Updated AI Modeler workflow, updated December 27, 2021

RESEARCH

Before the technical implementation, comes Research. Newer stories that come from Analytics, and are assigned to Modelers, usually required some investigation. That process can touch the following environments:

a. Deeper analysis of the Analytics report
b. Potentially requesting an additional report from Analytics, that focuses on a particular aspect of the story/pattern
c. Watson "brain" analysis: intents, entities and dialog nodes logic
d. Review and study of transcriptions in Splunk
e. NAS drive for review of audio files

IMPLEMENTATION (simple steps)

- **GitHub Desktop:** **Current repository** always has to be "IA-WA-Find-Care-Skill". Not "ia-testing".

- GitHub Desktop: **Current branch** always has to be "IA-WA-Care-Skill_Path1". Path 1 is for T&T. Path 2 is for everyone else. Today, and until Dec. 16, we are using Path 2. See email from Christian Yip, on Nov. 29.

1. **GitHub Desktop:** **FETCH ORIGIN.** Now the most recent code is in my local. To ensure the latest code, select "Pull" from Repository tab.

2. **LAUNCH DEV:** **LAUNCH FROM SCRATCH if it's not active and open. If it is, just refresh.**

3. **LAUNCH DEV:** **PUSH LOCAL → WATSON.**

4. **WATSON:** **MAKE EDITS IN WA. 1 story = 1 commit**

   - Make sure to always allow Watson to train.

5. **WATSON:** **TEST #1** **IN WA**, in "Try it out" panel. Don't forget limits when testing in WA, "Try It Out" panel. There's no real member data. The only thing that can be tested in WA, is if particular intent, entity or dialog response, is triggered. Take screenshots. Upload to the ticket.
   - When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL.

6. **EMULATOR:** **TEST #2** **IN EMULATOR**, that's directed to my own skill, before deploying to Dev. Take screenshots and upload to the ticket. All info is in the yaml file Dan left me.

7. **LAUNCH DEV: PULL WATSON → LOCAL.**
   GitHub Desktop: After pulling, my changes **miraculously appear in GitHub**.

8. **GitHub Desktop: CREATE NEW BRANCH**, from Path 1. Because edits cannot be made to Path 1. Reminder for that is the "Path 1 is a protected Branch. Do you want to switch branches?" warning message, in lower left.

   • Make sure you create branch based on the "Path 1" branch, and not master that it defaults to:

   • 1 story (ticket) = 1 commit

9. **GitHub Desktop: DESELECT EDITS.** If I see edits that are not mine, deselect them by clicking the checkboxes to the left. Delete/discard "skill.json" with nothing in it but the Push Time. See note in Process, "Other stuff" folder, from Jason Holford.

10. **GitHub Desktop: COMMIT:** edits in GitHub Desktop. Then add commit # to the ticket. Make sure I am on MY branch, in GitHub. Sometimes will need to click "Push Origin" blue button, after the commit, in order for "Create Pull Request" button to appear.

11. **GitHub Desktop: PUBLISH BRANCH:** Publish branch in GitHub Desktop. Sometimes, in order to be able to publish a branch, I will first have to click "Push to origin", then "Fetch origin", then wait for Git to "Refresh repository". Click the "Publish branch" blue button. This pushes your edits to the remote repository.
    12. GitHub Desktop: Sometimes have to Push Origin, before can COMMIT. If there are changes I haven't pushed yet, such as was the case with the Billing ticket. Sometimes Git will recognize that, and ask. Or just prevent the Commit.

12. **GitHub Desktop: CREATE PULL REQUEST.**
    • NOW, screen will refresh, and will show "Create Pull Request" blue button.

    • After I click this button, I will be take to GitHub web. There, I will have another "Create Pull Request" button, but this time green.

    • **When you make a Pull Request on GitHub web, make sure the chosen branch is Path1, NOT the Master branch, which it defaults to. This is in the upper left corner of this box, "base" should be set to "IA-WA-Find-Care-Skill-Path1".**

13. **GitHub Web: SEND THE GITHUB WEB LINK** to another Modeler to Review & Approve.

14. **RTC: Add Pull Request number to RTC ticket.** #227 for example.

15. **GitHub Web: MERGE THE PULL REQUEST.**

16. **Octo Deploy: Commit now appears in Octo Deploy, DEV.**

    • This is automatic. The commit # from GitHub, will be in the Octo Deploy name.

17. **EMULATOR: TEST #3 IN EMULATOR**, that's directed to DEV.

18. Octo Deploy: Deploy to QA. after successfully testing in DEV.

19. **EMULATOR: TEST #4 IN EMULATOR**, that's directed to QA.

20. **RTC:** Update ticket. Assig to QA.

21. **PHONE: TEST #5** sometimes Modelers test on the phone too. If time allows.

IMPLEMENTATION (with screenshots and notes)

• **GitHub Desktop: Current repository** always has to be "IA-WA-Find-Care-Skill". Not "ia-testing".

• GitHub Desktop: **Current branch** always has to be "IA-WA-Care-Skill_Path1". Path 1 is for T&T. Path 2 is for everyone else.

