IBM Cúram Social Program Management

Curam Wait List Developer Guide

Version 6.0.4

Note

Before using this information and the product it supports, read the information in Notices at the back of this guide.

This edition applies to version 6.0.4 of IBM Cúram Social Program Management and all subsequent releases and modifications unless otherwise indicated in new editions.

Table of Contents

Chapter 1

Introduction

1.1 Purpose

The purpose of this document is to explain to the reader how to design, implement, customize and configure Cúram Wait Lists. In order to best understand these concepts, the guide should be read in full.

1.2 Audience

This document is intended for designers and developers who need to know how to extend Cúram Wait Lists to meet their specific requirements. A reasonable degree of knowledge of the application is assumed. There are certain aspects of the document that may prove useful to business analysts, which do not require technical knowledge.

1.3 Prerequisites

It is assumed that the reader is familiar with the basic concepts of Social Enterprise Management (SEM) and the following concepts

- Persistence Cookbook
- Cúram Batch Processing Guide.

Chapter 2

Why Customize a Wait List

2.1 What is a Wait List?

A wait list is a list of clients who are awaiting the availability of a specified resource. The resources in question are usually required for the delivery of a service, such as funds or the availability of a place in a facility. Where the resource isn't available, a client can be added to wait list, either by allocating a certain position, or adding the client to the end of the list. Once added, a client's position on the wait list can be moved up or down, depending on how pressing the client's need for the resource is.

2.2 Why Customize a Wait List?

Cúram Wait Lists can be applied to any of the industry segments within the SEM model. Social Enterprise agencies servicing these industry segments provide a diverse array of benefits and services to their clients. These services may have different funding sources, and some may be contracted out to third party providers; there are therefore a wide range of ways in which wait lists could be used.

As a result of this, the intention of the Cúram Wait List functionality is to provide a lightweight, generic implementation, which can be used to meet as many of these needs as possible. Hence customizing the wait list to suit the various needs of SEM agencies becomes vital. To accommodate this, the Cúram Wait List includes extension points and customizable interfaces through which the default behaviour can be changed.

2.3 Code Package

The Cúram Wait List is implemented using the Persistence Infrastructure approach. This is placed under the following package:

- curam.waitlist.impl

Chapter 3

Customization Points

3.1 Replaceable Implementations

3.1.1 Introduction

Persistence Infrastructure makes use of Google Guice to act as a factory mechanism for interface objects. In general, it is not compliant to change an implementation which is bound to an interface. Note that if an implementation is replaced in a non-compliant way, difficulties may be encountered in a later upgrade to a newer version.

However, wait list does provide an interface which allows the customer to replace the default implementation with a new custom implementation, if required, by creating a new Guice module class and adding a corresponding entry in the MODULE table. For more information on Guice and how to use Guice bindings using a Module class, please refer to the Persistence Cookbook.

3.1.2 Renumbering of Wait List entries

Interface location

curam.waitlist.impl.WaitListRenumberLogic

Purpose

The agency may wish to renumber the wait list entries considering the priority of the entry, or on first come first served basis, or using some other criteria.

Default implementation

The default implementation renumbers wait list entries in a wait list. The wait list entry position is incremented or decremented if there is an insertion or removal or allocation or expiry of an entry in the list.

When creating a new wait list entry system increments the position by 1 for all the wait list entries in an 'open' state with a position higher than or equal to the position of the entry to be added to the wait list.

When a wait list entry is canceled or expired, system decrements the position by 1 for all wait list entries in an 'Open' state with a position higher than the position of the entry being removed or expired from the wait list.

The usage of this implementation in operations Create/Modify/Cancel/Expire/Allocate a Wait List Entry in a Wait List has been explained below with example.

Create Wait List Entry: When creating a new wait list entry, if a wait list requires renumbering then system calls wait list renumbering API. Wait list renumbering API increments the position by 1 for all 'Open' state wait list entries with a position higher than or equal to the position of the entry to be added to the wait list. After renumbering the existing wait list entries, system adds the client (i.e. a wait list entry) to the wait list as per the specified position.

For example, if there are 10 clients waiting for a resource with position from 1 to 10. If a new wait list entry to be created at position 5. System will first try to empty the position 5 by incrementing position by 1 for all entries which are greater than or equal to 5 by using wait list renumbering API and then insert the new record with position 5.

