:Body><env:Envelope>
```

Note: In this example we have added parameters to the delivery rule that indicate that this catalog offers content from the first of January through the end of December in 2004. We are also indicating that we will push content to the Subscriber using SOAP each Monday beginning at 12:01:00 am. Finally we are indicating that the Subscriber must confirm content deliveries.

Pushing the catalog to the Subscriber is the first step for the Syndicator in this recipe.

2.7.4 Subscriber Implementation Steps

4. Receive the ICE catalog from the Syndicator.
5. Process the catalog and evaluate the offer in terms of quantity and delivery policy. Select an offer that you wish to subscribe to.
6. Return a `<subscribe` message indicating your choice of offer and requesting the subscription to begin.

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
    <icemes:Sender name='bradsgadgets'

role='http://icestandard.org/ice/2.0/role/syndicator'
            sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
     <icesub:subscribe

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            offer-id='BradsGadgets04'>
      <icesub:offer

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
```

```
         offer-id='BradsGadgets04'
         name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
            stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
            weekday='Monday'
            starttime='00:01:00'>
          <icesub:transport protocol='soap'
              packaging-style='ice'>
           <icesub:delivery-
endpoint='http://www.mysite.com/ice/'
              username='gadgets'
              password='brad007'/>
           </icesub:transport>
         </icesub:delivery-rule>
       </icesub:delivery-policy>
      </icesub:offer>
   </icesub:subscribe>
  </env:Body>
</env:Envelope>
```

Note: In this recipe, the Subscriber must echo the offer inside the `<subscribe` message because they must communicate the delivery endpoint for a push delivery. In this example, the Subscriber has added security of `username=` and `password=` on the delivery endpoint.

2.7.5 Syndicator Implementation Steps

Following the receipt of a `<subscribe` message from the Subscriber, the ball is back in the Syndicator's court. The Syndicator must process the `<subscribe` message and decide whether to accept the subscription. In this example the subscription is accepted and the Syndicator must return a `<subscription` message. This message echoes the offer to which the Subscriber is subscribing to and provides a unique subscription identifier that will be used to track the subscription throughout its term.

2. Construct an ICE package containing the `<subscription` message:

```
<env:Envelope xmlns:env='http://www.w3.org/2002/12/soap-
envelope'>
 <env:Header>
```

```
  <icemes:Header
xmlns:icemes='http://icestandard.org/ICE/Spec/V20/message'
timestamp='2003-12-03' message-id='m0056'>
   <icemes:Sender name='bradsgadgets'
role='http://icestandard.org/ice/2.0/role/syndicator'
sender-id='http://www.bradsgadgets.com'/>
      </ice:Header>
  </env:Header>
  <env:Body>
     <icesub:subscription

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
            subscription-name='BradsGadgetsNewsletter04'
            subscription-id='BradsGadgets04kk3'/>
<icesub:offer

xmlns:icesub='http://icestandard.org/ICE/V20/subscribe'
         offer-id='BradsGadgets04'
         name='BradsGadgetsNewsletter04'>
      <icesub:description>
        Tech Tips from Brads Gadgets
      </icesub:description>
       <icesub:delivery-policy start-date='2004-01-
01T00:00:00'
             stop-date='2004-12-31T24:00:00'>
         <icesub:delivery-rule mode='push'
             weekday='Monday'
             starttime='00:01:00'
             confirmation='yes'/>
          <icesub:transport protocol='soap'
              packaging-style='ice'>
           <icesub:delivery-
endpoint='http://www.mysite.com/ice/'
              username='gadgets'
              password='brad007'/>
          </icesub:transport>
         </icesub:delivery-rule>
       </icesub:delivery-policy>
      </icesub:offer>
</icedel:subscription>
</env:Body><env:Envelope>
```

Note: In this message, the Syndicator provides the `subscription-id`.The `delivery endpoint` was provided by the Subscriber because this is a push delivery, as were the security parameters.

2.7.6 Subscriber Implementation Steps

1. Begin accepting SOAP deliveries of the ICE package from the Syndicator according to the <delivery-policy in the subscription message.