## ToolTips

### Releases:

Add mouse click action to draw\_items to take user to Release View page.

#### TFS:

* Name (hyperlink to TFS)
* Description
* Event List (hyperlink to ServiceNow)
* Team (Can we create links to Seiso User pages?)
  + Release Manager
  + Implementer
  + Project Manager
  + Test Lead
  + Dev Contact/Team
* HipChat Room (Can this be a link to HipChat, opening to the room?)

|  |  |  |
| --- | --- | --- |
| Name | AIRINT 2015\_3R1 Release with MStravObj Change | |
| Description | AIRINT 2015\_3R1 Release with MStravObj Change [Major Release] | |
| Events |  | |
|  | Production Release | March 10, 2015 1:00 PM PDT |
|  | SOAK | March 11, 2015 12:00 AM PDT |
| HipChat Room | T2 AIR | |
| Implementer | Matt Hageman | |
| Test Lead | Peter Kurpis | |
| Dev Contact | Ramasubbu Subbareddy | |
| Project Manager | Steve Well | |
| Release Manager | Michelline Bouvier | |

#### Mingle:

* Name (hyperlink to Mingle)
* Event List (hyperlink to ServiceNow)
* Code Change Count

|  |  |  |
| --- | --- | --- |
| Name | 2015-03-r3 | |
| Code Changes | 15 | |
| Events |  | |
|  | SOAK | March 11, 2015 11:00 PM PDT |
|  | Production Release | March 17, 2015 5:00 PM PDT |
|  | Hotfix | NA |

### Events:

#### TFS and Mingle: (They are the same as the source for both is ServiceNow records.)

* Name
* Details -- Truncate with default length (…more?)
* ServiceNow Item (hyperlink to ServiceNow)
  + Might not be trivial, need to see if a URI can be generated or if there’s another field to grab from the REST API to assist here.
  + <https://expedia.service-now.com/nav_to.do?uri=change_request.do?sys_id=7c474ac385c5b900fc3c4e03c2db4a6a>
* Approval
* State
* Status
* Coordinator
* Environment

|  |  |
| --- | --- |
| Name | Production Release |
| Schedule | Start\_time – End\_time |
| Details | In order for your change to be approved by the… |
| Summary | ??? (short\_summary) |
| CRQ | CHG0006542 |
| Approval | No Longer Required |
| State | Canceled |
| Status | success |
| Coordinator | Sandy Senapathi |
| Coordinator Group | u\_coordinator\_group |
| OU | ??? |
| Type | ??? |

## Installation Notes

* Potential requirement on python and c++ if using d3.
  + Download of minified file from <http://d3js.org/d3.v3.min.js> should eliminate this requirement.
  + Needs to be tested as the machine I used this on already has the requirements installed.
    - Amazon EC2 images appear to had Python installed by default, so probably isn’t an issue.
    - Make a note in any troubleshooting guides
* Amazon EC2 Install Notes for first dev environment
  + Install git.
    - It was already in the image.
  + Install nodejs and npm.
    - Followed <https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager>
    - sudo –i (Have to be logged in as root for the following curl command.)
    - curl –sL <https://rpm.nodesource.com/setup> | bash –
    - yum install –y nodejs
  + Resolve SELINUX=enforcing
    - sudo yum install -y policycoreutils-python
    - sudo semanage --- Already active???
    - <http://docs.mongodb.org/manual/tutorial/install-mongodb-on-red-hat-centos-or-fedora-linux/>
  + Install MongoDB
    - DO NOT CREATE SERVICE ACCOUNT ‘mongod’ NOR GROUP ‘mongod’, the yum install will create them automatically.
    - Followed <http://docs.mongodb.org/ecosystem/platforms/amazon-ec2/#deploy-mongodb-ec2>
    - sudo yum –y update
    - sudo vi /etc/yum.repos.d/mongodb.repo
      * [MongoDB]
      * name=MongoDB Repository
      * baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86\_64
      * gpgcheck=0
      * enabled=1
    - sudo yum install -y mongodb-org-server mongodb-org-shell mongodb-org-tools
    - sudo mkdir /mongo
    - sudo mkdir /mongo/data
    - sudo mkdir /mongo/journal
    - sudo ln –s /mongo/journal /mongo/data/journal
    - sudo mkdir /mongo/log
    - sudo chown -R mongod:mongod /mongo
    - sudo vi /etc/mongod.conf
      * dbpath = /mongo/data
      * logpath = /mongo/log/mongod.log
    - sudo vi /etc/security/limits.conf
      * mongod soft nofile 64000
      * mongod hard nofile 64000
      * mongod soft nproc 32000
      * mongod hard nproc 32000
    - sudo vi /etc/security/limits.d/90-nproc.conf
      * \* soft nproc 32000
      * \* hard nproc 32000
    - sudo service mongod start
    - ‘mongo’ to validate connection.
    - sudo chkconfig mongod on
      * Sets up to run on startup.
  + Forgoing “forever”, opting to use system.d’s service handling with its build-in restart abilities. -- ~~Install ‘forever’ npm~~
    - ~~npm install forever –g~~
  + Grab Dashboard from GitHub
    - Go to <https://github.com/CopperTopp/timeline-dashboard>.
    - Click on the HTTPS clone URL button on the right-hand side of the page to copy the URL into the clipboard.
    - Create a directory to put the files in, used ‘/usr/reldash’.
      * Note that git will create a directory ‘release-dashboard’ during the file copy process.
    - CD into the new directory.
    - Run ‘git clone <Git Clone URL>’ by pasting the URL we copied in the step above.
  + ~~Start the Cache Manager with ‘forever’~~
    - ~~forever start cache\_manager.js~~
  + ~~Start the Web Server~~
    - ~~forever start server.js~~
  + ~~Validate running by running ‘forever list’ and make sure both scripts are running.~~
    - ~~Forever is designed to restart the scripts if they stop, so make sure to put some sort of monitoring in place for SERIOUS breaks.~~
    - ~~Adding ‘-w’ to the command line on startup will watch the directory for file changes, so a git checkout will cause the server to restart to pick up the changed files. Sort of a zero down time…~~
* Chef Notes
  + Chef Image Account
    - Username: vagrant
    - Password: vagrant
  + Reloading image with changed recipe in Sandbox Environment
    - chef-repo/cleanup-chef-node.cmd <recipe>-<hosttype> [reldash-amzlinux]
    - knife cookbook delete <recipe> -y [reldash]
    - knife cookbook upload <recipe> [reldash]
    - vagrant up <recipe>-<hosttype> [reldash-amzlinux]
  + Update or customize .box image
    - Install the base image.
      * IGNORE – Don’t Use A “vagrant up” core. -- Create a “Clean” role.
        + Go into Sandbox chef-server, <http://localhost:4040>.

See chef-server.md for default passwords.

* + - * + Go to “Roles” tab.
        + Click “Create” in “Roles” header.
        + Enter role name “clean”.

Optionally give a description.\

Do NOT add a Role to the run-list, unless you want it permanently in your base image.

* + - * + Click “Create Role”.
      * IGNORE – Don’t Use A “vagrant up” core. -- Install the base image with the “Clean” role using vagrant.
        + “vagrant up clean-amzlinux”
      * Grab the .box file for the image you want to update.
        + <http://kitchensink.aws.sb.karmalab.net/vagrant/boxes/>
        + Right-click the .box file you want to update and copy it locally.
      * Extract the VirtualBox Import files
        + Rename the file from .box to .tar.gz
        + Open the archive and extract the .ovf and .vmdk files to another directory.

Note that the .box file actually contains another .tar file, so it is now a .tar within a .tar file, so you’ll need to extract the .tar file to get to the files.

Most GUI archivers will do this almost automatically.

* + - * Import the image.
        + Open Oracle VM VirtualBox Manager.
        + In the File menu, click Import Virtual Appliance.
        + Browse to the .ovf file extracted above from the .box (now .tar) file and click Next, then Import.
      * Go into the image and make the changes you need.
      * Create the new image.
        + Note the name of the instance in VirtualBox Manager.

It can be changed by stopping the server and updating Settings.

It’s also a convenient way of cut/pasting long names.

* + - * + If not already stopped, stop the instance through VirtualBox Manager.

Right-click, Close->ACPI Shutdown is safest way.

* + - * + Use vagrant to create the .box image. You’ll need the VirtualBox name we noted above.

vagrant package --base <name of VirtualBox instance> (optionally add “--output <.box file name>”)

In vagrant 1.7.0 and 1.7.1 there is a potential bug at this point.

If you get an argument error, something about “2 into 1”, see <https://github.com/mitchellh/vagrant/issues/4962>

Basically, you have to change one line in machines.rb or install a newer vagrant.

* + - * + Save or move the resulting package.box file.

Windows Tip, Stay on the same hard drive as your chef-repo as another bug with vagrant will fail to find the file if on another drive.

* + - * Add the new image to your Vagrantfile OSES array.
        + {  
           “name” => “mynewimage”,  
           “box” => “file://C/folder/my\_new\_image.box”  
          }
        + Warning!!

If “name” is already registered in the chef-server, it will use the already stored image.

To Check If “name” Is Already There

vagrant box list

Look for the “name” in the list.

To Remove “name”

vagrant box remove “name”

All instances using this image must be deleted before removing the box.

You don’t have to as ‘vagrant up’ combined with path in Vagrantfile will do this, but to ensure the .box file you want is registered, you can run

vagrant box add --name “name” <file://C/path/image.box>

Run vagrant box list to verify.

