Article
Ben Spead · Apr 25, 2018 6m read

Managing Many Environments and Protecting Production

NOTE: This content was originally presented at the InterSystems Global Summit in 2014, however related topics often come up on the Developer Community so I have decided to turn this into an article for easier reference and discussion. However, much of the content was pulled directly from the presentation slides so the article format resembles that of a PPT deck more than paragraphs.

Introduction

Teams that deal with many environments (e.g. for development, testing and production) multiplied by many systems (e.g. billing app, hr app, support app) can face a number of changes:

- Getting to a desired environment as quickly as possible
    - especially if that user has never accessed that particular environment before
- Keeping track of which environments exist for a given system
- Ensuring that the purpose of the environment is clear once connected
- Protecting production environments from accidental changes

The following discussion and sample code highlights best practices used internally within InterSystems for solving the above issues and more.

Identifying Environments

First a few Definitions…

- Environment
    - One ‘ copy ’ of an application
    - Depending on deployment architecture, this could be:
        - A Caché Namespace on a specific instance running specific functionality
        - An 3 tier architecture leveraging ECP and Mirroring
    - Each environment has a specific purpose:
        - SCRATCH – ‘ play ’ environment (non-controlled); periodically cloned from a controlled environment
        - BASE – development
        - TEST – test/validation
        - UAT – user acceptance
        - LIVE - production
- System
    - Complete collection comprising all environments for that application for a given site

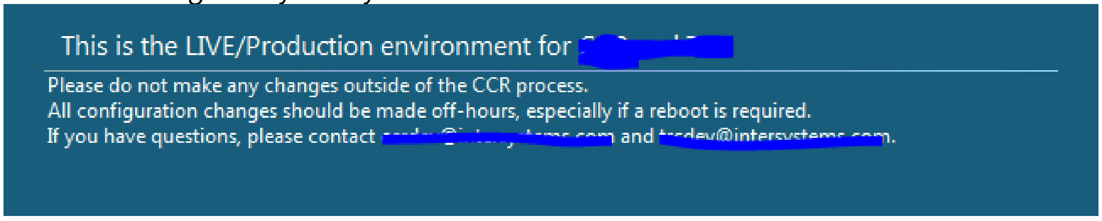
Naming Environments – an example from InterSystems

- Unique triad of identifiers:
    - SiteCode = organization where System is installed
        - E.g. ISCX for InterSystems in-house applications
    - SystemCode = aka “ Application Name ”
        - E.g. WRC for InterSystems Worldwide Response Center
    - EnvironmentCode = the specific environment

        - E.g. BASE for development work
    - E.g. The production WRC server is: ISCX/WRC/LIVE
- Additional identifiers may also be given based on a specific function of a server.
    - ‘ web ’ = web server
    - ‘ sql ’ = reporting server
    - ‘ app ’ = application server
    - ‘ db ’ = database server

Protecting Production

- When dealing with a plethora of Environments, people will occasionally connect to the wrong one from time to time
    - GOAL: Visually reinforce the Environment ’ s function
- Visual indicators
    - OS Login Message
        - Should be configured by OS System Administrator



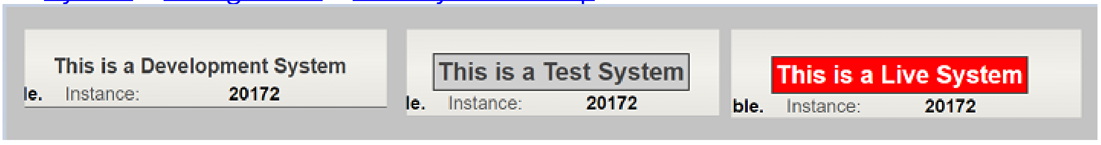
This is the LIVE/Production environment for [redacted]

Please do not make any changes outside of the CCR process.
All configuration changes should be made off-hours, especially if a reboot is required.
If you have questions, please contact [redacted] and [redacted].

