C. Dudley
Xaraya Development Group

Publish and Subscribe Module

Status of this Memo

This memo provides information for the Xaraya community. It does not specify an Xaraya standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The contents of this RFC contain the literal content of the old plain text version of RFC-0001

When time is a less scarcer good, someone might convert the plain text into structured XML so we can benefit from it.

Table of Contents

1. Introduction

The following RFC is a first summary and describes some solution proposals based on my own thoughts and discussions with assorted groups.

2. List of requirements for the Publish and Subscribe Engine

The publish and subscribe engine covers two different but related goals:

- Allow users to be able to set the system to automatically notify them when certain events happen.

- Allow admins to push information out to users when something on the site changes.

This notification would typically be an email but there is no reason why it should be limited to emails. A future improvement might be to allow users to request SMS notification instead of an email. For now the only medium will be email. However a user should be able to setup their preferences so that they can receive emails in either HTML or plaintext form. If the admin enables it then weekly digests could also be produced, but this would entail some form of scheduling agent (not sure of a platform independant way of doing that). The HTML email would be based on a theme layout specified by the admin for the site. All text would be stored in lang files so that the emails would be multi-national. The admin should be able to choose to post information out to users when posting and article or a new download (for example). They should be able to choose an individual user, several users or all registered users. There are spam implications with this blanket mailing idea and so some way of setting it up so that the use of it is only really used for emergencies like security bulletins.

Examples of the kind of events that could trigger email notification:

- new story in a particular topic added

- new web link

- new download in specific category

- reply to a forum thread

Users should be able to subscribe to an event by clicking on a small "monitor" icon next to that item similar to how forum and filelist monitoring works on sourceforge. However they should also be able to edit their subscription selections from a central page in their own user setings.

The subscription side is going to entail most of the work - simply allowing admins to email users when they fill in the form to post a new article or download is a minor change by comparison.

3. Solution proposals - database tables

Am not entirely sure how to structure the database tables or really even which are needed.

```
    Available events (pubsub_events)
    ----------------
        eventid        unique event identifier
        modid            holds a reference to the module
        cid        category ID
        iid        item ID
        groupdescr    defines a group of items in the module (e.g. category),
semantics are defined by the module

    Event registration (pubsub_reg)
        ------------------
        pubsubid    unique subscription identifier
        eventid        event identifier
        userid        user registered for this event
        actionid    requested action for that user+event (only mail for v1)

    Event handling (pubsub_process)
        --------------
    handlingid    unique identifier for handling a particular event for a
particular item and a particular user
        pubsubid    subscription identifier (= defines event + user + action)
        objectid    references an individual item in the module, semantics are
defined by the module
        status        pending / done

    Event Template (pubsub_template)
    --------------
    templateid    unique identifier for this template
    eventid        event identifier
    template    template text
```

Available events are created based on available hooks and/or other events defined by the module (e.g. publication of an article in a particular category). Event registration is done by users, signing up for a particular event (and specifying a particular action in the future). Event handling is filled in when an event is triggered for a particular item (e.g. via hooks), listing all user subscriptions that are affected by this event, the particular object ID for which action should be taken, and the status of that action. The actual handling of the actions for each user can then be done in background, or scheduled for later processing, or at the time the event was triggered (to be further defined).

4. Solution proposals - functions

xarPubsubAddEvent - add a new event

xarPubsubEditEvent - edit the event (to add or remove a new user for example)

xarPubsubDelEvent - delete the specified event

xarPubsubGetEvent - return the specified event, when only given a module name will return all events for that module for example. or if given an event id will return a specific event.

xarPubsubSearchEvents - search all events for matches (e.g. all events for a specified user or module)

xarPubsubGetUsers - return the list of users subscribed to a specific event

xarPubsubGetNextUser - return next user subscribed to a specific event

xarPubsubGetPrevUser - return previous user subscribed to a specific event

xarPubsubNotifyUser - actually send the notification

xarPubsubGetNotifyFormat - to retrieve the format in which a user has requested receiving the notification (e.g. HTML or plaintext)

xarPubsubGetTemplate - get the template for this event

xarPubsubDelTemplate - delete the template for this event

xarPubsubUpdateTemplate - update the template for this event

questions:

should adding/removing a user from an event have specific function when all it is is a database update of same row as rest of event info?

do i need functions to handle creation of a list of available users for the article post form for example so admins can choose to mail users?

do i need any functions to allow users to choose their notification format or for admins to specify the email template? not sure how digest form emails could be handled - ideas?

5. Relationship of the code to other areas

Since this is a utility module it would be up to each of the other modules to implement this functionality or not. The user and admin settings will have some impact on their respective modules (or rather the need for the user and admin to be able to specify the settings will)

erm what have i forgotten?

Create hooks could be used to provide basic functionality without impact on individual module code. The pubsub user interface could then list all modules that are "hooked up" to it, and the user could select which "create" events he'd like to be informed of...

6. Code that will need to be rewritten

Any modules that want to implement this functionality will need to be amended to take account of it. some way of those modules being able to behave appropriately whether the pubsub module is installed or not would be good. The monitor icon could then only be displayed when the module is installed.

7. Basic documents for this RFC

8. Tools that need to be created from scratch

9. Current issues

need to work out the queuing system - unsure how to do it in PHP.

10. Retractions

11. Changelog

0.1 initial draft

0.2 (mikespub) revise point 3 and 5 - table design and use of hooks

0.3 (miko) added template functions and table