zTPF DevOps Tool Chain for Travelport

Manish Limaye
Suganthi Manonmani

April 2018

TPF Software

Travelport

Agenda

- Need for DevOps @Travelport
  - Why?
  - Project Execution Approach
  - Why Partner with TSI?
- zTPF DevOps Toolchain - Technical Solution
  - Minimum Viable Product & SDLC Gates for the MVP Product
  - 2016: Impact Analysis Tool
  - 2018: Git, Jenkins, Activiti BPM
  - The end to end Process
- Product Demo and Screenshots
- Key Take-Aways and Questions

Need for DevOps @Travelport

Travelport powers technology solutions for the global travel industry.

The power of the platform

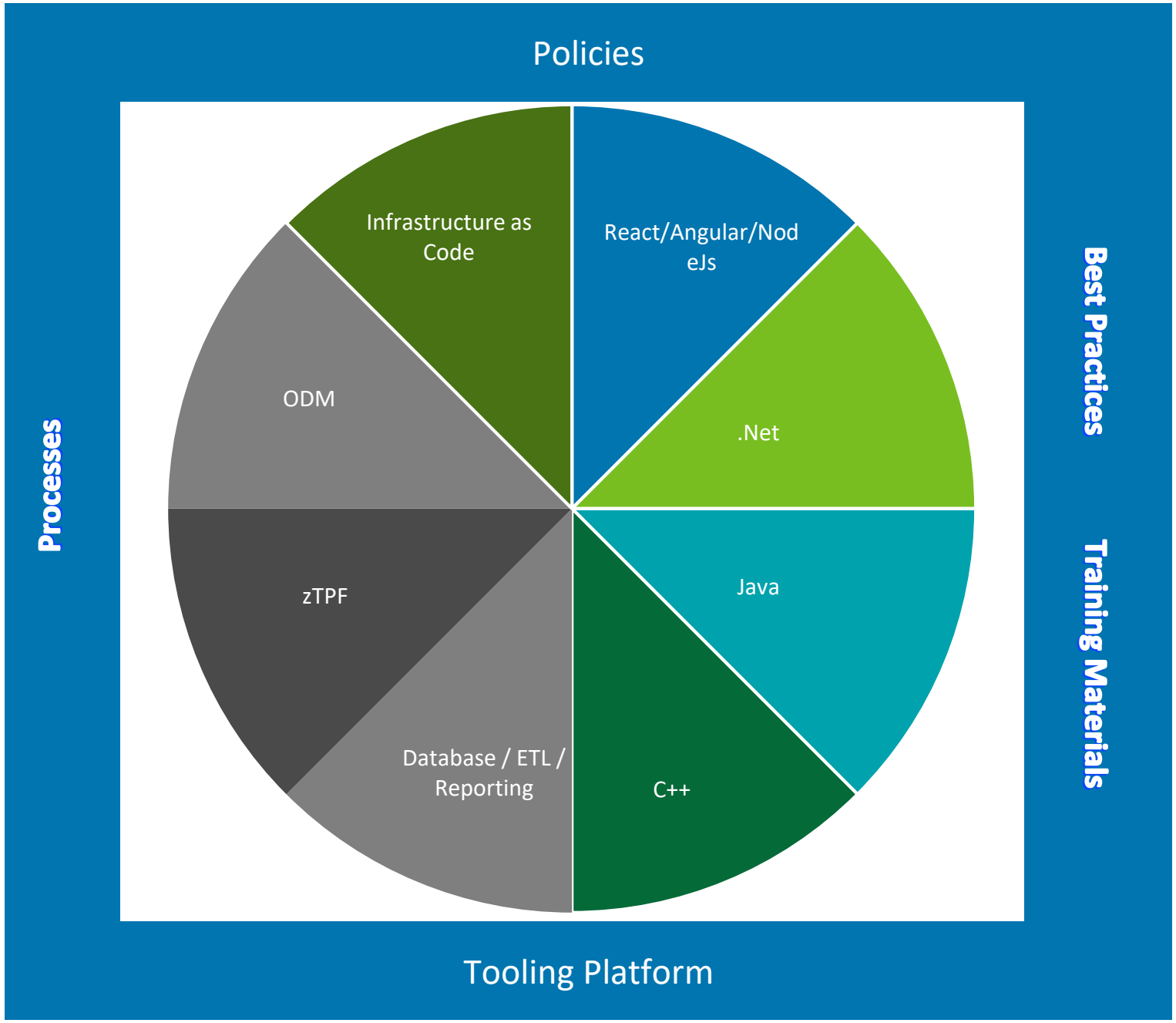
CHOICE PERFORMANCE EXPERIENCES INTELLIGENCE

Engineering Led Development / Testing

- Travelport is on a journey to take software/infrastructure development process to next level by fully embedding engineering practices into all development.

