# Overview

The ActiveKids project is designed to provide a portal application for all of Active’s children-related offerings, including ANet activities. The flow will be something like this:

* A parent will search for activities through the AK UI, with data provided by assets uploaded to the asset service by various Active applications, including ANet.
* When the parent clicks the register button, they will first be logged into Active Passport, creating a new person record if necessary.
* For the parent, if no ANet customer is associated with the AK person, a redirect to ANet will present a login / create account UI for that purpose.

Then, in the initial (Q1) implementation:

* ANet will continue through the entire registration and payment workflows, only redirecting to AK when payment processing is complete.

For a later (Q2+) release, more of the workflow will be implemented in AK:

* For the registrant (child), if no ANet customer is associated with the AK person, a redirect to ANet will present a select / create family member UI for this purpose.
* Once an ANet activity is selected, and the parent and registrant are mapped, a streamlined workflow in the AK application will enroll the child into an activity, via ANet web services, and take payment.
* If the registration workflow for the activity is too complex for AK, AK will redirect to ANet to present the registration workflow; when it is complete, ANet will redirect to AK to display the cart, optionally add additional registrations, and take payment.

This document provides an overview of the proposed integration.

# Mapping of AK persons to ANet customers

A key element of the design is mapping AK person\_id’s to ANet customer\_id’s. This mapping will be implemented on the ANet side as follows:

* ANet will store a person\_id in the customer record at the completion of workflows provided to select or create an ANet customer for an AK person.
* To allow AK to use existing ANet web service methods which require a customer\_id, a new web service method will be provided to allow AK to query ANet to determine the customer\_id associated with its person\_id.
* As a side-effect, by storing the person\_id with the ANet customer, it can be pushed downstream to the EDM (enterprise data mart) and used to support an enterprise view of transactions by a parent, or registrations by a child.

# Q1 release workflow

## Full workflow in ANet CUI

In the first use case, for the Q1 release, an activity and a person are selected in AK. AK then redirects to the ANet CUI, which does the full workflow through payment processing, and only redirects to AK after the payment confirmation page is displayed:



When the parent clicks on the “register now” button, the AK registration workflow begins:

**1, 2. If user is not already logged into Active, present login page:** If AK determines the current user has not logged in yet, a login / account creation page will be displayed in AK.

**3. Redirect to Anet:** AK will display a UI telling the user they are now going to the “organizer’s site”, then redirect to a new ANet CUI page designed for this purpose, providing the following information:

* The activity\_id
* The person\_id
* Personal information about that person which can be used to prefill an account creation form if necessary (JSON)
* URL to redirect to upon completion
* A token identifying the request is really coming from AK

Because of the data provided, this will need to be a secure redirect via a POST.

ANet will first parse the POST parameters; if the AK token is invalid, they will be discarded, and no special behavior will occur.

**4. Is person\_id already mapped?:** If ANet has a mapping for the person\_id already, that customer\_id will be logged in, and the flow will skip to the registration flow.

**5. Prompt for login or account creation:** At this point, ANet will display its login page, which also allows creation of a new account. Note that this will happen even if the user is already logged into the CUI.

**6. If logged into existing account:** If the user logged in as an existing customer, the flow will skip ahead to save the person\_id into that customer.

**7. Present account creation workflow:** If the user chooses to create a new account, an account creation page will be presented, filled in with the information provided by AK. Upon submission, it will be validated by the rules configured by that organization, as the data provided by AK may not be complete.

**8. Save person\_id:** Once a customer\_id has been established by any of the above paths, ANet will save the person ID into that customer record.

**9. ANet registration flow:** If the login page determines an activity\_id has been passed in, it will proceed immediately to the first registration page. The workflow will continue normally in the CUI, until the order is complete and payment processed.

**10. Redirect to AK:** When the payment confirmation page is displayed, if a redirect URL was passed in, ANet will redirect back to AK, which will display some page.

## Transaction history request

One requirement is that AK be able to display an enrollment history for the parent. The preferred implementation is that AK will get this data from ADP, by querying on the person\_id which ANet has recorded with the customer. This will automatically returns data for all ANet organizations which have customers linked to that person\_id.

If the ADP support will be too late, ANet will provide a WS method to allow AK to query for the transaction history. The WS method will be fairly simple to implement, but it will require that AK know which orgs ANet it’s requested enrollments in, so it can call the method for just those orgs.

# Q2+ workflows

In the next major release, it is intended for AK to present the registration workflow, at least in the standard case, using ANet web services. New workflows to support this are:

* After the parent has been logged into AK, it will redirect to ANet if necessary to login/create an account, and return a customer\_id.
* Because family members must also have customer\_id’s to be registered through the ANet web services, a second workflow will redirect to ANet if necessary to select/create a family member, and return this customer\_id.
* Finally, since some activities may have too complex a registration workflow for AK to handle natively, a workflow is provided for AK to redirect to ANet for the actual registration flow, returning to AK to display the cart, possibly add additional registrations, and take payment.