* + - * Install your new image.
        + “vagrant up role-mynewimage”
        + If you run into an issue where “vagrant up” complains that the VirtualBox Guest Additions are out-of-date

Install plugin vagrant-vbguest.

vagrant plugin install vagrant-vbguest

Rerun “vagrant up” command.

This module will update the VirtualBox Guest Additions during the “vagrant up” execution.

* + Reloading updated .box image
    - vagrant box remove <name>
    - vagrant box add –name <name> <file://d/path/file.box>
  + amzlinux SandBox fixes
    - selinux
      * Symptom – Though sudo’ing as the service account works, the system/service cannot start the service, usually with some sort of permissions issue.
      * Fix – Disable selinux
        + sudo yum install –y policycoreutils
        + sudo yum install –y policycoreutils-python
        + sudo setenforce 0 or Permissive

Temporarily sets for this boot instance.

* + - * + sudo sed -i ‘s/SELINUX=.\*/SELINUX=disabled or permissive/g’ /etc/selinux/config

Permanently set after reboot.

* + - firewalld
      * Symptom – Even after adding tcp/8080 to the ‘internal’ zone, could not access nodejs Express listener on port 8080.
      * Fix – Disable firewalld
        + sudo systemctl disable firewalld
        + sudo systemctl stop firewalld

## Meeting Ann Batt

Suggested use of D3, took a look at a zoom example she pointed out, VERY cool, was able to zoom without stretching text: <http://bl.ocks.org/mbostock/3892919> Demo and Code.

D3 Docs on Zoom behavior: <https://github.com/mbostock/d3/wiki/Zoom-Behavior>

D3 Date Axis, transitioning from year view down to hour view: <http://bl.ocks.org/mbostock/2983699>

## Angular Info

<https://docs.angularjs.org>

<https://docs.angularjs.org/tutorial>

<https://docs.angularjs.org/guide>

<https://docs.angularjs.org/api>

<https://egghead.io>

<http://www.thinkster.io>

(For Mac) <http://kapeli.com/dash> (For Windows) <http://velocity.silverlakesoftware.com/>

<https://www.codeschool.com/code_tv/soup-to-bits-warming-up-with-ember>

## D3 Info

* Mike Bostock's “Path Transitions” (<http://bost.ocks.org/mike/path/>)
* Mike Bostock's “Bar Chart tutorial” (<http://mbostock.github.com/d3/tutorial/bar-2.html>)
* Jerome Cukier's “Creating Animations and Transitions with D3” (<http://blog.visual.ly/creating-animations-and-transitions-with-d3-js/>)
* D3 wiki (<https://github.com/mbostock/d3/wiki/Transitions>)
* Look at using Nests to pivot data for different views later <https://github.com/mbostock/d3/wiki/Arrays> scroll down to Nest heading.
  + Can take a flat array and turn into hierarchical tree object based on values in keys in the array’s objects. Think SQL-ish sort by Product.

## Collision Detection

Collision Detection is needed to place overlapping release entries in different rows. Has to be calculated when zooming as labels begin colliding as they don’t maintain scale with the chart.

At some zoomed out level, should labels be hidden so that the graphic can represent reality and not be turned into a “skinny tree” because of colliding labels? At a range of 1 year? 6 months?   
Can this be detected with data and how congested the view is?

### Philosophy

* Each Product/Project has a Priority.
  + Initial Priority based on who is closest to releasing today and soonest in time, then descending by who is next.
    - After initial Priority, Priority can be set as part of the Product filtering mechanism and stored for return visits to the page.
  + Rather than having an integer rating system, from 1 to n, using RACI Interest levels would be more manageable for the users.
    - Responsible – High personal interest as they’re responsible for this product.
    - Accountable – High personal interest as they’re accountable for the actions of the responsible.
    - Consulted – Interested as a collaborator, contributor, or consumer of the Product. Their Product is affected by slips in this Product’s release status.
    - Informed – Interested as a watcher, a curious outsider.
  + View priority is then a calculation of Release Start Time and Interest Level.
    - Releases placed by Start Time first, then if there’s a collision in the view, by this Priority level.
    - Overlapping Releases bumped first by Time, then by Priority (should that be reversed????)
* View has rows, like swimlanes.
  + Everything is loaded into Row 1 at initialization.
  + Collision Detection takes a pass, bumping colliding Release Items to a higher (or lower priority) row.
    - Lower Priority (above) Product’s Releases are bumped, higher Priority get to stay in their row.
  + Then Row 2 is processed, then Row 3 until there are no more collisions.
    - This can easily be checked by completing Row (n) and seeing that Row(n+1) is empty, meaning no items in Row (n) collided and were bumped to the next higher row.
  + On Zoom, recheck all rows, don’t start over by placing everything in Row 1.
    - On Un-Zoom, dealing with checking to see if an item can descend into a lower row.
    - Starting with Row 2, compare collision of Row(n) with Row(n – 1) and move lower if no collision detected.
      * If moved to a lower Row, continue checking Row(n-2…) until no collision is detected.
    - Finally move into Row(n-x).
  + Should Expanded Tool Tips be added to Collision Detection? Would have to calculate how many rows a Tool Tip blocks, so that collided Release Items know which Row to move to.
* Alternate Method, Release Items load at bottom of the page in order of Priority, then bubble to the top without covering other Release Items, settling as close to y = 0 as possible.
  + Check out D3 Force Layout, with the entire Y = 0 thread as the attractor.

## Client-Side Classes

* ReleaseFactory
  + Constructor
    - Parameters
      * D3 root element object
      * xScale – From the zoom behavior for calculating draw updates after zoom.
      * Viewport – js object of type {x, y, width, height} defining the clip region for the UI framework.
      * Array of Mongoose ProductModels
    - Operation
      * Iterate ProductModel Array
        + Iterate ReleaseModel Array

Generate ReleaseItems

* + Properties
    - xScale – Capture scale from D3 zoom behavior mechanism
    - products – Array of Mongoose ProductModels
    - releases – Array of ReleaseItems
  + Methods
    - draw()
      * Operation
        + Render and manage clippable, zoomable timeline UI framework.
        + Call ReleaseItems through each item’s Draw method, returning an SVG element with children that make up a Release UI component.
        + Detect collisions of returned ReleaseItems SVG root, and place in zoomable, clippable UI framework
        + Should be designed to handle both creation and update mechanisms. Might as well do destruction while I’m at it. (data(), enter(), exit()).
    - zoom()
      * Operation
        + Handle changes to framework UI and ReleaseItems when zoom has been called. Place <instance>.zoom() in the zoom event handler.
* ReleaseItem
  + Constructor
    - Parameters
      * D3 ReleaseFactory Root – Should be the root of the content for the clippable, zoomable UI framework.
      * Mongoose ReleaseModel instance.
  + Properties
    - Inherit all properties from the Mongoose ReleaseModel.
    - events – Array of EventItems
    - changes – Array of ChangeItems
  + Methods
    - draw()
      * Operation
        + Render svg element through D3 containing all elements to show a Release in the UI framework home in ReleaseFactory.
        + Iterate through events and changes to place their markup on the Release Item element.
        + Create tooltip hidden and add functionality to show and hide based on mouse clicks.
        + Decided to use mouse clicks and toggled selection to show tooltips rather than hover states.

This will allow comparison of tooltip values between selected Release Item rather than moving back and forth trying to get the hoverstates to fire among the RIs your trying to compare.

* + - * + Also, thinking on keeping the change and event decorations hidden until an item is toggled to selected.

This will allow colors to be used initially to show which product a release belongs to, then on selection, the sub-events within the release.

* + - * + ToDo: How do we call attention to a Release Item that is “at risk”?

Color (intensity)

Animation

Motion

Flashing

Glowing

Warning Banner calling attention to critical events and Release Items “at risk”

* + - zoom()
      * Parameters
        + xScale – The new x axis scale value after the zoom, from the D3 zoom behavior mechanisms.
      * Operation
        + Update x and width values with the new xScale.
* ReleaseToolTip
  + Constructor
    - Parameters
      * ReleaseItem parent
      * Mongoose ReleaseModel
  + Properties
    - hideState – Whether tooltip is hidden or not.
    - releaseItem – D3/SVG element for the ReleaseItem, the parent for this control.
    - release – holder for the associated ReleaseModel object from the constructor.
  + Methods
    - toggleHide() – Switch between hidden and shown, flipping the value of hideState and correctly showing/hiding the tooltip element.
      * Will require the collisionDetection to fire, so maybe we just fire <nowiki>this.dr</nowiki>aw()
    - draw()
      * Render the view of the tooltip, hidden or shown based on <nowiki>this.hi</nowiki>deState.
      * Adjust location based on other UI elements, including other shown tooltips.
      * What’s the best way to deal with other items colliding with ToolTip?
        + Overwrite and overshadow? What if we want to compare two releases close to each other?
        + Detect collision and move ReleaseItems that tooltip would overwrite? Can UI Framework handle “tall” renderings, as each RI will get pushed down as more tooltips open?

## Planning

* NodeJS Middleware
  + Polling Service – Read Data Masters and populate Buffer Database (MongoDB)
    - Investigation
      * Preserve Product Configuration design
        + Product-Adapter.js: Business Intelligence, mapping master data to MVVM modeling/DB storage
        + Product.json file for layout and configuration information specific to each Product.

Is .json necessary with the need for and addition of Product-Adapter.js?

Should Product.json roll-up into Product-Adapter.js?