1. **GitHub Desktop: FETCH ORIGIN.** Now the most recent code is in my local. To ensure the latest code, select "Pull" from Repository tab.
2. **LAUNCH DEV: LAUNCH FROM SCRATCH EVERY TIME.**
   • Because the tool has to run some Python script every time. Keep the Python file open, and take a look in it, from time to time.

C:\Users\a539616\Userspace Programs\ia-testing\launch_dev_tools.bat

• Note that the Watson timestamp does not sync, in any meaningful way, with the Launch Dev API. Thus, only refer to the Launch Dev timestamp.

• When I just launch "Launch Dev Tools", it won't show the current date, hour and minute. But after I "Push Local → Watson", it will. May need to refresh browser.

3. **LAUNCH DEV: PUSH LOCAL → WATSON.**

4. **WATSON: MAKE EDITS IN WA. 1 story = 1 commit**

   • Make sure to always allow Watson to train.

• Utterances can be linked to an Intent, right in the “Try It Out” panel in Watson. By clicking on the dropdown menu, and choosing the right intent. For example, that’s how I fixed the Gastro ticket. I changed the intent in “Try It Out”, from Irrelevant, to “find_provider_master”. It trained automatically, and worked right away!

• Can also see confidence scores, for certain intents, by hovering over the eye icon.

5. **WATSON: TEST #1 IN WA**, in “Try it out” panel.
   • Don’t forget limits when testing in WA, “Try It Out” panel. There’s no real member data. The only thing that can be tested in WA, is if particular intent, entity or dialog response, is triggered. Take screenshots. Upload to the ticket.
   • Take screenshots. Upload to the ticket.
   • When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL of them, in all environments, and assign ALL OF THEM, for QA to test on the phone.

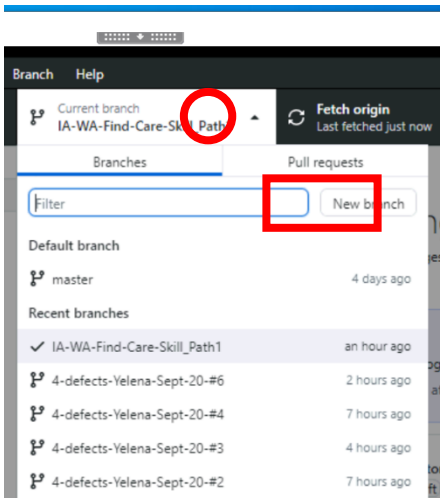
6. **EMULATOR: TEST #2 IN EMULATOR**, that’s directed to my own skill, before deploying to Dev.
   • Copy and paste test results into your RTC ticket.
   • Or, take screenshots and upload to the ticket.
   • When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL of them, in all environments, and assign ALL OF THEM, for QA to test on the phone.

7. **LAUNCH DEV: PULL WATSON → LOCAL.**

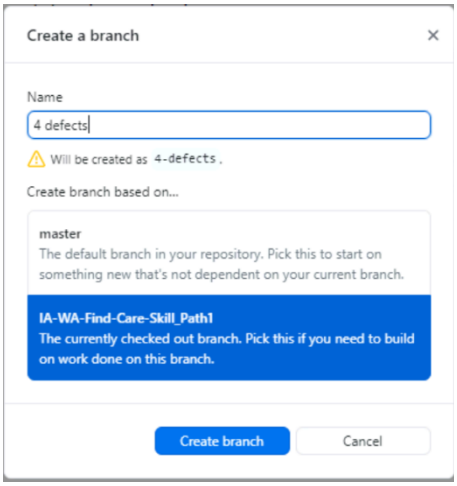
• GitHub Desktop: After pulling, my changes **miraculously appear in GitHub**.

8. **GitHub Desktop: CREATE NEW BRANCH**, from Path 1. Because edits cannot be made to Path 1. Reminder for that is the “Path 1 is a protected Branch. Do you want to switch branches?” warning message, in lower left.

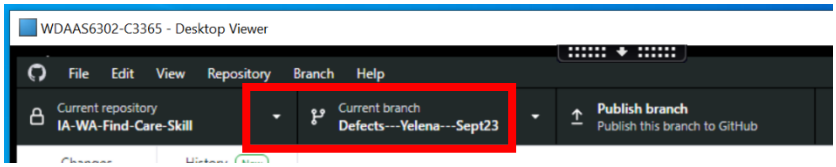
• Create a new branch



• Warning: the “Create branch button” won’t activate, before you start typing the name.
• Make sure you create branch based on the “Path 1” branch, and not master that it defaults to:



• After you create your branch, you should automatically be switched to it:



• 1 story (ticket) = 1 commit

• Note: Branch and commit are 2 different things, that both have to be named with meaningful names. Dan Strathman especially insisted commits be named with a ticket number, and ticket name. Like, “**1774923_Payment**”.

• The branches I like to also add my name and the date. **1774923_Payment_Yelena_Nov5”**

9. **GitHub Desktop: DESELECT EDITS.** If I see edits that are not mine, deselect them by clicking the checkboxes to the left.

• Sometimes I may need to go in, and deselect line by line of code, in the column under “SKILL\find-provider.json” headline. Where lines of code are highlighted in blue.