## CUI workflow design considerations

The model below is based on AK redirecting to the CUI for certain very specific purposes, which will then redirect to a provided URL when complete. In order to fit in AK and prevent the user from being confused, we assume that:

* The user should be prevented from doing anything in the CUI beyond the specific workflow. This means that the CUI navigation will need to be removed.
* The overall page styling will also need to be modified to fit with AK.

## 1. Link parent workflow

The first use case is the initial SSO login, in which ANet does not have a mapping for the AK person\_id, and redirects to ANet for an account login or creation:



When the parent clicks on the “register now” button, the AK registration workflow begins:

**1. Establish person\_id for parent:** The first part of the flow, to login / create account for the parent, is unchanged from Q1.

**2. Determine if there’s an existing customer mapped to the person:** AK call a new ANet web service method getPersonCustomerID to get the customer\_id associated with the person\_id. If there is no existing mapping, the method will return blank.

**3. If already mapped proceed to AK registration flow:** If ANet returned a customer\_id, AK will not redirect to ANet, but will proceed in its registration flow.

**4. If not already mapped, redirect to ANet for login / account creation:** The first part of the flow, for account creation and login, is identical to the Q1 flow. However, in this case, the redirect will not provide an activity\_id. This will direct ANet to not proceed to the registration and payment flows, but to return after the login / account creation is complete.

**5. Redirect to AK when login / account creation is complete:** After the parent login / account creation is complete, ANet will redirect to AK to provide the registration flow. To save AK another call to getPersonCustomerID, the redirect will return the ANet customer\_id

**Failure case:** If the link parent UI design allows for any cancel action, ANet must redirect back to AK without a customer\_id, and AK must detect/act on this state. (Not represented in the workflow diagram.)

## 2. Link family member workflow

The above process links the parent (the payer) in AK and ANet. In the typical case, however, the parent will be enrolling a family member (the registrant) in an activity. For this purpose, we also need to have the AK person for the registrant linked to an ANet customer.

However, the ANet parent may have existing family members, so we would like to link the AK registrant to an existing ANet family member if they exist; otherwise, to create a new family member for the registrant. In either case, the selected ANet registrant will be linked to the AK registrant, just as with the payer.

This will require some kind of UI listing existing family members and providing the ability to either link an existing member, or create a new one. The design of a simple, understandable UI for this purpose will not be simple, and this document will not address that problem, but only identify where the UI is in the workflow. ANet also allows the parent to be in multiple families, so this makes the design more difficult as well. It should be possible to partially automate the process by matching names and selecting a suggested match.



**1. Select / create the registrant in AK:** The initial condition of this workflow is that the registrant has been either selected or newly created in AK, and therefore, a person\_id has been established.

**2. Determine whether registrant is already linked to ANet:** As in the link parent workflow, AK will call the getPersonCustomerID method to get the customer\_id linked to this person, if any.

**3. If already linked, proceed to registration flow.**

**4. If not linked, authenticate with ANet using web services:** Before AK redirects to ANet, it must first use its own information to authenticate the parent with ANet, so the ANet CUI can bypass its login.

This is normally done by calling the method getCustomerLogonUUID, which takes a login name (or equivalent) and a customer password as input, and returns a UUID as a logon token which can be passed to the CUI. This is fine if another application is providing an ANet login UI, but won’t work in this case. Even if we provided a mechanism for passing the ANet password back to AK, that password could later be changed by the user.

We need a login mechanism with information which cannot be changed by the user in ANet. What is proposed here is a new method, getPersonLogonUUID, which takes the customer\_id and the person\_id as inputs, and returns the same UUID.

**5. Redirect to CUI page:** The redirect will have to provide the following information:

* Logon UUID of the parent
* The person\_id of the registrant
* Available information about the family member in AK
* URL to redirect to upon completion

Because of the data provided, this will need to be a secure redirect via a POST.

**6. Login customer:** ANet will use the processCustomerLoginWithUUID WS method to login the parent.

**7. UI to link or add family members:** At this point, ANet would display the UI allowing the user to select an existing ANet family member of the parent, or to add the registrant to a family.

**8. Save person\_id:** ANet will also save the person\_id provided for the registrant.

**9. Redirect to AK:** ANet will redirect to the AK, returning the customer\_id.

**Failure case:** If the link family member UI design allows for any cancel action, ANet must redirect back to AK without a customer\_id, and AK must detect/act on this state. (Not represented in the workflow.)

## 3. Redirect to CUI for registration flow

In the AK design, the intention is to provide an entirely new user experience for the typical use case, with minimal use of the ANet CUI. The designs above assume the CUI is used for linked persons to customers as necessary, but that AK provides the UI for the remaining registration flow.