* + - Implementation
      * ExpWeb Product
        + Mingle Adapter – Done
        + ServiceNow Adapter – Done
        + Business Intelligence – Putting Mingle and ServiceNow data together in a meaningful way. – Done
      * AirINT Product
        + TFS Adapter – Adapter written and working, need to analyze what is needed and how to grab it. – 50% Complete
        + ServiceNow Adapter – Is this even needed, is anything tracked in ServiceNow as reference by TFS?
        + Business Intelligence – Putting the pieces together in a meaningful way.
      * Cars Product
        + TFS Adapter – Adapter written and working, need to analyze what is needed and how to grab it. – 50% Complete
        + ServiceNow Adapter – Is this even needed, is anything tracked in ServiceNow as reference by TFS?
        + Business Intelligence – Putting the pieces together in a meaningful way.
      * BFS Product
        + TFS Adapter – Adapter written and working, need to analyze what is needed and how to grab it. – 50% Complete
        + ServiceNow Adapter – Is this even needed, is anything tracked in ServiceNow as reference by TFS?
        + Business Intelligence – Putting the pieces together in a meaningful way.
  + Data Service – Provide Dashboard page with MongoDB access (Is this part of AngularJS MVVM??)
    - Investigation
      * Does AngularJS’s MVVM design even require an intermediary NodeJS service to access MongoDB? – Not Started
* AngularJS MVVM
  + Data Modeling
    - Reading data either from NodeJS Data Service, or directly from MongoDB
    - Translate Data Model items into jQuery controls and instances.
* JQuery Webpage
  + Decide on Timeline Control
    - Almende Timeline Control
      * Timeline with Grouping, attempting to show grouped events on the same line.
        + While events aren’t in the same Release “bar”, they are grouped in “swimlanes”.
      * References
        + <http://almende.github.io/chap-links-library/js/timeline/doc/>
        + Which has become <http://visjs.org/>, which has an npm and bower install methods.
        + Examples: <http://almende.github.io/chap-links-library/timeline.html>
      * How to wire-up in jQuery/AngularJS to MVVM?
    - TimelineJS
      * Has nice multi-tiered, side-scrolling controls
      * Events are isolated and no way to create sub-events, so can’t associate steps/tasks within a Release.
        + Resolve by showing just the release, then a click takes us into a new view with each release’s tasks and steps?

How do we deal with losing association to external-to-release events that might still impact the release steps?

* + - * How to wire-up in jQuery/AngularJS to MVVM?
    - Write-My-Own
      * How to draw in javascript?
      * How to create custom controls to support timeline view?
      * How to wire up to jQuery/AngularJS to MVVM?

## Design Notes

* Backend
* Summary Page Data:
  + ReleaseItems:
    - Properties:
      * Product
        + The Product being released.
      * Project (Is there a use for a sub-Name?)
        + Not sure if this is a construct that makes sense, but a Product may have multiple Projects, each with a different Release Cycle.
      * Version Number
        + Label of this instance of a Release. Unique identifier when coupled with Product and maybe Project as well if used.
      * Status
        + Planned, InProgress, Succeeded, or Failed state.
      * Description
        + Short phrase describing the type of Release. (Release, HotFix, ???)
      * StartTime (Do we need if we have FCTime?)
      * FCTime (Feature Complete ??)
      * TSOTime (Test Sign-Off ??)
      * SOAKTime (SOAK ??)
      * StopTime (or RTWTime)
      * Responsible/Contact
        + Who is responsible for the instance. Who to contact for questions and issues.
  + HazardItems:
    - Properties:
      * Hazard
        + Short name or phrase for the Hazard
      * Status
        + Planned, InProgress, Resolved state.
      * Description
        + More detailed summary of the Hazard and it’s impact.
      * StartTime
      * StopTime
      * Responsible/Contact
        + Who do contact regarding the Hazard.
      * Product/RiskLevel List
        + List of ProductItems affected and RiskLevel posed to each as it may not be uniform across all Products.
  + RiskLevel:
    - Values:
      * Down/Outage (Red)
        + The item will be unusable during a period. Resource maintenance, known outages. Guaranteed that the environment will be impacted negatively.
      * Instability/Unknown Risk/Potential Outage (Yellow)
        + The item has a higher than normal chance of being down. Could be from a known threat to its resources, or to interaction of external-to-the-environment resources that will be down.
      * No Risk (Green)
        + As far as we know, no risks exist.
* Summary Page Data with Environments:
  + EnvironmentItem:
    - Properties:
      * Name
      * Releases
        + List of ReleaseItems where this environment is primary or secondary.
      * Products
        + List of Products that have a presence or a ReleaseItem in this environment. May just be calculated from the Releases Product fields.
      * Hazards
        + List of HazardItem instances for this EnvironmentItem.
  + ReleaseItems:
    - Properties:
      * Product
        + The Product being released.
      * Project (Is there a use for a sub-Name?)
        + Not sure if this is a construct that makes sense, but a Product may have multiple Projects, each with a different Release Cycle.
      * Version Number
        + Label of this instance of a Release. Unique identifier when coupled with Product and maybe Project as well if used.
      * Status
        + Planned, InProgress, Succeeded, or Failed state.
      * Description
        + Short phrase describing the type of Release. (Release, HotFix, ???)
      * StartTime (Do we need if we have FCTime?)
      * FCTime (Feature Complete ??)
      * TSOTime (Test Sign-Off ??)
      * SOAKTime (SOAK ??)
      * StopTime (or RTWTime)
      * Responsible/Contact
        + Who is responsible for the instance. Who to contact for questions and issues.
      * Environment/RiskLevel List
        + List of Environments affected by this Release and the RiskLevel posed when the Release is Active.
  + HazardItems:
    - Properties:
      * Hazard
        + Short name or phrase for the Hazard
      * Description
        + More detailed summary of the Hazard and it’s impact.
      * StartTime
      * StopTime
      * Responsible/Contact
        + Who do contact regarding the Hazard.
      * Environment/RiskLevel List
        + List of EnvironmentItems affected and RiskLevel posed to each as it may not be uniform across all Environments.
  + RiskLevel:
    - Values:
      * Down/Outage (Red)
        + The item will be unusable during a period. Resource maintenance, known outages. Guaranteed that the environment will be impacted negatively.
      * Instability/Unknown Risk/Potential Outage (Yellow)
        + The item has a higher than normal chance of being down. Could be from a known threat to its resources, or to interaction of external-to-the-environment resources that will be down.
      * No Risk (Green)
        + As far as we know, no risks exist.
    - Values are stored in each instance in a per Environment structure, so the RiskLevel per environment can be calculated.

Environments encapsulate sandboxes, groupings of inter-related resources. May still have external, shared resources, but mostly isolated. Mostly a logical construct used to express risks posed by Hazards. For instance, a SQL server going down for scheduled maintenance would affect he Environment of the applications and services directly accessing it, but it could pose a potential risk to environments that have resources accessing those applications and services. The primary environment would be at RiskLevel – Outage (Red), and any dependent environments, ones that we don’t know will be affected or not, but very well may be, would go into RiskLevel – Instability (Yellow). The Hazard instance would reference back to the known Outage.

A Release in one environment may be a risk to another, as it takes down dependent resources.

Should Environments be split into GeographicEnvironmentItems and LogicalEnvironmentItems? Geographic would be sites, like Chandler. Logical would be server pools and/or subnets.

Would it be worthwhile to add Teams/Groups as an item?

* Data Service Interface
  + GetItems(DateTime start, DateTime end);

Stub Time Calculations:

start – fc – tso – soak – end

Product has:

ReleaseTolerance

FCDuration

TSODuration

SOAKDuration

EndDuration

Overlap

QuietPeriod

Release has:

R(int) = Function Returning Random from -int to int

StartTime = Global Variable CurrentTime

FCTime = CurrentTime + FCDuration + R(ReleaseTolerance) => CurrentTime

TSOTime = CurrentTime + TSODuration + R(ReleaseTolerance) => CurrentTime

SOAKTime = CurrentTime + SOAKDuration + R(ReleaseTolerance) => CurrentTime

EndTime = CurrentTime + EndDuration + R(ReleaseTolerance) => CurrentTime

CurrentTime += QuietPeriod

If Overlap > 0

CurrentTime -= RANDOM(0-1) \* Overlap

Samples That Already Exist in Expedia

<http://teambasecamp/sites/675566/SitePages/Home.aspx>

<http://pmdb/cab_review.aspx>

Mention of AngularJS ServiceNow Plug-In, active by default

<http://wiki.servicenow.com/index.php?title=List_of_Plugins>

<https://community.servicenow.com/thread/166740>

<http://www.akeelnazir.com/#/blog>

Service Account:

s-ChangeDashboard [remedy]

ServiceNow Performance Analytics – ServiceNow App with dashboarding capabilities. Used to Mirror 42.

Mingle Project – Trunk Release

<https://ewemingle.karmalab.net/projects/trunk_release/cards/14591>

ServiceNow Structure:

Change Requests in SN Table [change\_request] <https://expedia.service-now.com/change_request.do?WSDL>

Configuration item: Expweb in SN Table [cmdb\_ci\_service] <https://expedia.service-now.com/cmdb_ci_service.do?WSDL> “name=’Expewb’”

Now, what maps them together?

Post about walking the CMDB, for the purpose of figuring out what CIs are affected by a Task.

<http://www.servicenowguru.com/scripting/script-includes-scripting/walking-servicenowcom-cmdb-relationship-tree/>

Points to Table [cmdb\_rel\_ci] <https://expedia.service-now.com/cmdb_rel_ci.do?WSDL>

Dump to CSV seems to point to that this is how ci relates to ci, not ci to change request…

May need to ask for access to create Database Views within ServiceNow, join the tables in a view, then access the view through the API.