- We have set the bar as “24 hours from ideas to production, no defects, no downtime”. Strategy is to introduce a set of:

- **Policies & Procedures:** know what “excellent” looks like
  - Design & coding standards
  - Repeatable processes
- **Practices:** eliminate waste & rework
  - Design / code reviews
  - Version control & configuration management
  - Stop on test failure
  - Escaped defect analysis & feedback
- **Automation:** make build / test / deploy “free, perfect, and now”
  - Continuous integration
  - Static code quality & security analysis
  - Multiple levels of automated testing (unit / acceptance / load)
  - Release & install packaging
- **Measurement:** objectively measure quality & velocity



Travelport
Redefining travel commerce

- TPF is a mission critical operating system for Travelport, and bringing modern engineering practices to TPF is as important as doing so in Cloud.
- The tools and processes that support the TPF development staff were developed over 20+ years using REXX and CMS Pipes, and not suited for nature of Agile development and CI/CD automation.

Project Execution Approach

- Challenged status quo and start with industry standard best practices, not what we do today.
- Created single DevOps end-to-end workflow for all our TPF Systems in collaboration with SMEs
- Built on top of Open Source Tools.

- Multi year project
  - Wave 1 : POC and Research Analysis tool
  - Wave 2 : End to end MVP focusing on migration of IBM systems code
  - Wave 3 : Migration of TPF application
  - Wave 4: Finish application migration and work on non-MVP features

- The solution under development is owned by Travelport

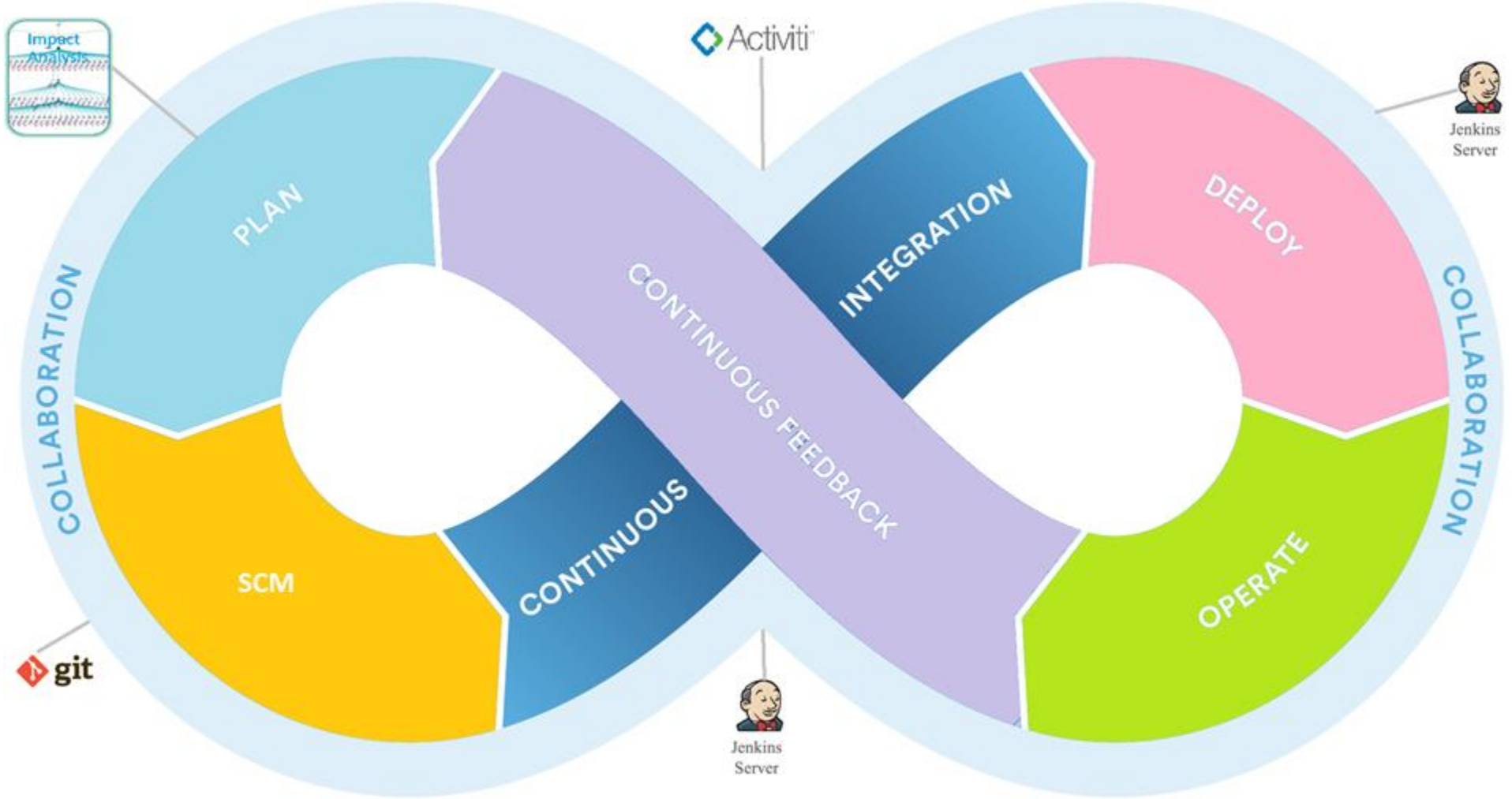
Why Partner with TPF Software Inc.(TSI)?

