G

pl

Department of Community Safety:

Integration of GIS Boundary Capture Accreditation Module into Neighbourhood Watch System

Technical Specification

ESRI SA Project Reference No: **WC/026/03/18v0**

2018-05-22

# Document Lineage

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Responsible |
| 25/05/2018 | 01 | Initial Draft | T Richards |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 

# Reference Documents

|  |  |  |
| --- | --- | --- |
| Ref: | Document Name | Author/s |
|  | Neighbourhood Watch Management System Accreditation Module BRS |  |
|  | Neighbourhood Watch Management System Functional Specification |  |
|  |  |  |

# Glossary of Terms

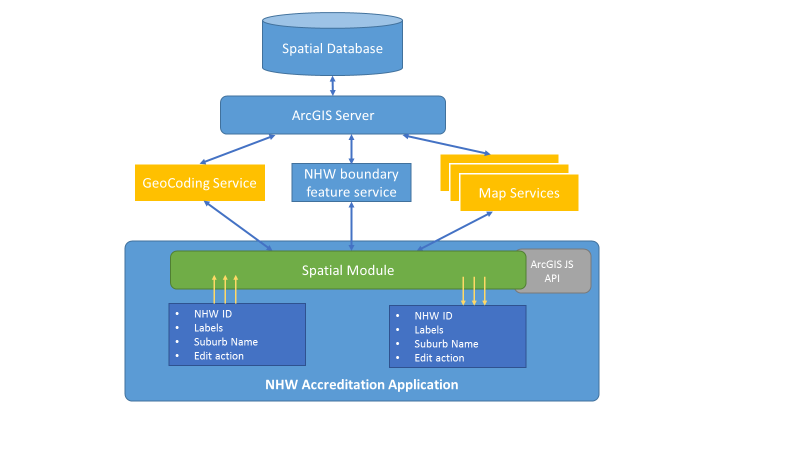
|  |  |
| --- | --- |
| AGS | ArcGIS Server |
| WCG | Western Cape Government |
| NWAS | Neighbourhood watch accreditation system |
| JS API | (ArcGIS) JavaScript API |
| IIS |  |
| NHW | Neighbourhood Watch |
|  |  |

# Executive Summary

This document provides the technical specifications for the spatial module of the Neighbourhood Watch Accreditation System being developed by the Western Cape Government.

The specifications are meant to meet the functional requirements outlined in the Functional Requirements document.

# Summary of Spatial Module components



1. A polygon feature class will need to be created in an SDE database.
2. A feature service will need to be published from the feature class.
3. The other map and geo-spatial services that need to be consumed by the spatial module must be configured and published (responsibility of CEI).
4. The spatial module will be an HTML 5 application written using the ArcGIS Javascript API.
   * It will be fully embedded within the NWA System (which in turn will be an MVC application)
   * The module will be launched from within the main NWA System
   * The main NWA System will pass a number of parameters to the spatial module including a unique identifier for the boundary area that needs to be created.
5. The spatial module will provide functionality to capture the boundary of a neighbourhood watch area and save it to the feature service
6. The spatial module will also feed data back to the main NHW System by populating hidden HTML controls in the main application.

Below are a table of components that will need to be developed/configured for the spatial module.

The following convention is followed:

CN – a component that needs to be configured

CD – a component that needs to be developed (i.e. custom development)

|  |  |
| --- | --- |
| **Component Code** | **Description** |
| CN001 | Polygon feature class for neighbourhood watch boundary |
| CN002 | MXD containing the polygon feature class that will need to be published as a feature service |
| CN003 | The neighbourhood watch polygon boundary feature service |
| CN004 | Other map and geo-spatial services that will need to be |
| CD001 | Spatial mapping module framework |
| CD002 | Neighbourhood watch capture tool |
| CD003 | Image export tool |
| CD004 | Precinct and street intersection tools |

Each of the components will now be discussed on more detail.

# CN001: Polygon feature class

A polygon feature class needs to be created in the corporate SDE database.

The coordinate system to be used is: (CEI to confirm)

Name of the feature class should be in line CEI naming conventions.

The schema should be:

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| OBJECTID | Esri Object ID |  |
| NHW\_ID | nvarchar(50) | The Neighbourhood watch ID supplied by the main application |
| Esri\_GUID | GUID | GUID generated by Esri application or database. |
| Date\_Created | DateTime | Timestamp of when the feature was first created |
| Date\_Edited | DateTime | Timestamp of when the feature was last edited |
| Status | Integer | cvdNHWPolyStatus  0 = unsaved  1 = unverified  2 = verified |
| Shape | Geometry |  |
|  |  |  |

# CN002: MXD

An MXD needs to be created with the Neighbourhood watch polygon feature service added.

# CN003: Feature service

A feature service should be published from the MXD created. The feature service should have feature creation and editing enabled.

