HEDNA White Paper:

Change Discovery Service

EXECUTIVE SUMMARY

Change Discovery is a fairly new concept to the hotel distribution industry. The various implementations thus far have created a need to define, at a high level, what Change Discovery is and how it can help reduce volume and improve accuracy for hotel availability. The goals of this document are defining the concepts, use cases, issues, and benefits of Change Discovery Service (CDS) within the Hotel Electronic Distribution industry.

Change Discovery is a method of discovering and communicating the existence of change in rates and availability. Change Discovery allows a data publisher or consumer system to determine whether there has, or has not been, a change in rates or availability status for a given time frame. With Change Discovery, a data originator, data publisher and data consumer can communicate with small, efficient messages that provide a base for building targeted full availability requests for the change detail. This makes each availability message more useful, meaningful, and efficient.

Background

The HEDNA Connectivity Working Group (CWG) was formed following HEDNA's 2011 winter conference based on general agreement by the represented membership that the challenges of developing and maintaining multiple distribution interfaces in the absence of any real standardized implementation is a problem that the industry needs to address. The group is comprised of members representing suppliers, distributors, and intermediaries that are committed to achieving the goal of simplifying the way in which they interact with each other to facilitate the distribution of hotels through automated interfaces.

About the Author

- **Rachel Neal** is a Solutions Engineer for DerbySoft. She has worked in the hospitality industry since 1998 with a focus on central reservation systems, distribution systems and interfaces. She specializes in designing solutions that enable the un-connectable to connect.

Contributors

- **Tim Kieschnick** is the Vice President of Enterprise Applications for Pegasus Solutions, Inc. He has worked in the hospitality industry since 1997 focusing on high speed and high volume transaction processing for distribution systems and central reservation systems.
- **Lew Harasymiw** is the Director of Interface Solutions for Sabre Hospitality Solutions and sits on the OpenTravel Alliance Board of Directors. He has been working in technology since 1996 and the hospitality industry since 1999 in a variety of roles. He is currently responsible for the planning and implementation of strategic partnerships and automation with the industry's leading booking sites.
- **Craig Barnby** is a Director, Hospitality Solutions Architect for Orbitz Worldwide. He has worked in the hospitality industry since 1994 with a focus on central reservation systems, distribution systems and interfaces.

ROLES

To better document the passing of data and the process flow of CDS, we have defined three roles. In the current electronic distribution environment, companies may perform more than one of these roles concurrently, so in the interest of eliminating confusion, the traditional labels of Supplier, Intermediary, Distributor, Channel Manager, etc. will be supplanted with the following three labels.

- **Data Originator -** Within the context of CDS and its discrete chain of distribution, the Data Originator is any system where the data is first created or first appears. In practice, the Data Originator is usually a PMS or a CRS, or possibly a Channel Manager, and the data within the Data Originator is still within the control of the "owner" of the hospitality product, typically the hotel's owner or manager.

- **Data Publisher -** The Data Publisher is any system that accepts data and then re-publishes or redistributes that data. The Data Publisher may provide value-added services to the owner of the data, to the previous publisher of the data, and/or to the subsequent receiver of the data. Examples of these services include additional packaging, markups, etc. Typical Data Publishers include CRSs, Channel Managers, Intermediaries, Switches, GDSs, OTAs, and so on.

- **Data Consumer -** The Data Consumer is any system that accepts the data and uses it to enable the search for and/or the sale of hotel rooms, including booking transactions, traffic referrals, data analysis, etc. Examples of Data Consumers are OTAs, GDSs, search engines, bed banks, and so forth.
- Examples:
    - A hotelier that supports a direct connection to an online travel agent.
        - The hotelier is both a Data Originator and a Data Publisher
        - The online travel agent is a Data Consumer
    - An online travel agent distributes a hotelier's retail rates through a third party metasearch site.
        - The hotelier is a Data Originator and a Data Publisher.
        - The online travel agent is a Data Publisher.
        - The metasearch site is a Data Consumer.