• Deselect changes that are not mine. By clicking on each. Whatever is not blue, won’t be included. Also, can click-and-drag, to select multiple edits at once.

• See notes I took by hand, on a printout screenshot, from session with Gurjit.

• Intent edits appear on separate line from dialog edits. Don’t deselect Dialog nodes, as I did in the Billing ticket, initially.

• Delete/discard “skill.json” with nothing in it but the Push Time. See note in Process, “Other stuff” folder, from Jason Holford.

• “Stashed changes” – click on the arrow, open up stashed changes, right-click, discard.

• We cannot delete anything from Octo Deploy. But can overwrite. So whatever GitHub edit has a plus icon in front of it, will be added. And whatever has a minus, will be taken out. Like when I overwrote the Billing ticket components (intent and dialog nodes). This happened to be in QA, that it needed 1hr for edits to be applied.

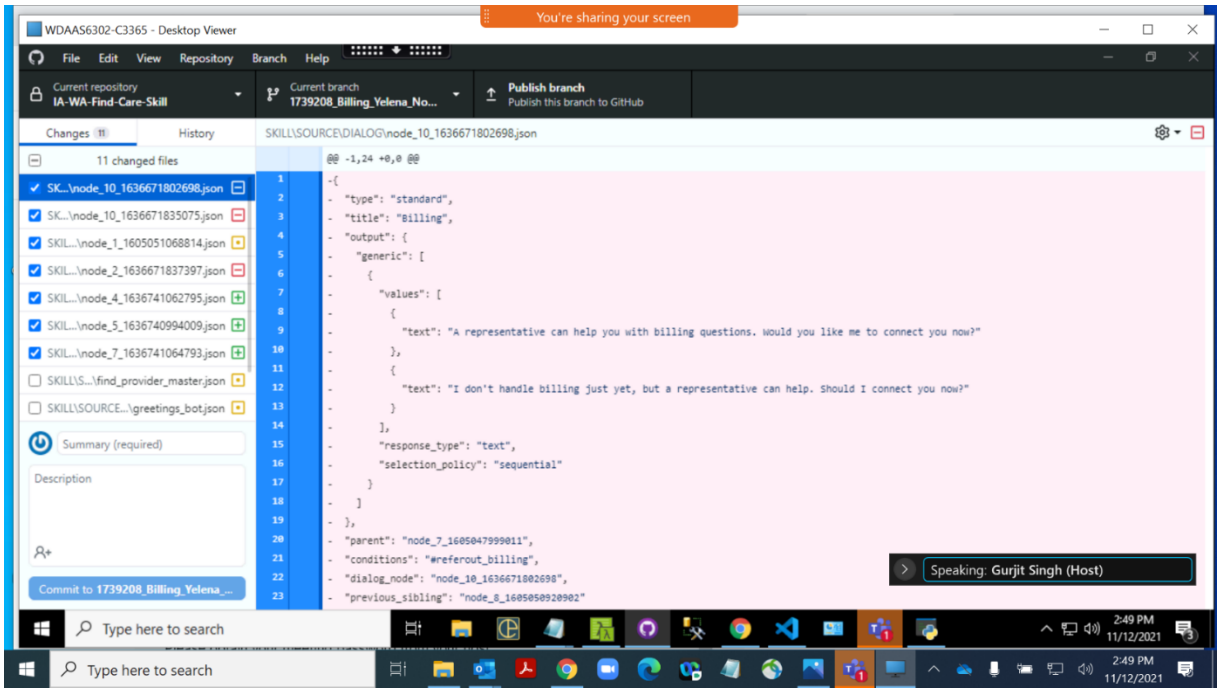
• Sometimes it takes 1hr or so, for edits to be applied in Octo Deploy. Like when I was doing the Billing ticket override.

Regarding the slide below, this was from the Billing ticket mess. This GitHub commit contains both items I want to add, and delete. Because I wanted to delete my initial Octo Snapshot, because intent and dialog were submitted separately. And it was not working, because I probably, unknowingly, deselected a tiny line of code, that was actually needed.