Cancel/Expire/Allocate a Wait List Entry: After cancelling/Expiring/Allocating a wait list entry, if a wait list requires renumbering then system calls wait list renumbering API. The wait list renumbering API decrements the position by 1 for all 'Open' state wait list entries with a position higher than the position of the removed entry from the wait list.

For example, if there are 10 clients waiting for a resource with position from 1 to 10.If a wait list entry (i.e. a client) at position 5 is removed from the wait list, then system will decrement position by 1 for all entries which are having the position greater than 5 by using wait list renumbering API.

Update Wait List Entry: In case of update wait list entry, system first checks whether there is change in the position. If yes then check whether position renumbering is required. If renumbering is required, then system calls wait list renumbering API to reshuffle the wait list entries position.

For example, if there are 10 clients waiting for a resource with position from 1 to 10. When modifying the wait list entry at position 6, if user changed the position value from 6 to 8 then system first decrements the position by 1 for all 'Open' state wait list entries with a position higher than the original position (i.e.6) and then increment the position by 1 for all 'Open' state wait list entries with a position higher than or equal to the newly specified position (i.e.8). This creates an empty block so that our modified wait list entry goes and gets placed there.

Note

For more information, on how to provide a compliant implementation for an interface, see the section 'How to provider a compliant implementation for an interface that is marked as an extension point within the framework' in Cúram Provider Management™ Developers Guide.

3.2 Events

3.2.1 Introduction

Following events are raised by wait list to allow the customer to write and attach custom workflows.

3.2.2 Event on Wait List Entry Creation

An event is raised when a wait list entry is added to a wait list.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYCREATED	waitListEntryID	curam.waitlist.impl.WaitListEntryImpl.createWaitListEntry

Table 3.1 Event on Wait List Entry Creation

3.2.3 Event on Wait List Entry Update

An event is raised when a wait list entry is updated.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYUPDATED	waitListEntryID	curam.waitlist.impl.WaitListEntryImpl.modifyWaitListEntry

Table 3.2 Event on Wait List Entry Update

3.2.4 Event on Wait List Entry Allocation

An event is raised when a wait list entry is allocated.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYALLOCATED	waitListEntryID	curam.waitlist.impl.WaitListEntryImpl.allocate

Table 3.3 Event on Wait List Entry Allocation

3.2.5 Event on Wait List Entry Removal

An event is raised when a wait list entry is deleted.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYREMOVED	waitListEntryID	curam.waitlist.impl.WaitListEntryImpl.cancel

Table 3.4 Event on Wait List Entry Removal

3.2.6 Event on Wait List Entry Expiration

An event is raised when a wait list entry is expired.Wait list entry is expired when the expiry date on the wait list entry is crossed. The wait list entry can be primarily expired by using expire wait list entry batch job.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYEXPIRED	waitListEntryID	curam.waitlist.impl.WaitListEntryImpl.expire

Table 3.5 Event on Wait List Entry Expiration

3.2.7 Event on Wait List Entry Review

An event is raised when a wait list entry is selected for review. A wait list entry can be selected for review by running the WaitListReview batch job. This will result in all wait list entries with a review date on or before the batch processing date being selected for review.

Event Detail	Primary Event Data	Raised From
WAITLIST.WAITLISTENTRYSELEC	waitListEntryID	curam.core.sl.impl.WaitListReview.processWaitListEntrie

Event Detail	Primary Event Data	Raised From
TEDFORREVIEW		sDueForReview

Table 3.6 Event on Wait List Entry Review

3.2.8 Event on Wait List Entry Deferring

An event is raised when the Wait List Entry review is deferred to a future date. This event is also raised when the review date is updated to be a date in the future.

Event Detail	Primary Event Data	Raised From
WAITL-IST.WAITLISTENTRYREVIEWDEFERRED	waitList-EntryID	curam.waitlist.impl.WaitListEntryImpl.deferReview

Table 3.7 Event on Wait List Entry Deferring

Chapter 4

Implementing a new Wait List

4.1 Implementing a new Wait List

A customer who wishes to create their own type of wait list will need administratively to add a new type of wait list. This can be done by adding a custom entry in the WaitListType codetable corresponding to the new wait list type or resource type.

The new resource will need to extend the interface curam.waitlist.impl.Resource and the corresponding implementation class needs to implement the methods defined in the Resource interface. This ensures that the new resource has the necessary methods implemented which are required during creation of a wait list.