<http://wiki.servicenow.com/index.php?title=Database_Views>

Appears that a Database View responds just like a Table. Testing the built-in Database View [change\_request\_metric] with <https://expedia.service-now.com/change_request_metric.do?WSDL> returned a WSDL. (Might be just because it IS built-in…)

PMDB ServiceNow Interface:

[http://pmdb/xml\_services/xml\_cab\_servicenow.aspx?q=u\_environment=Production^ORu\_environment=3rd%20Party^state%3E=-3^state%3C=7^start\_dateBETWEEN2014-09-21@2014-09-26](http://pmdb/xml_services/xml_cab_servicenow.aspx?q=u_environment=Production%5eORu_environment=3rd%20Party%5estate%3E=-3%5estate%3C=7%5estart_dateBETWEEN2014-09-21@2014-09-26)

This already exists, though maintenance is lower. And if I need something that isn’t there, it might be just as difficult to add new fields…

For Later:

Slimple, a simple nodejs framework for service creation. <https://www.npmjs.org/package/slimple>

Things We COULD Watch:

Sign-offs on Releases.

SOAK Analysis Results (Realtime?)

Build stability analysis of pre-Releases.

Test results of post-build testing.

Changes in Perforce associated with Change Requests associated with Releases.

Could we use this to show how complete an upcoming Release is?

Rate of check-ins, or maybe just the total count for the timeperiod.

Incidents in already released Instances. Points to Quality and Stability of past Release.

Environment Stability

Incidents or issue tracking against environment resources.

Environment up-time.

Planned outages.

Chef???

How often an environment was refreshed.

Are there any metrics associated with the deployment and delivery systems?

People

Track vacations as a Warning Hazard?

Track empty head-counts as Resource Availablity?

<http://timeglider.com/widget/>

Another 3rd party JS timeline.

<http://almende.github.io/chap-links-library/js/timeline/doc/>

Timeline with Grouping, attempting to show grouped events on the same line.

Which has become <http://visjs.org/>, which has an npm and bower install methods.

Examples: <http://almende.github.io/chap-links-library/timeline.html>

Mingle looks like it might be a better place to get release information for ExpWeb, see <https://ewemingle.karmalab.net/projects/trunk_release/cards/list?columns=Release+Type%2CRelease+Branch+Creation%2CRelease+Test+Signoff%2CProduction+Soak%2CRTW&filters%5B%5D=%5BType%5D%5Bis%5D%5BRelease%5D&style=list&tab=Release+List>

Barring API exploration, might have to scrape content to get information, but it is a regulated form, so that might not be so bad.

ServiceNow Access Request forms:

In Service Now, access “IT Service Management” in the Service Catalog, then Service Now.

In User and Group Maintenance, ask to be added to the “EWE-Operations” group.

Request for access to Dev/Test by Chris REQ0102912

Request for WSDL Service Account REQ0102909

Request for specific access to Database Views REQ0103168

Caching Design Thoughts

* Separate thread does regular polling on master content resources(ServiceNow, Mingle, TFS, etc.)
  + Polling intervals will depend on the resource and it’s constraints/requirements.
* Data is collated and stored in local data storage.
  + SQL, Mongo, .xml files, JSON files?
* Web service pulls web page requests from local data storage.

Local Data Storage Design

Store by Product and then Release.

Beneath Release, each org/Product may have different metrics to track. Allow for variable key as well as value, yet keep enough organization that the information is displayable without schema re-write. Something like the .xsl reporting design of CruiseControl.Net ?

Node.js – Backend Service

Angular.js – Communication with backend

jQuery.js – Interact with Dynamic HTML.

Backbone.js – Implementation of MVVM.

Mustache.js – Dynamic HTML assistance tools and shortcuts.

Ember.js – Layout templating framework, using a router to display different content based on the URI.

Supports data binding to data controllers.

Handlebars.js – Advanced HTML templating.

ServiceNow JSON Query:

<https://my.service-now.com/change_request.do?JSON&sysparm_action=getRecords&sysparm_query=number=CHG0073269>

To get records belonging to Configuration item field, read:

https://my.service-now.com/cmdb\_ci\_service.do?JSON&sysparm\_action=getRecords&sysparm\_query=name=<Name in Configuration item: field>

as cmdb\_ci\_service table.

Then read passing in ‘sys\_id’ from the result

https://my.service-now.com/change\_request.do?JSON&sysparm\_action=getRecords&sysparm\_query=cmdb\_ci=<sys\_id from previous query>

Mingle REST Api, get ‘Release’ type cards:

[https://my.mingle.com/api/v2/projects/trunk\_release/cards.xml?filters[mql]=Type%20is%20Release](https://my.mingle.com/api/v2/projects/trunk_release/cards.xml?filters%5bmql%5d=Type%20is%20Release)

https://my.mingle.com/api/v2/projects/trunk\_release/cards.xml?filters[mql]=type = change AND 'Release' = '2015-02-r2'

TFS REST Api:

Here's the url of the custom TFS rest server: <http://10.2.13.44:18080/>

Had to create web service running the TFS Java SDK to provide a Rest service NodeJS could ping.

New TFS Java SDK published 9/21/2014 - <http://www.microsoft.com/en-us/download/details.aspx?id=40785>

There is a REST api in TFS 2013 as of Update 2, but is not the full implementation. And the part I need, workitems, won’t make it into a TFS 2013 Update, but will be there in TFS 2015. For now need to make do with the Java SDK dbak implemented above.

Extending TFS – WorkItem Query using C# SDK – <http://msdn.microsoft.com/en-us/library/bb130306.aspx?cs-save-lang=1&cs-lang=csharp#code-snippet-1>

ExpWeb – Mingle Pages

Cars – TFS – Group=’CARS’ AND (ReleaseType =’Major Release’ OR ‘Minor Release’ OR ‘Maintenance’)

BFS – TFS – Group=’BFS’

AirINT – TFS – Group=’Air’ AND Title Contains ‘AirINT’

Date Fields:

t2DRStartTime Actual DR Start

t2DRStartDate Planned DR Start

t2DREndTime Actual DR End

t2DREndDate Planned DR End

t2ProdStartTime Actual Prod Start

t2ProdStartDate Planned Prod Start

t2ProdEndTime Actual Prod End

t2ProdEndDate Planned Prod End

Data Required For Each Release Item

* Release Label
* Title
* Description
* StartTime
* EndTime
* Events
  + Type or Title
  + StartTime
  + EndTime
* Contact
* State
* CRQs
  + Production Early Deployment
  + Production Soak
  + Production Deployment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **Mingle - ExpWeb** | **TFS - Cars** | **TFS – AirINT** | **TFS - BFS** |
| **name** | | Grab name field as part of set of all releases from the ‘Release List’ tab at [EWEMingle Trunk Release Page](https://ewemingle.karmalab.net/projects/trunk_release/cards/list?columns=Release+Type%2CRelease+Branch+Creation%2CRelease+Test+Signoff%2CProduction+Soak%2CRTW&filters%5B%5D=%5BType%5D%5Bis%5D%5BRelease%5D&style=list&tab=Release+List). This may be just an iteration of ALL Release Records’ Name field. |  |  |  |
| **description** | | N/A, might pull from CRQ for Prod Deployment |  |  |  |
| **start\_time** | | Calculate from Events, picking lowest Event StartTime |  |  |  |
| **end\_time** | | Calculate from Events, picking the highest Event EndTime |  |  |  |
| **contact** | | Do I need this? Doesn’t seem to be a clear field. |  |  |  |
| **State** | | Parse CRQs, Have States matching Event Types, then appending CRQ Status (Cancelled, Draft, Pending Team Approval, Pending Approval, Approved for Implementation, Implementation in Progress, Closed). |  |  |  |
| **Events** | **Type or Title** | Parse from Properties:  Integration Test Signoff, RTW, Production Soak, Release Branch Creation, and Release Test Signoff |  |  |  |
| **StartTime** | date = rr.properties.property.name=”<Type>”..value  time = rr.properties.property.name=”<Type> Start Time”..value |  |  |  |
| **EndTime** | date = rr.properties.property.name=”<Type>”..value  time = rr.properties.property.name=”<Type> End Time”..value |  |  |  |
| **CRQs** | **Production Early Deployment** | rr.properties.property.name=”CRQ – Production Early Deployment”..value |  |  |  |
| **Production Soak** | rr.properties.property.name=”CRQ – Production Early Deployment”..value |  |  |  |
| **Production Deployment** | rr.properties.property.name=”CRQ – Production Early Deployment”..value |  |  |  |

JSON Data Object

release{

label:

title:

description:

owner:

startTime:

endTime:

events {

label:

description:

owner:

startTime:

endTime:

}

change\_requests{

title:

status:

number:

startTime:

endTime:

}

}

Mingle – Finding HotFixes:

Mingle Internal:

Hotfix Change Requests for {{

value query:

SELECT Name

WHERE Name = THIS CARD.name

}}%

{{

Table query: SELECT number, name, Team,'Change Owner','Test Lead','Change Category'

WHERE Type = 'Change' and Release = THIS CARD.name and 'Change Category' = 'Hotfix'

}}

ServiceNow Sign-Off query

<https://my.service-now.com/sysapproval_approver_list.do?sysparm_query=sysapproval%3D71e316def51e6580a5f26e338956eaee%5Estate!%3Dnot_required%5EORstate%3D>

TFS REST api workitem fields:

{ id: 1010316,

project: '5th Column',

systemWorkItemType: 'Release Record',

systemAreaPath: '5th Column',

systemTitle: 'CARS .NET Admin Tool Release',

systemAssignedTo: 'Runa Roy',

systemChangedBy: 'Michael Van Hoff',

systemChangedDate: '2014-10-08 17:58:34 UTC',

systemCreatedBy: 'Michelline Bouvier',

systemCreatedDate: '2014-09-03 02:16:55 UTC',

systemDescription: 'Completed successfully on 09.23',

systemRev: 9,

systemState: 'Closed',

expediaDevLead: 'Gurmit Ghatore',

systemIterationPath: '5th Column\\Q1 Rel 14',

expediaTestLead: 'Krishna Gurung',

expediaTfsPromoteRunbookId: 'd0874360-d4c3-4fde-841e-c4a6a34ef265',

snBusinJust: null,

snImpact: null,

t2ActualDR: null,

t2ActualFeatureComp: null,

t2ActualProd: null,

t2ActualTestSignOff: null,

t2CRNumber: 'CHG0079987',

t2DREndDate: '2014-09-12 07:00:00 UTC',

t2DREndTime: '2014-09-02 19:00:00 UTC',

t2DRStartDate: '2014-09-12 07:00:00 UTC',

t2DRStartTime: '2014-09-23 18:00:00 UTC',

t2FeatureCompDate: null,

t2FuncStressSignOff: 'Yes',

t2Group: 'CARS',

t2HipChatRoom: 'T2 Cars',

t2MingleId: null,

t2MingleUrl: null,

t2NextReleaseMilestone: null,

t2PreReleaseType: 'DR',

t2ProdEndDate: '2014-09-23 07:00:00 UTC',

t2ProdEndTime: '2014-09-12 21:00:00 UTC',

t2ProdMEndDate: null,

t2ProdMStartDate: null,

t2ProdStartDate: '2014-09-23 07:00:00 UTC',

t2ProdStartTime: '2014-09-02 17:00:00 UTC',

t2ProgramManager: 'Jonathan Lin',

t2ReleaseDescr: null,

t2ReleaseManager: 'Michelline Bouvier',

t2ReleaseType: 'Minor Release',

t2SoakCrq: null,

t2StressTestSignOff: 'No',

t2TestSignOffDate: null },

TFS WorkItem Fields:

"Expedia.DevLead", "Dev Lead", "String"

"Expedia.TFS.PromoteRunbookID", "PromoteRunbookID", "String"

"Expedia.TestLead", "Test Lead", "String"

"Microsoft.Sync.ProjSrv.IsLinkedToProjSrv", "Project Server Is Linked", "String"

"Microsoft.Sync.ProjSrv.LastReviewStatus", "Project Server Last Review Status", "String"

"Microsoft.Sync.ProjSrv.LastReviewedDate", "Project Server Last Reviewed Date", "DateTime"

"Microsoft.Sync.ProjSrv.LastSubmitStatus", "Project Server Last Submit Status", "String"

"Microsoft.Sync.ProjSrv.LastSubmittedDate", "Project Server Last Submitted Date", "DateTime"

"Microsoft.Sync.ProjSrv.ProjectName", "Project Server Enterprise Project", "String"

"Microsoft.Sync.ProjSrv.Submit", "Project Server Submit", "String"

"Microsoft.VSTS.Build.IntegrationBuild", "Integration Build", "String"

"Microsoft.VSTS.Common.ActivatedBy", "Activated By", "String"

"Microsoft.VSTS.Common.ActivatedDate", "Activated Date", "DateTime"

"Microsoft.VSTS.Common.Activity", "Activity", "String"

"Microsoft.VSTS.Common.ClosedBy", "Closed By", "String"

"Microsoft.VSTS.Common.ClosedDate", "Closed Date", "DateTime"

"Microsoft.VSTS.Common.Priority", "Priority", "Integer"

"Microsoft.VSTS.Common.StackRank", "Stack Rank", "Double]

Working ["

"Microsoft.VSTS.Common.StateChangeDate", "State Change Date", "DateTime"

"Microsoft.VSTS.Scheduling.CompletedWork", "Completed Work", "Double"

"Microsoft.VSTS.Scheduling.FinishDate", "Finish Date", "DateTime"

"Microsoft.VSTS.Scheduling.OriginalEstimate", "Original Estimate", "Double"

"Microsoft.VSTS.Scheduling.RemainingWork", "Remaining Work", "Double"

"Microsoft.VSTS.Scheduling.StartDate", "Start Date", "DateTime"

"Microsoft.VSTS.Scheduling.StoryPoints", "Story Points", "Double"

"Mirror.Microsoft.VSTS.Scheduling.CompletedWork", "Project Server Completed Work", "Double"

"Mirror.Microsoft.VSTS.Scheduling.OriginalEstimate", "Project Server Original Estimate", "Double"

"Mirror.Microsoft.VSTS.Scheduling.RemainingWork", "Project Server Remaining Work", "Double"

"SN.BusinJust", "Business Justification (SN)", "PlainText"

"SN.Impact", "Impact (SN)", "PlainText"

"System.AreaId", "Area ID", "Integer"

"System.AreaPath", "Area Path", "TreePath"

"System.AssignedTo", "Assigned To", "String"

"System.AttachedFileCount", "Attached File Count", "Integer"

"System.AuthorizedAs", "Authorized As", "String"

"System.AuthorizedDate", "Authorized Date", "DateTime"

"System.ChangedBy", "Changed By", "String"

"System.ChangedDate", "Changed Date", "DateTime"

"System.CreatedBy", "Created By", "String"

"System.CreatedDate", "Created Date", "DateTime"

"System.Description", "Description", "Html"

"System.ExternalLinkCount", "External Link Count", "Integer"

"System.History", "History", "History"

"System.HyperLinkCount", "Hyperlink Count", "Integer"

"System.Id", "ID", "Integer"

"System.IterationId", "Iteration ID", "Integer"

"System.IterationPath", "Iteration Path", "TreePath"

"System.Links.LinkType", "Link Type", "Integer"

"System.NodeName", "Node Name", "String"

"System.Reason", "Reason", "String"

"System.RelatedLinkCount", "Related Link Count", "Integer"

"System.Rev", "Rev", "Integer"

"System.RevisedDate", "Revised Date", "DateTime"

"System.State", "State", "String"

"System.Tags", "Tags", "PlainText"

"System.TeamProject", "Team Project", "String"

"System.Title", "Title", "String"

"System.Watermark", "Watermark", "Integer"

"System.WorkItemType", "Work Item Type", "String"

"T2.ActualDR", "Actual Dr/PreSoak", "DateTime"

"T2.ActualFeatureComp", "Actual Feature Complete", "DateTime"

"T2.ActualProd", "Actual Production", "DateTime"

"T2.ActualTestSignOff", "Actual Test SignOff", "DateTime"

"T2.CRNumber", "CR Number", "String"

"T2.DREndDate", "DREndDate", "DateTime"

"T2.DREndTime", "DREndTime", "DateTime"

"T2.DRStartDate", "DRStartDate", "DateTime"

"T2.DRStartTime", "DRStartTime", "DateTime"

"T2.FeatureCompDate", "Feature Complete", "DateTime"

"T2.FuncStressComments", "Functional Stress Comments", "Html"

"T2.FuncTestSignOff", "Functional Test Sign Off", "String"

"T2.Group", "Group", "String"

"T2.HipChatRoom", "Hip Chat Room", "String"

"T2.MingleID", "Mingle ID", "String"

"T2.MingleURL", "MingleURL", "String"

"T2.NextReleaseMilestone", "Next Release Milestone", "String"

"T2.PreReleaseType", "Pre Release Type", "String"

"T2.ProdEndDate", "ProdEndDate", "DateTime"

"T2.ProdEndTime", "ProdEndTime", "DateTime"

"T2.ProdMEndDate", "ProdMEndDate", "DateTime"

"T2.ProdMStartDate", "ProdMStartDate", "DateTime"

"T2.ProdStartDate", "ProdStartDate", "DateTime"

"T2.ProdStartTime", "ProdStartTime", "DateTime"

"T2.ProgramManager", "PM", "String"

"T2.PromoteComponentComments", "Deployment Units Additional Comments", "Html"

"T2.PromoteGB1", "Promote Golden Bits 1", "String"

"T2.PromoteGB2", "Promote Golden Bits 2", "String"

"T2.PromoteGB3", "Promote Golden Bits 3", "String"

"T2.PromoteGB4", "Promote Golden Bits 4", "String"

"T2.PromoteGB5", "Promote Golden Bits 5", "String"

"T2.PromoteGB6", "Promote Golden Bits 6", "String"

"T2.PromoteOpsSpecLink", "Link to Ops Spec", "String"

"T2.PromotePbP1", "Promote PbP 1", "String"

"T2.PromotePbP2", "Promote PbP 2", "String"

"T2.PromotePbP3", "Promote PbP 3", "String"

"T2.PromotePbP4", "Promote PbP 4", "String"

"T2.PromotePbP5", "Promote PbP 5", "String"

"T2.PromotePbP6", "Promote PbP 6", "String"

"T2.PromoteService1", "Promote Service 1", "String"

"T2.PromoteService2", "Promote Service 2", "String"

"T2.PromoteService3", "Promote Service 3", "String"

"T2.PromoteService4", "Promote Service 4", "String"

"T2.PromoteService5", "Promote Service 5", "String"

"T2.PromoteService6", "Promote Service 6", "String"

"T2.PromoteStressTestSignOffName", "Stress Test Sign Off Name", "String"

"T2.PromoteTestSignOffName", "Functional Test Sign Off Name", "String"

"T2.PromoteVersion1", "Promote Version 1", "String"