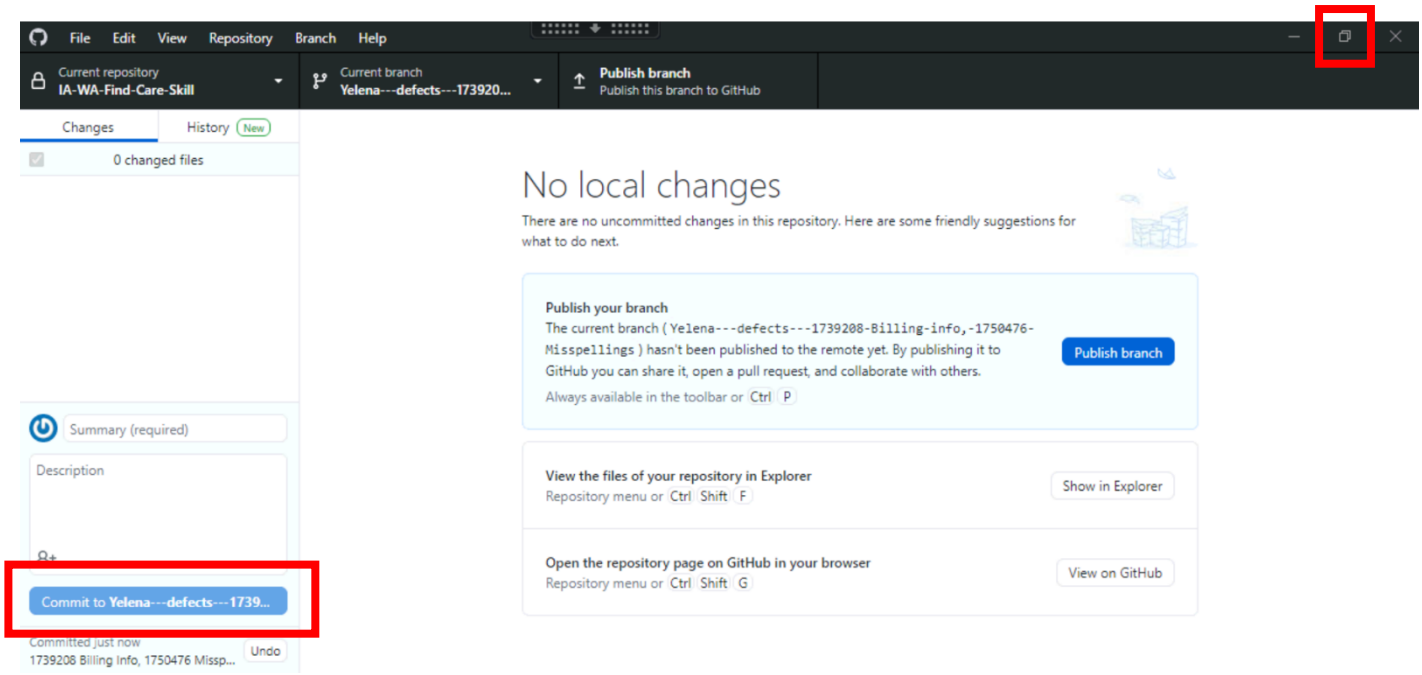
A. **RED:** If the background is red, I am taking that part out. Hence, the “minus” in the box. If the minus is selected, box and the minus are blue. If they are not selected, box and minus are red.

B. **YELLOW:** Not sure what yellow means. It may be stuff that I am neither adding, or taking away, because I did not apply those edits in the first place. Yellow can be a mix of adding and removing, selected and deselected, line by line.

C. **GREEN:** Green pluses are edits I am adding or applying.



10. GitHub Desktop: **COMMIT:** edits in GitHub Desktop. Then add commit # to the ticket.
    • Make sure I am on MY branch, in GitHub.
    • Sometimes will need to click "Push Origin" blue button, after the commit, in order for "Create Pull Request" button to appear.
    • **COMMIT:** Click the "Commit to defect_123456" blue button, in lower left. If you don't see the button, maximize the GitHub Desktop window.



~~GitHub Desktop: Sometimes have to Push Origin, before can COMMIT. If there are changes I haven't pushed yet, such as was the case with the Billing ticket. Sometimes Git will recognize that, and ask. Or just prevent the Commit.~~ Not sure this is still relevant?

11. GitHub Desktop: **PUBLISH BRANCH:** Publish branch in GitHub Desktop. Sometimes, in order to be able to publish a branch, I will first have to click "Push to origin", then "Fetch origin", then wait for Git to "Refresh repository". Click the "Publish branch" blue button. This pushes your edits to the remote repository.

12. GitHub Desktop: **CREATE PULL REQUEST.**
    New page should appear, with big blue "Create Pull Request" button on it. If it doesn't, click on current branch, select "Pull Requests" tab under it. In the middle of that panel, there should be a tiny "Create a pull request" hyperlink.

• GitHub Desktop: **Switch to my branch**, that I just created…If I was not switched automatically, right after creation. Choose "Leave my changes…" in the popout.

• If I had picked up some random edits, in my code, they should show in the "Stashed edits" dropdown. Left hand side of the screen, mid-page vertically. Click on the arrow.

• "Stashed changes" should take over the page, and be the page title.

• “Discard” changes. Popout will show. Choose “Discard” again.

• NOW, screen will refresh, and will show “Create Pull Request” blue button.

• After I click this button, I will be take to GitHub web. There, I will have another “Create Pull Request” button, but this time green.

• **When you make a Pull Request on GitHub web, make sure the chosen branch is Path1, NOT the Master branch, which it defaults to. This is in the upper left corner of this box, “base” should be set to “IA-WA-Find-Care-Skill-Path1”.**

13. GitHub Web: SEND THE GITHUB WEB LINK to another Modeler to Review & Approve.

14. RTC: Add Pull Request number to RTC ticket. #227 for example.

15. GitHub Web: MERGE THE PULL REQUEST.

• GitHub Desktop: Pull request should then show in History tab. If it doesn’t, click Fetch Origin, Pull Origin, and wait for GitHub Desktop, to refresh Repository; ensure I am in Path 1.