The code snippet of the interface which extends the Resource interface:

```
public interface Provider extends ProviderOrganization,
    Insertable, OptimisticLockModifiable, Resource {
    // method declaration goes here
    }
```

The code snippet of the implementation which implements the Resource interface:

```
public class ProviderImpl extends
        ProviderOrganizationImpl<ProviderDtls< implements Provider {
          .
          .
            public long getResourceID() {
              return getDtls().providerConcernRoleID;
            }
          .
          .
            public WAITLISTTYPEEntry getResourceType() {
              return WAITLISTTYPEEntry.PROVIDER;
            }
        }
```

The necessary APIs to add/modify/cancel a wait list entry, list the history of wait list entries or search a wait list can be obtained in the corresponding Javadoc provided along with the Cúram Enterprise Framework™ .

Chapter 5

Integration with Other Systems

5.1 Introduction

The Cúram Wait List is a generic functionality which holds the clients waiting for a resource to be available; this can be integrated with other systems. Examples of wait lists being integrated with other systems are described below.

5.2 Integration with Cúram Provider Management

Wait listed clients in Cúram Provider Management (CPM) wait for a provider or provider offering (service), which can be either a placement service or a non-placement service. For a placement service, the available places can be searched for within the required time period. Once it is determined that a place is available for the client, a reservation or placement can be created. If no place has been specified for a wait listed client, a place from a list of available places can be selected when creating the reservation or placement. There is no OOTB automated allocation of clients when a place becomes available.

Clients can be wait listed for multiple providers, but once a reservation or placement is made, wait list entries for that client for the other providers can be removed. If there are active reservations for the same or overlapping periods, the reservations will be canceled when a place is allocated to a wait listed client.

5.3 Integration with Cúram Funded Program Management

Cúram Funded Program Management™ (FundedPM) wait lists allow clients to be placed on a wait list when funds are insufficient. For example, when a

client is authorized to receive child care services and in turn, an attempt is made to obligate an amount from the Child Care fund, the client can be placed on a wait list if the child care fund does not have a sufficient amount to cover the obligation.

When funds become available, the request(s) to obligate funds on behalf of the client for the child care case plan items can be reassessed, and the obligation associated with the fund can be processed.

Chapter 6

Wait List Batch Jobs

6.1 Introduction

Wait list functionality is provided with following batch jobs.

6.2 Expire Wait List Entry

Expire Wait List Entry expires any wait list entry for which the expiration date has passed.

Batch Name	Implementation Class
ExpireWaitListEntry	curam.core.impl.ExpireWaitListEntry

Table 6.1 Wait List Expiration Batch Job

6.3 Review Wait List Entry

This batch process selects the eligible wait list entries for review and raises a workflow event to generate wait list review reminder task. All wait list entries with a review date on or before the batch processing date will have a review reminder task generated. Review date is provided while creating a wait list entry. If there is no review date set for the wait list entry, it is derived by subtracting the Status Review Reminder Period from the expiry date. Status Review Reminder Period is configured in property administration.

Batch Name	Implementation Class
WaitListReview	curam.core.sl.impl.WaitListReview

Table 6.2 Wait List Review Batch Job

6.3.1 Batch Launcher configurations for Wait List Review batch job

Since Wait List Review batch process raises work flow events, for successful generation of review reminder tasks, following batch launcher properties should be configured in property administration:

- curam.batchlauncher.dbtojms.enabled (should be set to true)
- curam.batchlauncher.dbtojms.notification.host (should have appropriate host value set)
- curam.batchlauncher.dbtojms.notification.port (should have appropriate port value set)

Please refer to the Cúram Batch Processing Guide for more information.

Chapter 7

Wait List Application Properties

7.1 Introduction

Following wait list properties are provided for configuration in property administration.

7.2 Configuration of Wait List Review properties

7.2.1 curam.waitlist.statusreviewreminderperiod

This property determines the reminder period for the wait list status review. It is subtracted from the expiry date to get the review date when review date is not specified. Default value is set as -1 in which case it is not considered.

7.2.2 curam.waitlist.statusreviewintervalperiod

This property determines the number of days between two wait list entry reviews. Default value is set as zero.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

1623-14, Shimotsuruma, Yamato-shi

Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typograph-

ical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources.

IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products

should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This publication documents intended programming interfaces that allow the customer to write programs to obtain the services of IBM Cúram Social Pogram Management.

Trademarks