    - Caché ZWELCOME.MAC banner for terminal users

```
▪ ZWELCOME displays text when someone logs into Caché via terminal; ro
  utine can be automatically created via environment configuration uti
  lity
  ?-bash-4.1$ csession myapps
  Environments are configured in the following namespace(s): [org/sys/
  env]
     %SYS: ISCX/MyApps/LIVE  [This is a LIVE environment; Use caution!
  ]
     FACLIVE: ISCX/FacMgr/LIVE  [This is a LIVE environment; Use cauti
  on!]
     PERSONDIR: ISCX/PersonDir/LIVE  [This is a LIVE environment; Use
  caution!]
     PLAN: ISCX/Plan/LIVE  [This is a LIVE environment; Use caution!]
```

    - Management Portal Header
        - The System Mode is displayed in the System Management Portal header and can be set via System > Configuration > Memory and Startup

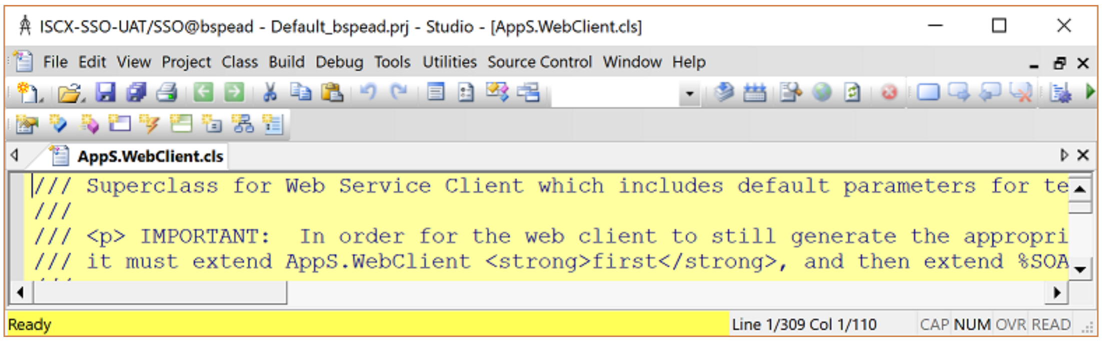


This is a Development System
le. Instance: 20172

This is a Test System
le. Instance: 20172

This is a Live System
ble. Instance: 20172

        - This can currently be set programmatically via the ^%SYS("SystemMode") global which can be given a value of "LIVE", "TEST", or "DEVELOPMENT" (IMPORTANT NOTE: There is no guarantee that this global will continue to work this way in the future)
    - Studio Background Colors
        - Studio background colors are saved on local client in the Windows Registry and can be used to highlight different colors for TEST/UAT vs LIVE


ISCX-SSO-UAT/SSO@bspead - Default_bspead.prj - Studio - [AppS.WebClient.cls]
File Edit View Project Class Build Debug Tools Utilities Source Control Window Help
AppS.WebClient.cls

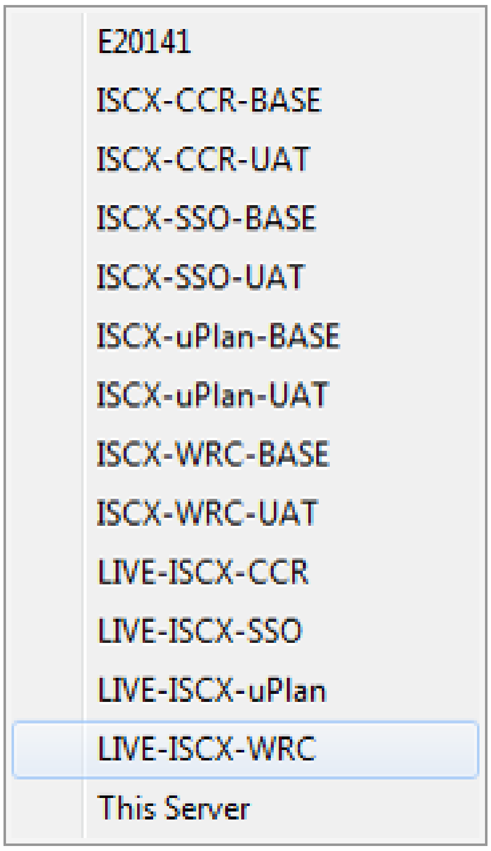
```
/// Superclass for Web Service Client which includes default parameters for te
///
/// <p> IMPORTANT:  In order for the web client to still generate the appropri
/// it must extend AppS.WebClient <strong>first</strong>, and then extend %SOA
```