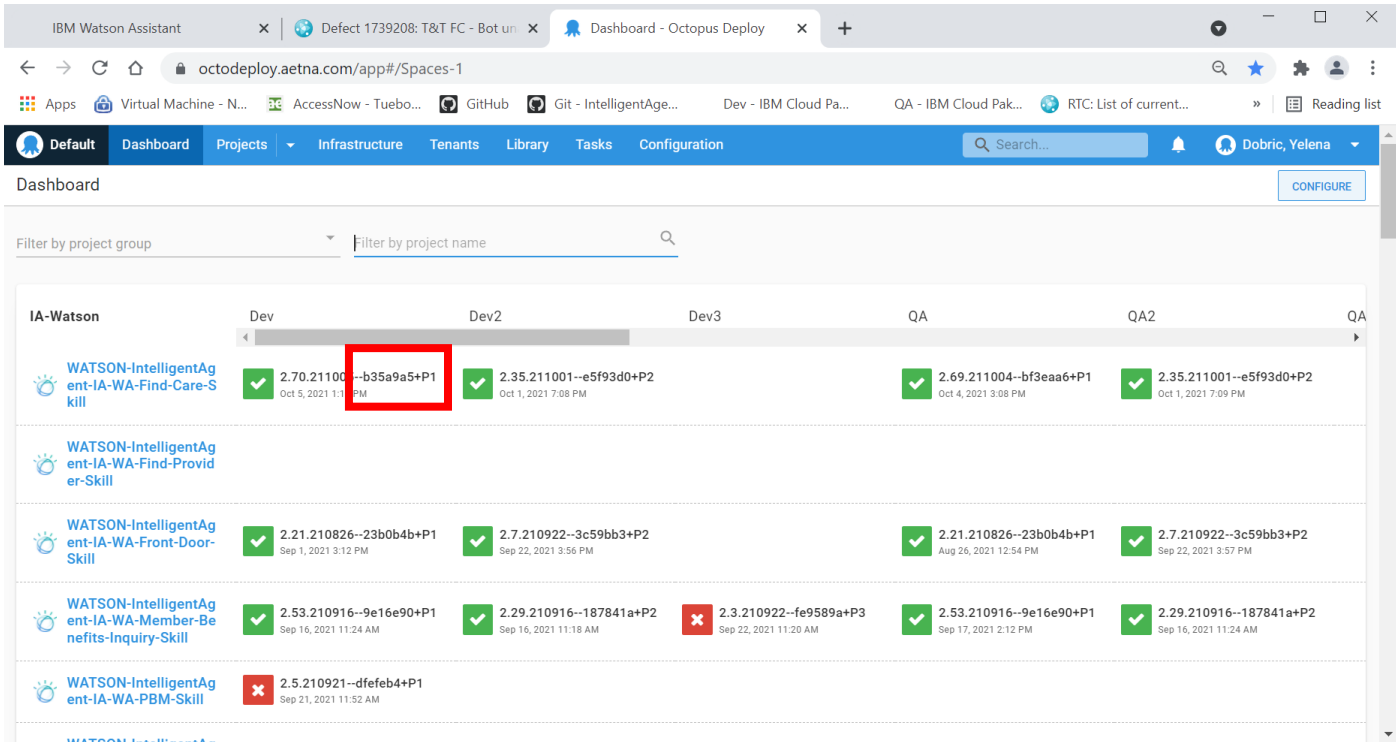
16. Octo Deploy: Commit now appears in Octo Deploy, DEV.

• This is automatic. The commit # from GitHub, will be in the Octo Deploy name. **Insert screenshots.**

• Anytime that the Path 1 or Path 2 are modified, it triggers Octo to deploy to the Dev environment. To the Skill specific to that branch; Dev-1 or Dev-2.

• So, deployment to Octo-Deploy Dev, is automatic. Deployment to Octo Deploy QA, is manual.

• In Octo Deploy, AI Modeler’s commits come together. Usually several commits are pushed to QA at the same time. You can see individual commits, and their numbers, by clicking on the name of the item, and going “inside”, on a separate page, where all the commits that belong to that item, are listed as line items.



• We cannot delete anything from Octo Deploy. But can overwrite. So whatever GitHub edit has a plus icon in front of it, will be added. And whatever has a minus, will be taken out. Like when I overwrote the Billing ticket components (intent and dialog nodes). This happened to be in QA, that it needed 1hr for edits to be applied.

• Sometimes it takes 1hr or so, for edits to be applied in Octo Deploy. Like when I was doing the Billing ticket override.

17. EMULATOR: TEST #3 IN EMULATOR, that's directed to DEV.

• When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL of them, in all environments, and assign ALL OF THEM, for QA to test on the phone.

• Make sure Emulator is pointing to the correct files, as per Path 3 cutover, on 11/19.

• Make sure Emulator is directed to a "configDEV.ini" files, and not "configQA.ini" file, in the .yml file.

• Take screenshots. Upload to the ticket.