STATEMENT OF PROBLEM

Data Consumers' needs for complete and accurate data strains Data Originators' and Data Publishers' systems as they attempt to provide real-time, accurate data that takes into account all of the relevant revenue management strategies, policies, and restrictions. A "pull" or shopping interface allows the Publisher/Originator to give an up-to-date response that takes all of these factors into consideration, but this requires the receiver of data to flood the publisher's system with requests, each request checking to see whether there has been any change to the previously retrieved data. This flood of transactions causes performance problems and higher operational expenses for both systems and can decrease the publisher's ability to be available to other data consumers. The majority of these transactions yield no change, and yet they are necessary to keep data fresh just in case there *is* a change. If a Data Originator or Publisher is able to determine rates and availability proactively, typically they can "push" this data to reduce the volume of "hits" or queries on their system. However, if a push is not possible, then a "pull" must be used, but if the Data Consumer chooses, for the sake of economy, not to query frequently enough, the data is inevitably inaccurate.

If the Data Consumer queries more frequently, the Publisher's system suffers. The Originators, Publishers and Consumers have had to either accept poor performance and/or high costs in order to maintain accuracy, or keep costs low and maintain performance by sacrificing accuracy. The issue lies in poor communication -- the consuming system does not know when a change has occurred, and the publishing system is unable to proactively determine what the rates and availability will be prior to the time of the request.

Change Discovery Service (CDS) is a solution developed to resolve this issue. CDS allows a consuming system to frequently "discover" whether something has changed and allows the publisher to respond without fully transacting an availability response. Once the existence of changes is known, the Data Consumer can then make a full availability request specifically for (and only for) data that is known to have

changed. This significantly increases the efficiency of the transaction set without sacrificing the accuracy of the data.

When using CDS, a Data Consumer can maintain within its own system an accurate representation of the availability status and rates from a Publisher, allowing for fast search and retrieval without needlessly hitting the Publisher's system with each query.

DETAILED PROBLEM STATEMENT

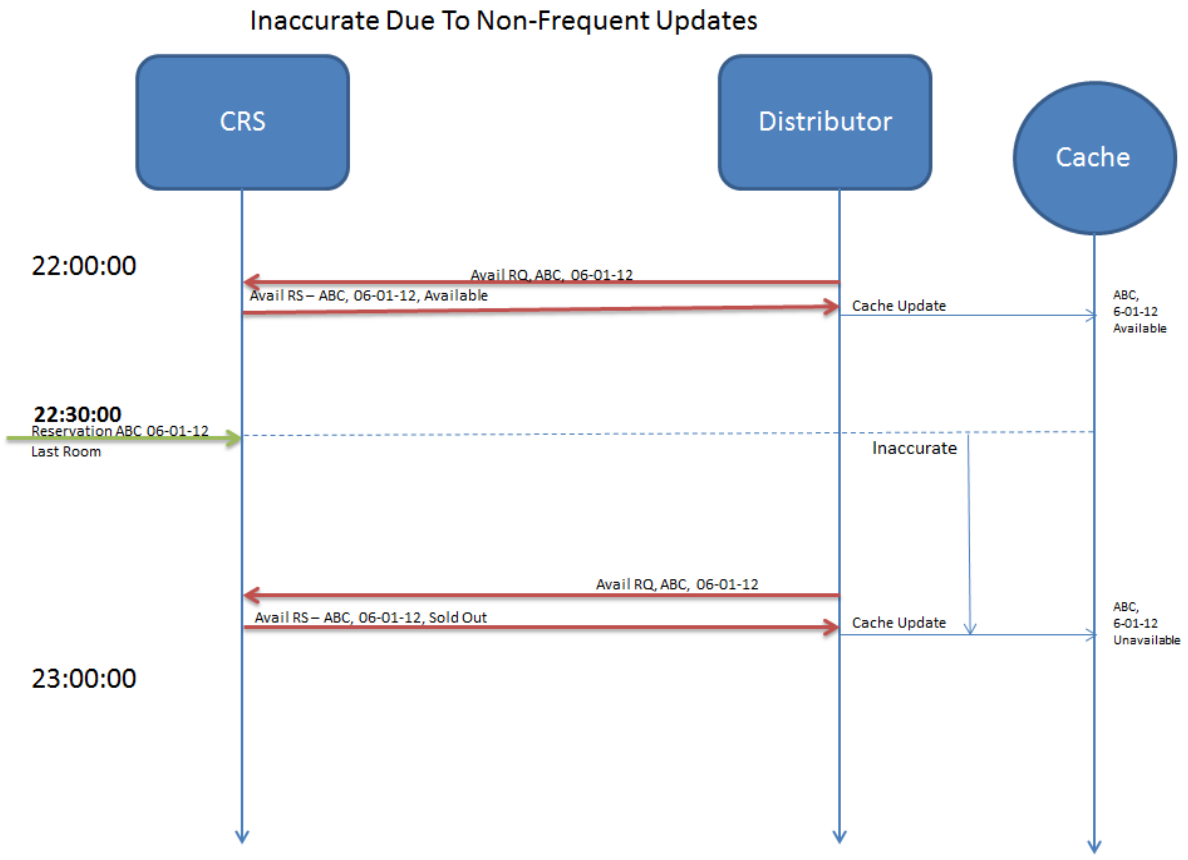
Industry - Accurate and timely ARI data from pull data publishers must be "shopped" with an immense volume of transactions incurring hardware, software, latency, and accuracy issues. Two problem conditions arise:

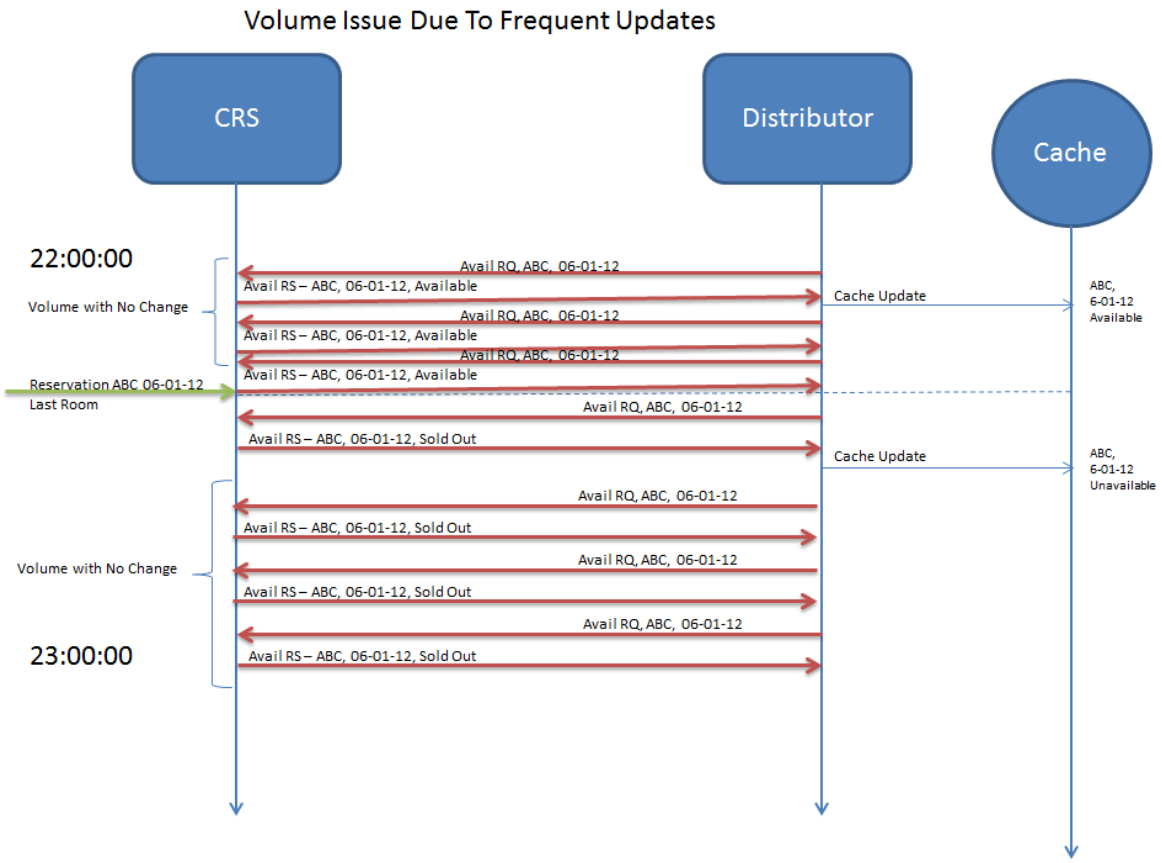
1. Inaccurate data to the Data Consumer due to infrequent updates. This can deliver incorrect pricing to the end consumer and put the hotels at risk for overselling.
2. Capacity, performance, cost issues due to very frequent updates.

Data Originators/Publishers - Publishers must support millions of inquiries per day from consuming systems due either to an inability to push, or to complex and detailed restrictions, pricing, and policy information that cannot be provided easily in a push transaction.

1. This volume will only increase in the future, and hardware/software limitations and high costs preclude solving for all issues involved.
2. Not all pricing, policies and restrictions are relevant for any extended time period making frequent updates a necessity for price accuracy and parity.