"T2.PromoteVersion2", "Promote Version 2", "String"

"T2.PromoteVersion3", "Promote Version 3", "String"

"T2.PromoteVersion4", "Promote Version 4", "String"

"T2.PromoteVersion5", "Promote Version 5", "String"

"T2.PromoteVersion6", "Promote Version 6", "String"

"T2.ReleaseDescr", "Release Description", "String"

"T2.ReleaseManager", "Release Manager", "String"

"T2.ReleaseType", "Release type", "String"

"T2.STComments", "Stress Test Comments", "Html"

"T2.Soak\_CRQ", "Soak CR", "String"

"T2.StressTestSignOff", "Stress Test Sign Off", "String"

"T2.TestSignOffDate", "Test SignOff", "DateTime"

]

reference name, friendly name, data type

"Expedia.DevLead", "Dev Lead", "String"

"Expedia.TFS.PromoteRunbookID", "PromoteRunbookID", "String"

"Expedia.TestLead", "Test Lead", "String"

"Microsoft.Sync.ProjSrv.IsLinkedToProjSrv", "Project Server Is Linked", "String"

"Microsoft.Sync.ProjSrv.LastReviewStatus", "Project Server Last Review Status", "String"

"Microsoft.Sync.ProjSrv.LastReviewedDate", "Project Server Last Reviewed Date", "DateTime"

"Microsoft.Sync.ProjSrv.LastSubmitStatus", "Project Server Last Submit Status", "String"

"Microsoft.Sync.ProjSrv.LastSubmittedDate", "Project Server Last Submitted Date", "DateTime"

"Microsoft.Sync.ProjSrv.ProjectName", "Project Server Enterprise Project", "String"

"Microsoft.Sync.ProjSrv.Submit", "Project Server Submit", "String"

"Microsoft.VSTS.Build.IntegrationBuild", "Integration Build", "String"

"Microsoft.VSTS.Common.ActivatedBy", "Activated By", "String"

"Microsoft.VSTS.Common.ActivatedDate", "Activated Date", "DateTime"

"Microsoft.VSTS.Common.Activity", "Activity", "String"

"Microsoft.VSTS.Common.ClosedBy", "Closed By", "String"

"Microsoft.VSTS.Common.ClosedDate", "Closed Date", "DateTime"

"Microsoft.VSTS.Common.Priority", "Priority", "Integer"

"Microsoft.VSTS.Common.StackRank", "Stack Rank", "Double"

"Microsoft.VSTS.Common.StateChangeDate", "State Change Date", "DateTime"

"Microsoft.VSTS.Scheduling.CompletedWork", "Completed Work", "Double"

"Microsoft.VSTS.Scheduling.FinishDate", "Finish Date", "DateTime"

"Microsoft.VSTS.Scheduling.OriginalEstimate", "Original Estimate", "Double"

"Microsoft.VSTS.Scheduling.RemainingWork", "Remaining Work", "Double"

"Microsoft.VSTS.Scheduling.StartDate", "Start Date", "DateTime"

"Microsoft.VSTS.Scheduling.StoryPoints", "Story Points", "Double"

"Mirror.Microsoft.VSTS.Scheduling.CompletedWork", "Project Server Completed Work", "Double"

"Mirror.Microsoft.VSTS.Scheduling.OriginalEstimate", "Project Server Original Estimate", "Double"

"Mirror.Microsoft.VSTS.Scheduling.RemainingWork", "Project Server Remaining Work", "Double"

"SN.BusinJust", "Business Justification (SN)", "PlainText"

"SN.Impact", "Impact (SN)", "PlainText"

"System.AreaId", "Area ID", "Integer"

"System.AreaPath", "Area Path", "TreePath"

"System.AssignedTo", "Assigned To", "String"

"System.AttachedFileCount", "Attached File Count", "Integer"

"System.AuthorizedAs", "Authorized As", "String"

"System.AuthorizedDate", "Authorized Date", "DateTime"

"System.ChangedBy", "Changed By", "String"

"System.ChangedDate", "Changed Date", "DateTime"

"System.CreatedBy", "Created By", "String"

"System.CreatedDate", "Created Date", "DateTime"

"System.Description", "Description", "Html"

"System.ExternalLinkCount", "External Link Count", "Integer"

"System.History", "History", "History"

"System.HyperLinkCount", "Hyperlink Count", "Integer"

"System.Id", "ID", "Integer"

"System.IterationId", "Iteration ID", "Integer"

"System.IterationPath", "Iteration Path", "TreePath"

"System.Links.LinkType", "Link Type", "Integer"

"System.NodeName", "Node Name", "String"

"System.Reason", "Reason", "String"

"System.RelatedLinkCount", "Related Link Count", "Integer"

"System.Rev", "Rev", "Integer"

"System.RevisedDate", "Revised Date", "DateTime"

"System.State", "State", "String"

"System.Tags", "Tags", "PlainText"

"System.TeamProject", "Team Project", "String"

"System.Title", "Title", "String"

"System.Watermark", "Watermark", "Integer"

"System.WorkItemType", "Work Item Type", "String"

"T2.ActualDR", "Actual Dr/PreSoak", "DateTime"

"T2.ActualFeatureComp", "Actual Feature Complete", "DateTime"

"T2.ActualProd", "Actual Production", "DateTime"

"T2.ActualTestSignOff", "Actual Test SignOff", "DateTime"

"T2.CRNumber", "CR Number", "String"

"T2.DREndDate", "DREndDate", "DateTime"

"T2.DREndTime", "DREndTime", "DateTime"

"T2.DRStartDate", "DRStartDate", "DateTime"

"T2.DRStartTime", "DRStartTime", "DateTime"

"T2.FeatureCompDate", "Feature Complete", "DateTime"

"T2.FuncStressComments", "Functional Stress Comments", "Html"

"T2.FuncTestSignOff", "Functional Test Sign Off", "String"

"T2.Group", "Group", "String"

"T2.HipChatRoom", "Hip Chat Room", "String"

"T2.MingleID", "Mingle ID", "String"

"T2.MingleURL", "MingleURL", "String"

"T2.NextReleaseMilestone", "Next Release Milestone", "String"

"T2.PreReleaseType", "Pre Release Type", "String"

"T2.ProdEndDate", "ProdEndDate", "DateTime"

"T2.ProdEndTime", "ProdEndTime", "DateTime"

"T2.ProdMEndDate", "ProdMEndDate", "DateTime"

"T2.ProdMStartDate", "ProdMStartDate", "DateTime"

"T2.ProdStartDate", "ProdStartDate", "DateTime"

"T2.ProdStartTime", "ProdStartTime", "DateTime"

"T2.ProgramManager", "PM", "String"

"T2.PromoteComponentComments", "Deployment Units Additional Comments", "Html"

"T2.PromoteGB1", "Promote Golden Bits 1", "String"

"T2.PromoteGB2", "Promote Golden Bits 2", "String"

"T2.PromoteGB3", "Promote Golden Bits 3", "String"

"T2.PromoteGB4", "Promote Golden Bits 4", "String"

"T2.PromoteGB5", "Promote Golden Bits 5", "String"

"T2.PromoteGB6", "Promote Golden Bits 6", "String"

"T2.PromoteOpsSpecLink", "Link to Ops Spec", "String"

"T2.PromotePbP1", "Promote PbP 1", "String"

"T2.PromotePbP2", "Promote PbP 2", "String"

"T2.PromotePbP3", "Promote PbP 3", "String"

"T2.PromotePbP4", "Promote PbP 4", "String"

"T2.PromotePbP5", "Promote PbP 5", "String"

"T2.PromotePbP6", "Promote PbP 6", "String"

"T2.PromoteService1", "Promote Service 1", "String"

"T2.PromoteService2", "Promote Service 2", "String"

"T2.PromoteService3", "Promote Service 3", "String"

"T2.PromoteService4", "Promote Service 4", "String"

"T2.PromoteService5", "Promote Service 5", "String"

"T2.PromoteService6", "Promote Service 6", "String"

"T2.PromoteStressTestSignOffName", "Stress Test Sign Off Name", "String"

"T2.PromoteTestSignOffName", "Functional Test Sign Off Name", "String"

"T2.PromoteVersion1", "Promote Version 1", "String"

"T2.PromoteVersion2", "Promote Version 2", "String"

"T2.PromoteVersion3", "Promote Version 3", "String"

"T2.PromoteVersion4", "Promote Version 4", "String"

"T2.PromoteVersion5", "Promote Version 5", "String"

"T2.PromoteVersion6", "Promote Version 6", "String"

"T2.ReleaseDescr", "Release Description", "String"

"T2.ReleaseManager", "Release Manager", "String"

"T2.ReleaseType", "Release type", "String"

"T2.STComments", "Stress Test Comments", "Html"

"T2.Soak\_CRQ", "Soak CR", "String"

"T2.StressTestSignOff", "Stress Test Sign Off", "String"

"T2.TestSignOffDate", "Test SignOff", "DateTime"

Design with events:

* Get Mingle Cards
  + Emit: cards\_loaded
  + On: cards\_loaded
    - Iterate card list creating Release model instances
    - Get ServiceNow records.
      * Emit: record\_loaded
      * On: record\_loaded
        + Parse and load records into Change and Event models