• DNIS information for the foreseable future :
1751 / T&T / QA**1**
1752 / T&T / Dev1
1787 / Everest / QA2
callerIntent = findcare unless you're using your side skill then it will be the same callerIntent as before....eg, for me callerIntent = yelenadobric

• Every time I change comments in or out, in the YAML file, I have to SAVE, in order for Emulator to apply the edit!!!

Emulator User Guide

https://github.aetna.com/Channel-MCOE/ia-testing/blob/master/docs/Testing%20Guide.md

• Emulator messages disappear very fast. So if need to take screenshots, take them right away.

• Direct Emulator to Dev1. We don't do Dev2. In "callerInfo_2.yml" file, comment the value we want to work. So no hashtag # on Dev1, if I want to test in Dev1.

• Environments need to be commented in or out. Comment IN means no hashtag next to that environment.

• Uploading screenshots is easier and faster for me, then copy/pasting text, cause I, unlike Amanda, do not have a connection between my Emulator (on my Virtual Machine), and my RTC (that I usually keep open on my physical machine).

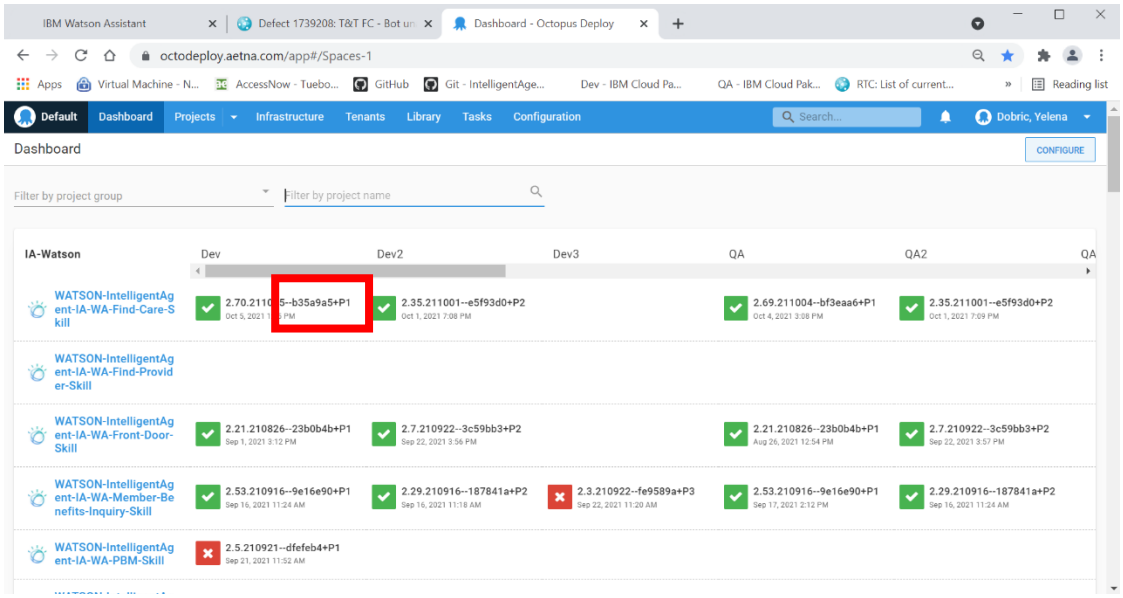
• Or, copy and paste results in each ticket, for tracking.

18. Octo Deploy: Deploy to QA. after successfully testing in DEV.

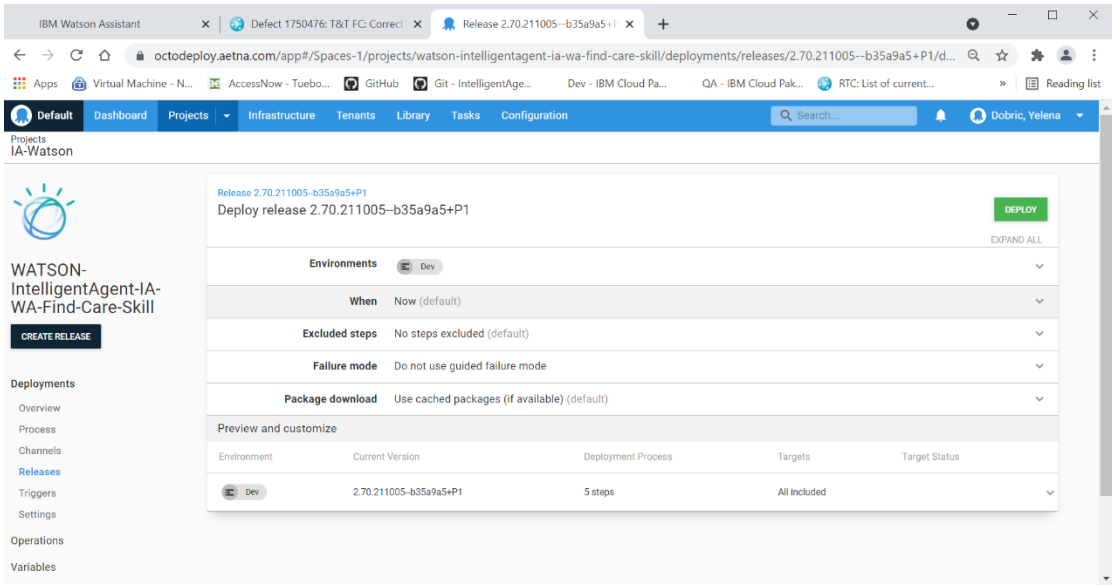
• I can recognize my batch, by the code. The commit code translates into Octo Deploy.