However, ANet has many variations of activity registration workflows, such as for different models of team registration, package registration, etc. It is expected that the initial version of AK will not address these more complex cases, and some cases may never be addressed.

In this case, it will be necessary for AK to redirect to the ANet CUI for the registration flow up to the point where the registration has been added to the cart.



**1. Prior workflow:** At this point, it is assumed that the following have been resolved:

* An activity has been selected, so AK has an ANet activity\_id.
* The parent has been resolved through SSO as above, so AK has an ANet customer\_id for the parent.
* The child has been resolved as above, so AK has an ANet customer\_id for the child.

**2. Determine what workflow features are required:** In some way, AK must determine whether it can handle the complexity of the registration flow for this activity. This level of detail is not provided in the existing wsActivity object. What is proposed here is a new API, getRegistrationWorkflowFeatures, taking the activity\_id as input. Since initial analysis may not identify all possible variables, this would be kept generic by returning a set of keyword/value pairs describing attributes of the workflow; e.g., whether it is a team activity, an “activity package”, etc.. AK could use those values to determine whether it can handle the workflow. Later more competent AK releases could handle more cases, without needing to be synchronized with ANet releases.

**3. If not too complex, continue to standard AK workflow.**

**4. Authenticate with ANet using web services:** Before redirecting to ANet, AK will login the parent to the CUI using the getPersonLogonUUID method, as in the family member flow above.

**5. Redirect to ANet CUI registration page:** At this point, AK will redirect to the ANet CUI, providing the following information:

* The logon UUID
* The activity\_id.
* The payer customer\_id
* The registrant customer\_id
* The URL to which to redirect when the registration is complete.

**6. Login customer:** ANet will use the processCustomerLoginWithUUID WS method to login the parent.

**7. If login fails, redirect to AK with a failure result code.**

**8. Begin ANet CUI registration flow:** Now the user will be presented with the existing CUI registration flow, which handles \*all\* registration cases. However, there will have to be modifications to the existing flow, so it fits with AK. For the parent and family member flows above, we will be building new pages for this specific purpose. But the registration pages will need to remain our standard registration pages; we don’t want to duplicate development here. So the existing registration page(s) will have to be modified to eliminate CUI navigation, be restyled as AK pages, and redirect upon completion, as discussed above in “CUI workflow design considerations”.

**9. If registration workflow is cancelled, redirect to AK with a failure result code.**

**10. Redirect to the AK cart with success result code:** Because AK will already have built a cart and payment mechanism for the simple case, we assume that we should redirect back to AK with a success code, so it can present the cart and take payment.

## Technical documentation

## Q1 Redirect protocol

### POST parameters

The following parameters must be provided in the POST when AK redirects to ANet:

|  |  |
| --- | --- |
| Parameter name | Contents |
|  | Activity\_id to register in |
|  | Person\_id of parent |
|  | Person data from AK, to prefill into new |
|  | URL to redirect back to |

### Person data JSON

(Provide sample of AK person JSON, with comments as to which fields we use).

## Notes

## General assumptions

* Web service calls to ANet will only be done from the AK server code, not a mobile device or web page.
* Web service calls to ANet will be performed as a configured ANet “system user” dedicated to this purpose, like “Active.com”.
* No customer login is required for web services; this is only needed in the case of redirect to the CUI registration workflow.

## Assumptions about integration

* We will synchronize individuals (parents and children), but not families (groups).
* AK will not (initially at least)honor credit on account of the parent / family members and use it for payment

## Q1 development tasks

See EPIC <http://jirafnd.dev.activenetwork.com/browse/ANE-22248>

## Q1 design questions

* In terms of security, AK will have to have a ANet systemuser and password configured for the org, and will then provide a customer\_id plus AK person\_id to login the customer. Is this secure enough?
* What person data can be passed in for ANet account creation?
* Design getTransactionsByPerson WS API; what transactional data should be returned?

## Q2+ development tasks

Identified so far:

* Modifications to pages in general to allow no navigation and redirect back when an AK redirect has been provided.
* Modify Q1 login/account creation pages to redirect back immediately returning the customer\_id, if no activity\_id is provided.
* New CUI select family member workflow
* Modify registration pages to redirect back to AK on completion/cancellation, when a redirect URL is present.
* New PublicWS method getPersonCustomerID
* New PublicWS method getPersonLogonUUID
* New PublicWS method getRegistrationWorkflowFeatures
* Test harness pages for each workflow
* Development support for AK team

## Q2+ design questions

* For Q2+, what are the styling requirements of any pages presented by the CUI? We assume the normal CUI menu won’t be present.
* Design parameters for new CUI pages
* Document parameters for new WS methods
* Document keywords for getRegistrationWorkflowFeatures.