UI Design For SVG

* At least 2 scales/ranges
  + Pixels per day
  + Pixels per hour
  + Zoom in increases pixels/day until pixels/day ===’24’, then transition scale to pixels/hour and set initial scale to ‘1’ (24 pixels in 24 hour day)
  + Zoom out decreases pixels/hour until pixels/hour === ‘1’, then transition scale to pixels/day and set initial scale to ‘24’ pixels.
  + Farthest zoom out will be 1 pixel/day, so 365 pixels would represent 1 year.
  + function zoomIn()
    - Check if transition required.
    - Decrease distance between axis lines ( size of axis rectangle?)
* Using rect (rectangles) to denote axis lines.
  + Create set of rectangles.
  + Set rectangles to only have left-edge borders.
  + Set rectangle’s x1 to be last rectangle’s x2
  + On zoom in, shrink width of all rectangles by scale
* axis\_grid object (extend SVG.Line class/object, looks like there’s a function to extend SVG classes; SVG.extend(SVG.Line, {funcName: function(){} })
  + scale property
    - How many pixels mean a unit
    - Setter should perform self-move and update of mapOffsets
  + pixel\_unit property
    - What unit the pixels mean, day or hour.
  + mapOffsetX1
  + mapOffsetX2
  + mapOffsetY1
  + mapOffsetY2
    - The area of the background that creates a hotzone for clicking in this axis.
    - Use with mouse event offSets to determine if mouse clicked in this axis’ zone.
* Can we extend SVG.Set to make an AxisGrid object? Self-aware, self-formatting, self-scaling…
* Timeline Class
  + Discussion
    - Once have a better handle on graphics, add handlers for setting viewports, doing panning and zooming, etc.
  + Properties
    - public
      * pxHour
        + How many pixels represent 1 hour.
    - private
      * axis
        + TimelineAxisCollection
      * dataList
        + Collection of TimelineData Class
  + Methods
    - GetDate(x)
      * Using the x value location against the scale of the timeline, return the date that the x location represents.
        + Can be used to return what date the mouse is currently hovering over.
* TimelineAxisCollection Class
  + Discussion
    - Manually Add Label Array and Scale, or use a mechanism to determine the best Label set and Scale based on the StartDate and EndDate range.
      * So, if the range covers a day, we show axis in hours.
      * If the range covers 6 months, we show axis in months or weeks.
      * Need heuristic, possibly externally defined, to make the decision to transition up or down in scale.
    - While placing label, use width determination to pick between just shorter and longer label representations.
      * For instance, "12" to "12th" to "01/12" to "Jan 12th" to "January 12th" based on available viewable area (width of axis).
      * Couple with determination to switch from showing hours to showing dates to showing months.
  + Properties
    - public
      * StartDate
        + Date object starting date range covered by timeline.
      * EndDate
        + Date object ending date range covered by timeline.
    - private
      * axisList
        + Object or Array containing the axis that describe a time range.
      * svg
        + SVG object containing the collection of TimelineAxis items.
  + Methods
    - Generate()
      * Create the items in the axisList and add thier svg (by reference) to the collected svg here.
      * Need to make sure that changes in TimelineAxis instances bubble up to this collection, and then up to the Timeline itself.
* TimelineAxis Class
  + Discussion
    - Can we extend an svg.js class and have it still work well as an SVG element? Say extending the SVG.Line class or maybe the SVG.G class?
    - Change 'label' public and remove SetLabel and GetLabel if we can learn how to define getters and setters in JS.
      * Need the functionality in the background to update the 'svg' object when this value changes.
  + Properties
    - public
    - private
      * label
        + Set to the time represented by the line.
      * svg
        + SVG object defining the elements that comprise an axis.

SVG.Text for the label

SVG.Line for the vertical divider.

* + Methods
    - SetLabel(label)
    - GetLabel()
    - SetDate(Date)
    - GetDate()
    - SetLocation(x)
    - GetLocation()

Sign-Offs and Approvals

CRQ – Approval – Work on CRQ – SOAK – Sign-Off

Approvals

Approve the work on a CRQ.

SOAK Sign-offs

Sign-off is captured log events during SOAK and their frequency. Increased frequency generates a card that is sent to the responsible team. Team either signs-off that it’s alright or a shipstopper is created if the increased frequency is an unexpected issue.

Test Sign-offs

Get details on process from RelMan

EOS – Web Application – Michael Nash

Deployment Tool

May have tracking of real-time state of environments and CIs.

## Use Cases

Releases

..How upcoming Releases are Scheduled through the next fiscal unit

....Month

....Quarter

....Year

..How often is a Product shipping.

..What changes went into a past Release.

Changes

..Which are in progress

..Status

....Sign-off

....Approvals

..Filter changes by

....Product

....Release

....Status

......Signed-off

......Approved

....Functional Group (ie Operations, DepEng, etc.)

....User

..Show potential conflicts with a given change

....Other changes scheduled for the same time

....Incidents(???) currently in progress.

..Show changes within a given timeframe for planning purposes.

..Show today's scheduled changes.

....Work starting today

....Delivery (Work ending) today

....Due today

..

Change Freeze

..Show Emergency Change Freezes as an Incident

..Show Impacted Changes

....To know which to suspend

....To know which to reschedule

Deployment

..Historical analysis of past deployments,

....How Long Did It Take

....How many issues encountered.

....What kinds of issues were encountered.

Environment

..Server rotation state for a given CI

....??? More specific on what a 'CI' is...

..Product (service) Health for a given CI

..What version/what release is in a given environment.

....Show by Product

......Where it's installed and what version it is.

......Show by Version

........What version of a product is installed where.

....Show by Environment

......What's installed in it and what version it is.

## Data Associations and their Sources

Data

..Product - Configuration File

....Releases belonging to - TFS, Mingle, JIRA

....Currently Installed Locations - ServiceNow, Seiso

......Environments - Seiso

......CIs - Seiso

..Release - TFS, Mingle, JIRA

....Changes belonging to – Mingle Cards, ServiceNow \*\*\*\*Release association??

....Deployments belonging to - ???ServiceNow CRQs

....SOAK Sign-off – Mingle or ServiceNow

....Test Sign-off - \*\*\* Ask RelMan

..Deployments - ??? ServiceNow CRQs

....Schedule - ??? ServiceNow CRQs

....Issues/Incidents associated - ???Manual Entry, Release Manager

..Changes - ServiceNow

....Approval - ServiceNow

....Schedule - ServiceNow

....Contacts as Pertinent

......Approver - ?ServiceNow

......Owner/Requestor - ServiceNow

......Implementer - ServiceNow

..Incidents - ??? Manual Entry, ServiceNow, Webpage Parsing – Ask RelMan

....Change Freezes - Manual Entry

......Schedule - Manual Entry

......Impacted Changes - Compare to ServiceNow Change Schedules

....Outages - Manual Entry, ServiceNow

......Scheduled

......Emergency

..Environment – Seiso, EOS - ???Michael Nash

....Server Rotation State (Need more info to know how best to categorize.) - EOS

....Products Installed

......Version/Release Installed

## Views Targeting Use-Cases

General Browsing View

..List of Products

....Product Instance

......Name

......Schedule

........Research: Do we know or care about the Product's Lifecycle?

......List of Releases

..List of Releases

....Release Instance

......Name

......Product Belonging To

......Schedule

......List of Changes

......List of Incidents

......List of Deployments

........Research: Can there be more than one for each Release?

......List of Environments

..List of Changes

....Change Instance

......Amount Complete or if Pending

......Title

......Sign-off State

......Approval

......Contacts

......Colliding Incidents

..List of Incidents

....Incident Instance

......Title

......Schedule

......Current State

........Pending

........In Progress/Fixing

........Fixed

......Type

........Change Freeze

........Outage

..List of Deployments

....Deployment Instance

......Elapsed Time

......List of Incidents

..List of Environments

....Environment Instance

......List of CIs

......List of Products

..List of CIs

....CI Instance

......Update Level

........Research: Can we track the update level of a machine

..........What Updates were Applied and When they were Applied

..........Pending/Missing Updates

Can We Generalize Views to Leverage Browsing Views To Customize?

Can we combine "Currently Pending" and "Planning" Views by customizing the Time Range?

Currently Pending View - What's Happening Now

..Uses

....See What Work is Planned Now or over a Time Range

......See What is being Worked on Currently

....See What Incidents are Active Now or over a Time Range

....See What is Impacted over a Time Range

....See What is Already Planned over a Time Range

....Predict and Avoid Conflicts

......See Also Planning View's Event Request Feature for Finding Open Windows

....See Approval State at end of Time Range, up to NOW (can't predict approvals)

..Allow Date to be Changed to show "What Will Be Happening" or "What Happened"

..Allow Date Range to be Used to show "What Will Happen Over The Range"

..List of Products

..List of Releases

..List of Changes

..List of Incidents

Planning View

..Uses

....Review Release Schedules

....How Often a Product is Releasing

....Given a Time Length and Resources Required, Return Open Windows of Time

....See What Changes Shipped in a Release

..Show List of Products and their Releases Scheduled Over a Period

....Drill Into List of Changes in a Release

....Drill into List of Deployments in a Release

..Provide Event Request Feature

....Set Details such as Length of Time and Resources Required

....Returns Scheduling Windows Where Time and Resources are Free

....Research: Can we detect secondary Windows?

......Low collision Window, where something with lower priority could bump to free that Window.

..Provide Export of Managerial Data Type With Data

....Text Table

....CSV

....Excel??

....Clipboard??

Current Resource State

..Uses

....See What Changed In Given Resource Over A Time Range

....Given a Resource, Get Its State

......See What version of a Product is Live and Where

......Get Current Server Rotation State

........Need to understand this better to be sure it fits here.

..Select Resource

....Product

....Release

....Change

....Environment

....CI

..Select Time Range

....Returns Details Over Time Range

......Changes Implemented

......Deployments

......Updates Applied

......Updates Missing

..Show Diff of Environments (This might be a v2 Feature...)