Securing of feature services??

# CN004: Other web mapping and geo-spatial services

The web mapping, geo-coding, geometry and print services that the spatial module requires should be provided by CEI.

The mapping services should at least include:

* Suburbs – published with geometry and attribute data
* Streets – published with geometry and attribute data
* Police Precincts – published with geometry and attribute data
* Police Stations – published with geometry and attribute data

# CD001: Framework for mapping module

A mapping framework based on the ArcGIS JavasScript API needs to be developed that will accept the following URL parameters:

* A Neighbourhood Watch ID parameter
* A labelling parameter
* A suburb name
* An action parameter that denotes the type of editing that must occur (e.g. capture new boundary / verify existing boundary)

(**Note**: The action parameter will not be used in this phase of the project by will be needed in later phases )

The application must have a configuration file(s) where the following parameters can be set:

* The basemap for the application
* The operational map layers
* The URL of the feature service where the NHW polygon needs to be saved
* The URL of the printing service
* Suburb, Precinct, Street, Police station map service URLs
* Field names for the above services where the attribute fields that will be used to get the names of streets, suburbs, police precincts and police stations.
* Identify parameters for popup
* Geo-coding service
* Geometry service?
* Any other configurable parameters

The application must allow the user to zoom and pan the map using the mouse and keyboard (or using gestures in the case of a mobile browser).

The URL parameters supplied when the application starts should be saved as variables in the map application (or be added to the in-memory configuration of the application).

If a suburb name is provided as a URL parameter, the map should automatically be zoom and pan to the suburb provided.

# CD002: Neighbourhood watch boundary capture tool

The neighbourhood watch boundary capture tool should be a widget that is activated when an icon on the map is activated.

The widget should accept the following parameters when it is instantiated:

* NHW\_ID – The system ID of the NHW boundary to capture
* Labelling parameters – the labelling parameters obtained from the main application will be used to label the polygon captured.
* Edit Action - For this phase of the project, the edit action will always be to add a new feature, but in the future the edit action may also be to verify an existing feature.
* Urls for streets, etc plus field for street names
* The URL of the neighbourhood watch boundary feature service

As the widget opens, the Neighbourhood watch feature class should be queried to ensure that no active polygon with the NHW\_ID supplied currently exists. Id one does, the user must be provided with an error message and the editing tools disabled.

The activation of the widget should bring up small panel as below.



To define the neighbourhood watch boundary:

* The user will click on the Polygon icon. This will activate the drawing toolbar in the widget (<https://developers.arcgis.com/javascript/3/jsapi/draw-amd.html>).
* The user is then able to draw a polygon to define the outline of the neighbourhood watch area.
* When the polygon has been draw the ‘on-draw-end’ event will trigger the deactivation of the draw toolbar.
* The polygon defined will be added to a graphics layer.
* The polygon should be drawn in red to indicate that it has not yet been saved.
* The edit toolbar should then be activated and the polygon graphic selected (<https://developers.arcgis.com/javascript/3/jsapi/edit-amd.html>).
* Clicking on the map should disable the edit toolbar.
* Clicking on the graphic should enable the edit toolbar.

Clicking on the save button should:

* Insert a new polygon and attributes into the Neighbourhood Watch Boundary feature service
* The geometry of the feature should be the geometry of the graphic that has been drawn on the screen
* The attributes that are send through should be:
  + NHW\_ID
  + GUID
  + Current Timestamp for DateCreated
  + Null for last edited
  + Status = 1
* If the polygon is successfully inserted, change the symbology of the polygon on the graphics layer from red to blue.
* The map should also be re-centred on the polygon and processes in CD003 and CD004 initiated.

Clicking on the delete button should:

* Prompt the user for confirmation
* Change the status of the graphic in the feature service to 0 (i.e. retired/inactive)
* Remove the graphic from the screen

Populate a hidden text box on the main application to indicate that the feature has been successfully saved.

# CD003: Creation of image export

Once a feature has been saved and centred on the screen, generate the Webmap JSON that needs to be send to the print task.

The print should be returned as a PNG using the MAP\_ONLY option.

Make sure that the following layers are turned on in the print:

What to do with the image returned to get it to the main application – should we give them the url of the

# CD004: Street and precinct tool

Create a buffer around the polygon

Intersect the buffered polygon and the precints and streets

Format the results as JSON

Write to hidden fields on the main app

# External dependencies

The following are a list of external dependencies and services that need to be provided by CEI.

* ArcGIS server infrastructure
* SDE spatial database infrastructure
* The map services that will be consumed by the application. Apart from the feature service that is required to save the polygon boundaries of the neighbourhood watch areas, all map services needed for the application will need to be designed and published by CEI.
* Esri will provide the schema for the neighbourhood watch boundary feature service, but the feature service will need to be published by CEI when the system is deployed into the WCG infrastructure.
* A geo