Intermediary Publishers and Consumers - The volume of requests needed to maintain accurate rates and availability with a pull originator/publisher incurs high cost and overhead while "wasting" most requests on unchanged data.





DESCRIPTION OF SOLUTION

Overview

CDS allows Consuming systems to query a Publisher's system with increased frequency to determine if any material changes have occurred to the availability/rates data that would require a refresh of the cache. The Publisher needs to maintain only a change or no-change status making the service extremely lightweight and cost effective. Upon receipt of a "change" response, the Data Consumer sends a targeted availability request message to the Publisher to get details of the change. Use of this service eliminates extraneous requests for no-change time periods.

Data Originator/Publisher

Originators and Publishers trigger an update to their CDS system whenever any change to availability or rates has occurred for a given time frame. This system does not need to maintain the details of this change, only the minimum data set necessary to form an availability request by a Data Consumer. Minimally this data includes Property ID and Change Date. CDS messaging *may* also include rate plan code, room type code, and whether the change was to rate or status.

Data Publisher/Consumer

Data Consumers can query a Publisher's CDS frequently (i.e. every 5-15 minutes) to ask for changes that occurred in that time period. If there is no change since the last query, no action is needed. If change is returned, a full availability request is built based on the data returned by CDS to retrieve the details from the Publisher. This allows the Consumer to make full queries to the Publisher only for data they know has changed.

Mechanisms for delivery of CDS messaging (push, pull, hybrid)
The mechanism for delivery and interaction between a Data Originator/Publisher and Data Consuming system depends on the ability of the Publisher to note change in its system, the level of detail at which that change is noted, and the data controls available to the Publisher and the Consumer. Rather than shopping a Publisher for all dates in order to cache hotels' rates, the Consumer wants to know whether the hotel made changes during a certain period.

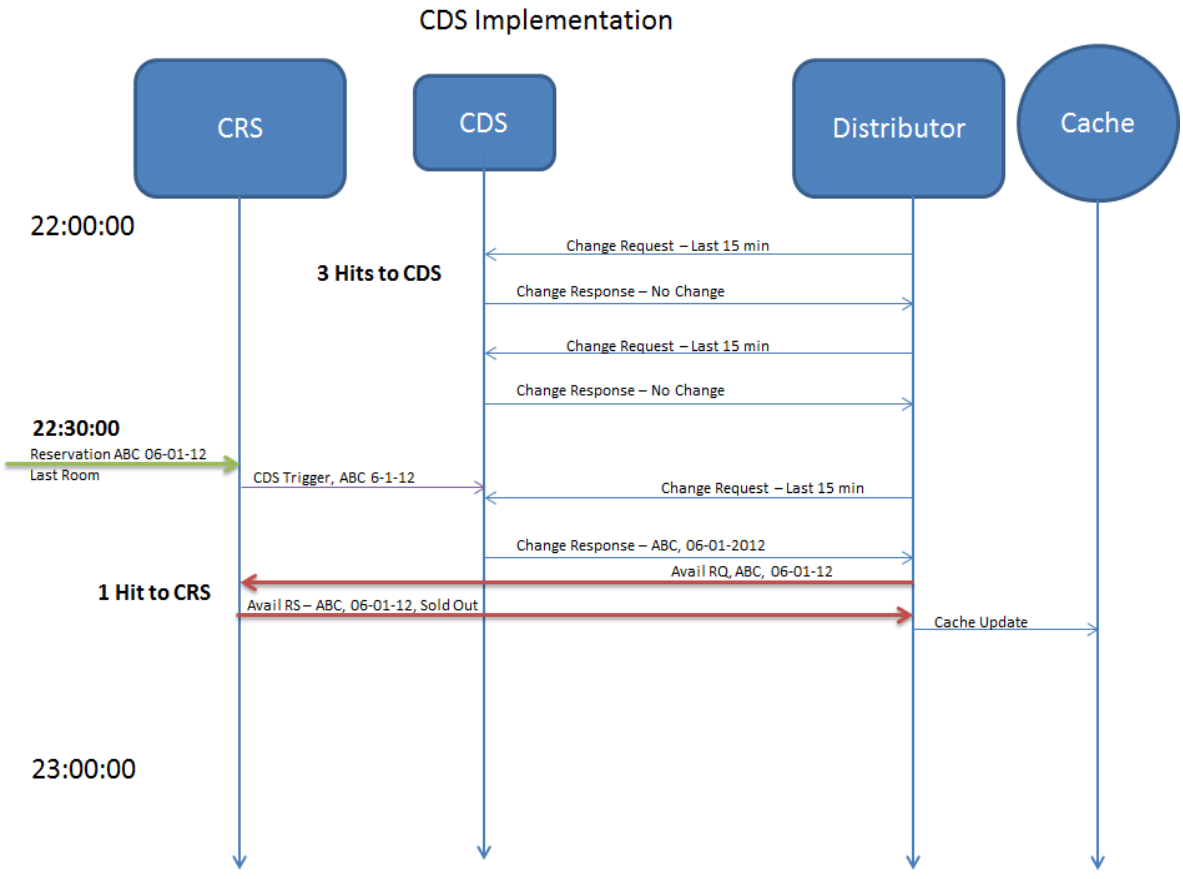
Push: If an Originator/Publisher is able to trigger an update when rates or availability has changed, they may have the capabilities to support a CDS push message or perhaps a full ARI push interface. Often a push interface is not feasible or cost effective for a publisher as it cannot reflect all the revenue management and policy enforcement necessary.