• Additional commits will build into this one item. From Amanda, for example. They form a batch. They are deployed to QA together.

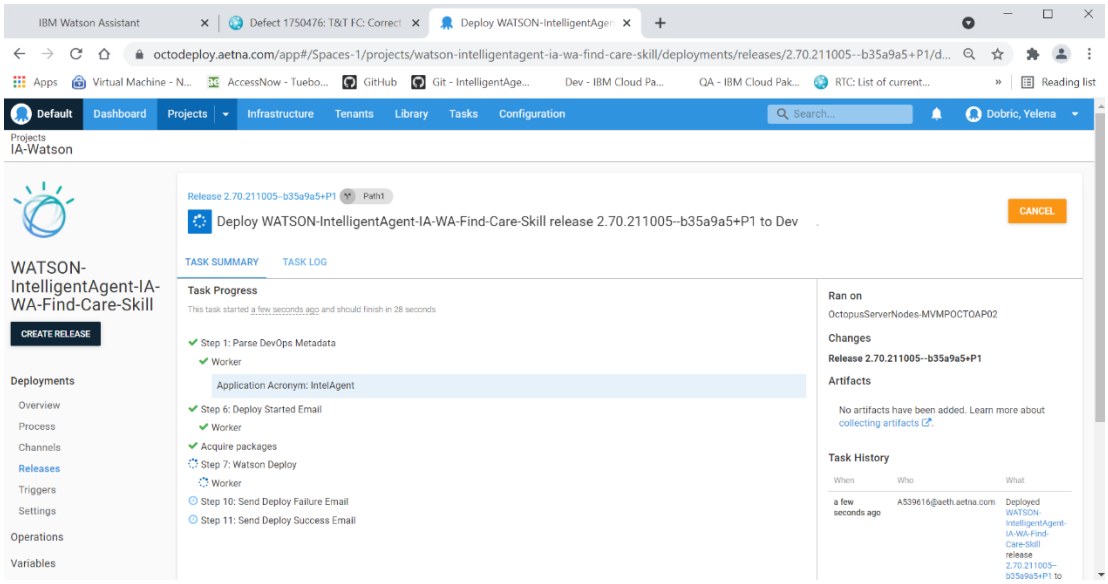
• Can click into the batch, to see all commits in it.



Between these 2 is a screenshot with green button “Deploy to QA”. Click that. Then comes the bottom one, with green “Deploy”.



This is how it looks while it’s processing.



Then the confirmation screen appears. Item is now in QA.

Once my batch shows a green check, it's been deployed to QA.

• We cannot delete anything from Octo Deploy. But can overwrite. So whatever GitHub edit has a plus icon in front of it, will be added. And whatever has a minus, will be taken out. Like when I overwrote the Billing ticket components (intent and dialog nodes). This happened to be in QA, that it needed 1hr for edits to be applied.

• Sometimes it takes 1hr or so, for edits to be applied in Octo Deploy. Like when I was doing the Billing ticket override.

19. EMULATOR: TEST #4 IN EMULATOR, that's directed to QA.

• When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL of them, in all environments, and assign ALL OF THEM, for QA to test on the phone.

• Make sure Emulator is pointing to the correct files, as per Path 3 cutover, on 11/19.

• Make sure Emulator is directed to a "configQA.ini" files, and not "configDEV.ini" file, in the .yml file.

• Take screenshots. Upload to the ticket.

• Every time I change comments in or out, in the YAML file, I have to SAVE, in order for Emulator to apply the edit!!!

• If Emulator says DEV, don't worry about it. If DNIS is #3751, I'm in QA. If I want to fix this, open QA WA, go to Dialog, go to Conversation Start node, change "Deployment_environment" from DEV, to QA-1.

• **#clear in Emulator**, to start over, without exiting. So I don't have to paste in the 3 lines of code, in Emulator notepad file on my desktop.

• Take screenshots. Upload to relevant tickets…If Env says QA.

• If Env says DEV, copy text and paste. Change DEV to QA.

20. RTC:

• Change ticket status from "In Dev" to "Complete Dev".
• Assign to Bobbie. Let them know via other channels: Teams, email, or in Daily Stand Ups.
• When I add multiple synonyms to an Entity, or utterances to Intent, I have to test ALL of them, in all environments, and assign ALL OF THEM, for QA to test on the phone.
• Add "candidate_20211118" (release date). If release date does not exist, start typing.
• Add "#dialog" if there are dialog node changes.
• See the Working Agreement, in T&T Teams Channel, under Wiki tab.
• QA staff tests on the phone and marks Pass/Fail in the RTC ticket.