Ready
Line 1/309 Col 1/110
CAP NUM OVR READ

            - These are controlled via the following Windows Registry Keys which control Studio behavior (this is not supported at this time for Atelier):
                - [HKEY_CURRENT_USER\Software\InterSystems\Cache Studio\Editor\Server Background Color]
                - [HKEY_CURRENT_USER\Software\InterSystems\Cache Studio\Editor\Status Bar Color]
    - Studio Environment Summary
        - Output Window in Studio or Atelier can show details written out by server-side source control hooks which indicate the identify and purpose of the environment:

```
------------------------------------------------------------------
---------
Source Control class %Studio.SourceControl.ISC is enabled for 'MyApp
'
This namespace is Permanently Disconnected from Perforce
Current CCR Configuration
   Organization: ISCX
   System:       MyApp
   Environment:  UAT
^Sources = C:\Perforce_D\custom_ccrs\us\ISCX\MyApp\UAT\
```

- Lock down non-BASE via Source hooks
- Separate LIVE from other environments in Server Names witin the Caché Server Manager
    - [SiteCode]–[SystemCode]–[BASE¦TEST¦UAT]
      vs.
    - LIVE–[SiteCode]–[SystemCode]
    - Naming these automatically in Server Definitions can make it obvious in the Caché Server Manager when connecting to remote instances:



Accessing Environments

The importance of CNAMEs

- CNAMEs are critical for our environment management because:
  - InterSystems has a myriad of servers running a plethora of environments
  - Environments frequently need to be moved from server to server, due to hardware migrations, system re-configuration, etc
  - As the environment landscape evolves, it ’ s important to keep connectivity details constant for the benefit of users and integrations
- NOTE: we standardize all Caché configurations to use ports 1972 and 57772 to ensure consistency
- CNAMEs are constructed based on various pieces of the environment name and function:
  - ([function].)([EnvCode].)[SystemCode].iscinternal.com
- sql.test.wrc.iscinternal.com = host for the TEST WRC reporting node
- base.trc.iscinternal.com = development environment for the TRC
- download.iscinternal.com = internal IP of Download production host

Simplify Access through Automation

- Given compliance with naming and configuration standards, access becomes much simpler
- We have created a “ System Dashboard ” which allows the tracking of different Systems, their Environments and their specific connectivity details
  - Quick links to Home page, Management Portal, Telnet, etc
- This database of System configuration data can be exported on “ Caché Server Manager ” Windows Registry keys, eliminating the need for local hand-configuring of access

Client Configuration and Accessing Environments Demo Code

See this gist of a Studio Project or this XML File for sample code that shows all of this together:

- Client Configuration

-
  - Automatic ZWELCOME message and Management Portal Banner for BASE, TEST, UAT, LIVE environments
  - Configure^Config to change configuration
  - Source Control Hooks and ‘ environment data summary ’ when connecting to a namespace
  - Hooks warnings when connecting to non-development
  - Optional enforced ReadOnly mode
- System Dashboard (including auto-population)
  - Working Telnet link, SMP Link, App Link, etc
  - Downloadable environment details via Windows Registry Key to populate Caché Server Manager
    - Automatic server definitions and studio coloring keys

Sample Code Instructions:

1. Import HandlingEnvironments.xml into the SAMPLES namespace (tested on Caché 2014.1 RC)
2. Configure Environment Details for SAMPLES namespace by running:
   1. Do Configure^Config
      1. This will impact ZWELCOME, and Management Portal banner
3. Open the "HandlingEnvironments" project via Studio, right-click on csp/samples/SystemView.csp and select 'show webpage'
   1. from here, you can play with Caché Server Manager export, telnet connection (PuTTY installation is required)
   2. export of environment definitions will include Studio background coloring, and Caché Server Manager listings

#Caché #Code Snippet #Development Environment #Studio #Tips & Tricks

Source URL:https://community.intersystems.com/post/managing-many-environments-and-protecting-production