Pull: Many Publisher systems are unable to determine the exact nature of the change due to complex revenue management rules, complex data structures, or highly specific distribution rules per distribution system. In these cases, it is typically easier for the Publishing system to note when an event has occurred that could or would cause the availability and/or rates to change, and then add an entry to the CDS store for the Consumer to retrieve.

Hybrid: There are some data publishers and consumers that have created systems that use both push and pull to take advantage of the benefits of each model while mitigating some of the less desirable qualities associated with each.

The mechanism of delivery, though inherent to some of the issues, is not directly related to the definition and implementation of CDS. Therefore, the delivery mechanisms are only briefly defined and described above. Regardless of delivery mechanism, once change is communicated, the Consumer would only need to pull the specific changes relevant to its agreement and configuration with the publisher

Example Use Cases/Process Flow



EXAMPLE CDS TRANSACTIONS:

Implementation 1

Request from the data consumer:
This request is for all changes since “2012-05-02 12:32:30”

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<m: HotelChangeRequest xmlns:m=http://www.supplier.com/webservice
Token="E30ED3AA-65DE-48F9-BEA4-BA021B119625" UserName="test" Password="TEST">
<m: HotelChangeCriteria>
<m: HotelChangeCriterion HotelCode="54321" Timestamp="2012-05-02 12:32:30" Start="2012-05-01" End="2012-05-01"/>
<m: HotelChangeCriterion HotelCode="55621" Timestamp="2012-05-02 12:32:30" Start="2012-05-01" End="2012-05-01"/>
</m: HotelChangeCriteria >
</m: HotelChangeRequest >
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Response from publisher’s CDS:
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<m: HotelChangeResponse xmlns:m=" http://www.supplier.com/webservice " Token="E30ED3AA-65DE-48F9-BEA4-BA021B119625" Status="Successful">
<m: AvailabilityChanges >
<m: AvailabilityChange HotelCode="54321" Start="2012-05-01" End="2012-05-01" ChangeDateMask="00000101001010101010101010101010"/>
<m: AvailabilityChange HotelCode="55621" Start="2012-05-01" End="2012-05-01" ChangeDateMask="00000101001011000010101010101010"/>
</m: AvailabilityChanges >
</m: HotelChangeResponse >
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

Multi-Availability built off of CDS Response (if a supported transaction and only applicable if change dates are the same):
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<m:MultiAvailabilityRequest xmlns:m= http://www.supplier.com/webservice
Token="E30ED3AA-65DE-48F9-BEA4-BA021B119625" UserName="test" Password="TEST">
<m:MultiAvailabilityCriteria NumberOfUnits="1" >
<m:StayDateRange CheckIn="2012-05-06"
CheckOut="2012-05-07"/>
<m:GuestCount AdultCount="2" ChildCount="0"/>

```
                    <m:HotelCodes >
                        < m:HotelCode>54321< /m:HotelCode>
                        < m:HotelCode>55621< /m:HotelCode>
                    </m:HotelCodes >
                </m: MultiAvailabilityCriteria >
        </m: MultiAvailabilityRequest>
            </SOAP-ENV:Body>
    </SOAP-ENV:Envelope>
```