21. PHONE: **TEST #5** sometimes Modelers test on the phone too. If time allows.

PRODUCTION RELEASES

Prod Release: AI Modeler's responsibilities:

1. **Provide Snapshot** to Gurjit, Avish, Dnyanoba, Neha and Singh, Vikram vjsingh@cvshealth.com. It's the latest file in QA, with +P1 at the end.

• That last snapshot is going to be in QA. As of today, T&T does not use QA2, QA3, etc. We will, in the future.

• Snapshot is composed of GitHub merges, between the Friday after the previous release, and code freeze leading up to the upcoming release. So for example, Friday, Nov. 5, after code-backfill – Tuesday, Nov. 16, EOD.

• To ensure there's no loose ends, compare GitHub merges, with the candidate list in RTC/Rally.

• GitHub merges: Current Repo needs to be Find Care Skill. Current branch Path 1. Review all merges between the Friday after the previous release, and code freeze leading up to the upcoming release. So for example, Friday, Nov. 5, after code-backfill – Tuesday, Nov. 16, EOD.

2. Neha: “We have tasks list and owners for each task. Its called Release checklist in Wiki.”

3. **Fetch session IDs, from Splunk.** Of the calls Jill or Danny make, during the Production Release Checkout process. Provide session IDs, to another Modeler, who is filling out the Excel file.

4. **Populate the Excel file, with PASS/FAIL,** during the Production Checkout.
   • This file is prepared by PO and QA at this point. Its what all test cases and test scripts would be run on the prod day.
   • But filling it out during the Prod Checkout, can be done by a Modeler.

5. Update the release checklist to say that code sync is done. The document is on sharepoint. Product Checkout Scripts. There should be a folder for every release and checklist in that folder. The task owners are responsible for updating the checklist status. Dates gives are recommended dates

6. **Sometimes tests in Emulator in QA, in real time.**

On the day of prod checkout, you would triage problems reported. Whatever you need to do to troubleshoot it – session ids, emulator etc. You would essentially prove if the issue reported is part of the ticket or not.

7. **Code-backfill** is incredibly complex, from my standpoint. We should manage expectations for this, from my end at least.

   • The task owner is responsible to get this activity done and let everyone know when the code freeze can be lifted.

   • See Octopus Deploy document, in this folder, for more information. This task is done by a Dev and Tech lead together.

   • Don’t merge any code, until code-backfill is completed. Can push to a separate branch, though.

MISC BEST PRACTICES & LESSONS LEARNED

Watson is buggy when trying to delete an utterance from an intent

1. Try different browsers OR try changing browser zoom level.

OR

2. Export skill. Open the intent file (source / intents / <name of intent>), in a text editor, and delete from there. Then re-upload the file. It's all text files under the hod, you can manually edit them….OR

3. Discard all edits done in a batch. Start over. Pull new code, do all edits all over again.

Environments we use:

1. RTC. Soon to be Rally.
2. Watson Assistant
3. GitHub Desktop and GitHub Web
4. Emulator
5. Splunk
6. Platform Services – Telephony Connector
7. OctoDeploy
8. NAS drive (Network Access Storage) – where we listen to audio

---

Searching in WA, and applying edits in Excel:

• WA allows for searching intents, entities, dialogs, synonyms, etc. But I have to be in a section, to search it. So for example, have to be in intents, to search intents, etc.

• Upper right, tiny little magnifying glass icon. Click on the icon, to enable search function.

• Or, can download the Excel. In the same area, there's a "Download" icon.

• Control + A to select all, then Control + F to search the Excel.

• Edits can also be applied in Excel, and uploaded back into WA. I think it has to be a .CSV.

Path 3 cutover checklist and best practices:

1. Path cutovers will happen quarterly.

2. DNIS information for the foreseable future :

1751 / T&T / QA1

1752 / T&T / Dev1

1787 / Everest / QA2

callerIntent = findcare unless you're using your side skill then it

will be the same callerIntent as before....eg, for me callerIntent = Strathman

Config.ini files

3. **Update my WA workspace ID in config.ini files.** Do this only where it says "workspace_ID". Not where it says "oidc-client_id=…".

4. So, out of 4 config.ini files, only 1 had 1 workspace id line. The other 3 had 2 workspace id lines. But out of the 2, 1 always started with "w/;" which means that it was commented. So, ignore the commented one. Only update the uncommented one.

5. In config.ini files, where is says "[AUTH] client_id=a261dcf7_3b45…blahblah…Do nothing. The "a261dcf7" just happens to look like an AID, but it's not. It's a coincidence. Plus, this string is 1 digit longer than an AID. My AID is #A539616.

Launch Dev Tool

6. **Do not update my WA workspace ID in Launch Dev tool!** This has nothing to do with Path cutovers. If it works, leave as is.

7. Amanda said that her workspace ID changed, on its own. So she had to update everywhere. Just keep in mind, to check if it changed for me.

GitHub

6. GitHub. T&T still uses Path 1. Path 1 is for T&T. Path 2 is for everyone else. GitHub Path has nothing to do with Path 3 cutover, either. It's just overuse of the word Path, in Intelligent Agent universe.

Questions:

2. Why multiple files? Both .ini and .yml?

3. What about phone testing? Do I need to setup anything for that? If I want to test via phone?