....Drill into Diff of CIs

......Show Drive Diff

......Show Update Diff

......Show Application Install Diff

Outliers

..See how long Deployments for a Product take.

....Two-level analysis required.

......Product->Releases->Deployments->Lengths

..Validate sign-off status.

....Is this possible?

## Database Redesign

* <http://www.white-skies.com/2013/02/how-to-set-up-mongoose-mongo-schemas.html>
* Schema List
  + Product
    - name
  + Product2ServiceMap
    - product
    - service
  + Product2ReleaseMap
    - product
    - release
  + Release
    - name
    - foreign\_id
    - description
    - start\_time
    - end\_time
    - state
    - hip\_chat\_room
  + Release2ServiceInstanceMap
    - release
    - service\_instance
  + Release2EventMap
    - release
    - event
  + Release2CodeChangeMap
    - release
    - code\_change
  + Event
    - name
    - description
    - start\_time
    - end\_time
    - actual\_start\_time
    - actual\_end\_time
  + CodeChange
    - name
    - description
    - foreign\_id
    - category
    - signoff
    - team
      * relman
      * implementer
      * pm
      * test
      * dev
  + Service
    - name
    - description
    - group
    - platform
    - scm
    - type
    - owner
      * first\_name
      * last\_name
      * username
      * email
  + Service2ServiceInstanceMap
    - service
    - service\_instance
  + ServiceInstance
    - data\_center
    - environment
    - load\_balancer
  + ServiceInstance2NodeMap
    - service\_instance
    - node
  + Node
    - name
    - health\_status
    - ip\_addresses
      * ip\_address
      * endpoint
        + port
        + protocol
        + rotation\_status
        + rotation\_type
    - machine (Only available if load each node additionally, do we want that?)
      * name
      * fqdn
      * ip\_address
* Keys
  + Product
    - name
  + Release
    - product
    - name
  + Event
    - product
    - release
    - name
  + Change
    - product
    - release
    - name
  + Service
    - product
    - name
  + Service Instance
    - key
  + Node
    - name
      * Guaranteed unique globally per W. Wheeler

### Data Population Using References With String Ids:



### Data Population Using Map Documents:



### Comparison (GSR-Generated String References vs MD-Mapped Documents)

Averages (ms):  
----------------------- GSR ---------- MD ------ Ratio (MD/GSR)  
Writes Started: ------ 3291 -------- 5459 ------ 1.659 (66% increase)  
Writes: ------------ 579255 ----- 3036004 ------ 5.241 (over 5 times longer)  
Reads: ---------------- 224 -------- 4728 ----- 21.107 (over 20 times longer!!)

Averages (m:ss.ms):  
------------------------ GSR ---------- MD  
Writes Started: ---- 0:03.290 ----- 0:05.458  
Writes: ------------ 9:39.255 ---- 50:36.004  
Reads: ------------- 0:00.224 ----- 0:04.728

### Data Population Using findOneAndUpdate

Previous GSR testing used .save(), always saving. Trying .findOneAndUpdate instead to see if there is savings. Further test might be to used find, then only write if diff detected.



Odd, what I didn’t expect to change, actually improved, the ‘find’ query speeds. But everything else as slower, even after the database was fully populated.

### Data Population Using find, then object compare, then save if change detected.

Taking one more stab, this time I’ll do a find, compare the result to what we just generated, then only save if they are different. Will the get and compare speed save over .save()’ing everything?



Only minor savings over just saving every time, with a maintenance overhead, so the ROI doesn’t appear to be there for find-compare-save over just saving.

### Mingle Language for Code Change Query

h1. Change Cards

{{

release-dashboard

where: " type = change AND 'Release' = THIS CARD.name "

columns: 'Name','Change Owner','Test Lead', 'Team', 'Change Category','Gated Change', 'Release Test Signoff'

status-columns: 'Release Test Signoff'

status-success: ...

status-in-progress: ...

status-failure: ...

status-not-applicable: ...

}}

### Mingle Language for SOAK Events

h1. Soak Events

h2. Events Above Threshold (Requires Explicit Signoff)

{{

table

query: SELECT 'number', 'EventCode', 'CardName',

'CardOwner', 'SourceName', 'Release-Next Count',

'Release-Live Count', 'NormalizedEvent%', 'Release Test Signoff'

WHERE type = 'Soak Event'

AND Release = THIS CARD.name

AND 'AboveThreshold'= 'Yes'

ORDER BY 'CardOwner','NormalizedEvent%' desc

}}

h2. Other Events Observed During Soak Below Threshold

{{

table

query: SELECT 'number', 'EventCode', 'CardName',

'CardOwner', 'SourceName', 'Release-Next Count',

'Release-Live Count', 'NormalizedEvent%', 'Release Test Signoff'

WHERE type = 'Soak Event'

AND Release = THIS CARD.name

AND 'AboveThreshold'= 'No'

ORDER BY 'CardOwner','NormalizedEvent%' desc

}}

### Mingle API Examples

https://ewemingle.karmalab.net/api/v2/projects/trunk\_release/cards.xml?filters[mql]=type = change AND 'Release' = '2015-02-r2'

# Michael Nash

## Use Cases

### Most Pertinent Use Cases

Operations need to confirm what has changed in the environment over the last n hours or minutes

Operations have observed an uptick in events and need to determine what changed in the environment

Depeng want to know what version of a product is currently live (seiso)

RelMan want to know the server rotation state or service health for a given CI (seiso)

### Remaining Use Cases

Leadership would like to review the release schedule for a product/stack for the next quarter

An emergency change freeze has been imposed and RelMan need to see and suspend the planned changes for the next n hours

We need to reschedule the changes that were stopped or did not ship due to an emergency change freeze

AppEng/DepEng need to know if their change is approved before they commence shipping

Leadership want to know how frequently a product/service is shipping

Before AppEng/DepEng start a change, they need to know if there are any other changes scheduled that may conflict with their change

RelMan would like to know the status of a change (pending test sign off, approved, etc)

Depeng want to know what work they have scheduled today

A change needs to be rescheduled, we need to validate the new schedule has no conflicts

Leadership or operations want to know which changes shipped in a particular release (change scope)

AppEng/Depeng want to know if there are any incidents currently in progress (before implementing a change)

Leadership/Operations want to know which changes are currently in progress

RelMan want to know how long the last n service deployments actually took

Leadership would like to validate the sign off status for a particular change

Where a change may be a CRQ/CHG, AWS deploy, config/content, A/B test change, etc.

## Which Brings Up These Questions

Willie Wheeler – Talk to him about Seiso.

How do I detect what versions of a product are installed in the environments?

Seiso to have soon. A week or two.

What is the best way to detect what products and components are installed to what environments and servers?

Low ROI.

How are the environments aligned? Wholly in Product? Wholly in a Component (as in one component per server)?

Is there a way to detect changes to a server instance? Through state monitoring and history? Through installer/deployment logging?

Best way to detect Health of a service? Seiso? IsWorking URLs?

## Seiso URIs

Get Services by key <https://seisoapi.idx.expedmz.com/v1/services/airint?view=instances>

* Get Services By Product
  + AirINT: <https://seisoapi.idx.expedmz.com/v1/services/airint?view=instances>
  + ExpWeb: <https://seisoapi.idx.expedmz.com/v1/services/expweb?view=instances>
  + CARS: cars-na and cars-eu
  + BFS: too many to list
    - Use Group filter maybe?
    - <https://seisoapi.idx.expedmz.com/v1/service-groups/bfs> has array of all services belonging to BFS.

## ServiceCatalog

<http://servicecatalog/>

<http://phel0121c52dd20/ServiceLib/HTMLClient/default.htm>

# Angular Prototyping

* Try an overall controller to handle globals (the data) for the entire page, making the remaining controllers subs with access to overall controller’s variables.
  + Can we put a controller on the <body> element? Yes we can per CodeSchool example!!!
* How to share data collections across controllers.
  + Understand $scope
  + Explore global data management class.
* How to have other controls update when another control changes.
  + Explore $watch.
  + Explore ng-change.
* How to read shared data as needed and have dependent controls update on change.
  + Figure out how to trigger other controllers to populate.
  + If there’s an overall data-manager, to have it update and return the right data collections.

# Gadget Framework Concept

## Freeform Notes

* Gadgets
  + Draggable to any location in view
  + Resizeable
  + Snap-to-grid
  + Zoomable?? V2?
  + Some sort of Framework header to allow selection of gadgets and setting global settings.
  + Remember where they are placed and which are shown/hidden
  + Have global settings and per gadget settings
  + Support HTML5 inside gadgets
* Design
  + Use message bus to communicate between gadgets
    - Use to make selections in one gadget reflect to other gadgets.
    - Define common communications
      * Selecting a product in a gadget, sets product for all gadgets.
      * Setting a time range in a gadget, sets that time range for all gadgets.
      * Add mechanism to prevent this when not wanted…
  + Each Gadget module
    - Back-End – Provides data source
    - Middle-ware – AngularJS provider
    - Front-End – Angular directive and html template
    - Added as a named folder in ‘gadgets’ folder.
      * mygadget/init.js
        + name of gadget
      * mygadget/app.js
        + AngularJS code
      * mygadget/template.html
        + Angular directive template
        + Static HTML code for gadget view.
  + Framework API
    - Provide default data access
    - Provide API to common service access
    - Message bus message
      * From
      * To
      * Type
      * Data/Payload/Message
      * Receiver responsible for understanding type and converting Data to Typed data.
        + Gadgets responsible for registering Custom Types if possible.