Single Availability built off of CDS Response:

```
    <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
                    <SOAP-ENV:Body>
        <m:AvailabilityRequest xmlns:m= http://www.supplier.com/webservice
            Token="E30ED3AA-65DE-48F9-BEA4-BA021B119625" UserName="test" Password="TEST">
                    <m:AvailabilityCriteria NumberOfUnits="1" HotelCode="54321">
                        <m:StayDateRange CheckIn="2012-05-06"
                    CheckOut="2012-05-07"/>
                        <m:GuestCount AdultCount="2" ChildCount="0"/>
                    </m:AvailabilityCriteria>
        </m:AvailabilityRequest>
            </SOAP-ENV:Body>
    </SOAP-ENV:Envelope>
```

Implementation 2

Request from data consumer:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <OTA_HotelAvailChangeRQ xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05
OTA_HotelAvailChangeRQ.xsd" TransactionIdentifier="9c7e03d73aab47e18a09491507bb8e7e"
Version="1.000" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opentravel.org/OTA/2003/05">
     <POS> <Source> <RequestorID Type="0"> <CompanyName/> </RequestorID> </Source> </POS>
<AvailChangeRequestSegments>
    <AvailChangeRequestSegment AvailReqType="Room">
    <StayDateRange End="2012-03-31" Start="2012-03-01"/>
    <HotelSearchCriteria ModifiedSince="2012-03-06T22:45:17.0Z" Rate="true" Availability="true">
    <Criterion> <HotelRef HotelCode="12345"/> </Criterion>
    <Criterion> <HotelRef HotelCode="67890"/> </Criterion>
    <Criterion> <HotelRef HotelCode="ABCDE"/> </Criterion>
    <Criterion> <HotelRef HotelCode="FGHIJ"/> </Criterion>
    </HotelSearchCriteria>
    </AvailChangeRequestSegment>
     </AvailChangeRequestSegments>
     </OTA_HotelAvailChangeRQ>
```

Response from publisher's CDS:

```
    <?xml version="1.0" encoding="UTF-8" standalone="no"?>
    <OTA_HotelAvailChangeRS xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.opentravel.org/OTA/2003/05"
xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 FS_OTA_HotelAvailChangeRS.xsd"  Version="1.000"
TransactionIdentifier="9c7e03d73aab47e18a09491507bb8e7e">
                <POS>          <Source>        <RequestorID Type="0"/>       </Source>          </POS>
```

```
                <AvailabilityChange IsChanged="true">
                <Availability HotelCode="12345">
                <TimeSpan Start="2012-03-01" End="2012-03-31"
ChangeDateMask="00000000000100000000000000000000"/> 30 day range starting with a date within requested stay range
                </Availability>
                <Availability HotelCode="ABCDE">
                <TimeSpan Start="2012-03-01" End="2012-03-31"
ChangeDateMask="00000011111110000000000000000000"/>
                </Availability>
                <Availability HotelCode="FGHIJ">
                <TimeSpan Start="2012-03-01" End="2012-03-31"
ChangeDateMask="00000000000000000000000000000000"/>  NO CHANGE
                </Availability>
                </AvailabilityChange>
                <Success/>
        </OTA_HotelAvailChangeRS>
```

DEFINITIONS

- **ARI -** Industry standard term for the type of data exchanged in an online distribution environment. Typically used within the context of a “push” model. Stands for Availability (aka restrictions, e.g. close, minimum length of stay etc.), Rates (pricing of room/rates) and Inventory (count of rooms available for sale)
- **CDS -** Change Discovery Service: A small transaction set that allows a distribution system to determine whether there has or has not been a change in rates or availability status for a given time frame.
- **Change Date -** The date *for* which rates or availability have changed, not the date *on* which changes were made. Example: today, the rates for 1st June changed, therefore the Change Date is 1st June , not today.
- **CRS -** Central Reservation System: A system containing information about availability, rates, and related services, and through which reservations can be made. The term used within this document refers to a hotel company's central reservation system.
- **Distribution System -** Any system transacting directly with consumers or distributing availability, rates, inventory and/or reservations for the purpose of booking hotel room reservations. For this document, this will include all intermediary, OTA and consolidator systems.
- **PMS** - Property Management System: A reservations and inventory system implemented at the property level that enables the property to check in and check out guests, manage rates and availability, etc.
- **Pull -** A mechanism of data delivery where the data recipient initiates a request for data which is then returned by the data originator
- **Push -** A mechanism of data delivery where the data originator initiates the delivery, “pushing” it to the data recipient who passively accepts the transaction
- **Supplier -** Any hotel, chain, or reservation service provider that transacts with Distribution Systems