Travelport chose TSI for building the solution because:

- TSI has shown leadership in DevOps for TPF conducting a DevOps Symposium in January 2016 for the TPF user community.
- TSI's technology and vision closely align with Travelport's long term tools' strategy.
- TSI's brought in new talent to create a right mix of TPF and Open source skills. In addition, their background in TPF tools development has been of great help.
- TSI's POC showed innovation and talent
- TSI has shown their flexibility in working with Travelport in identifying and addressing user requirements.

zTPF DevOps Toolchain – Technical Solution

zTPF DevOps Tool Chain



TPF Software

Travelport

zTPF DevOps Tool Chain - Updates

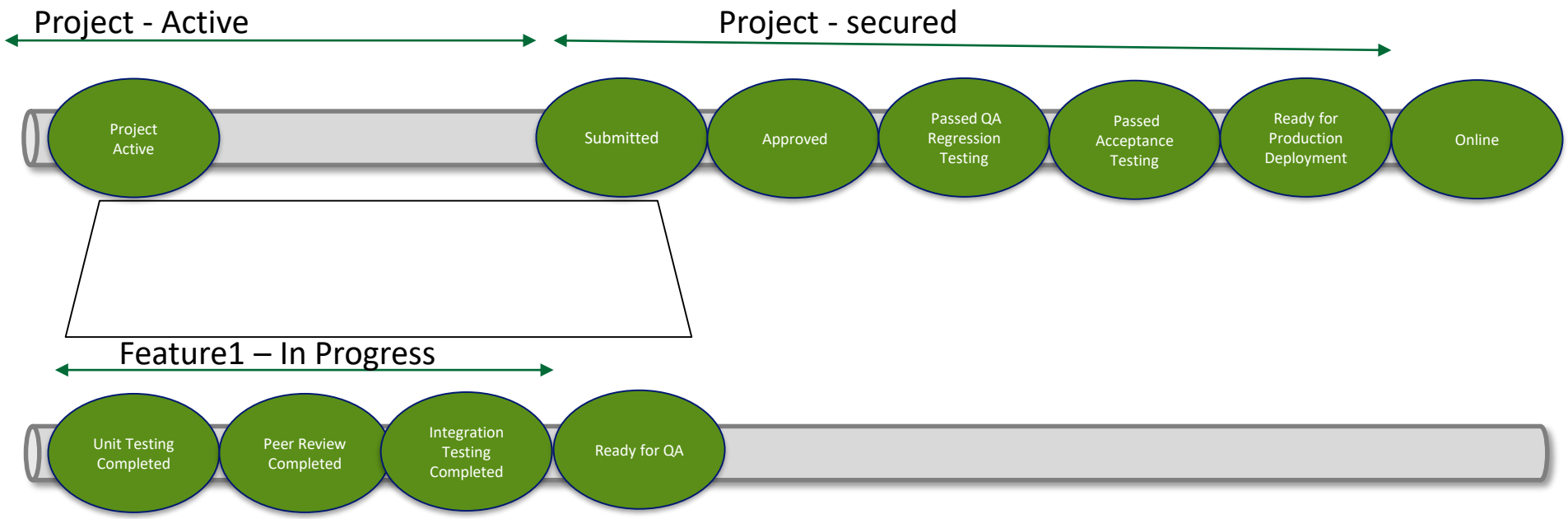
- 2017: Impact Analysis developed and rolled out to users
- 2018: zTPF DevOps Toolchain was used to develop, build and deploy the following in zTPF Production
  - DFDL schema
  - Assembler segments
  - C segments
  - DBDEFs

Minimum Viable Product (MVP)

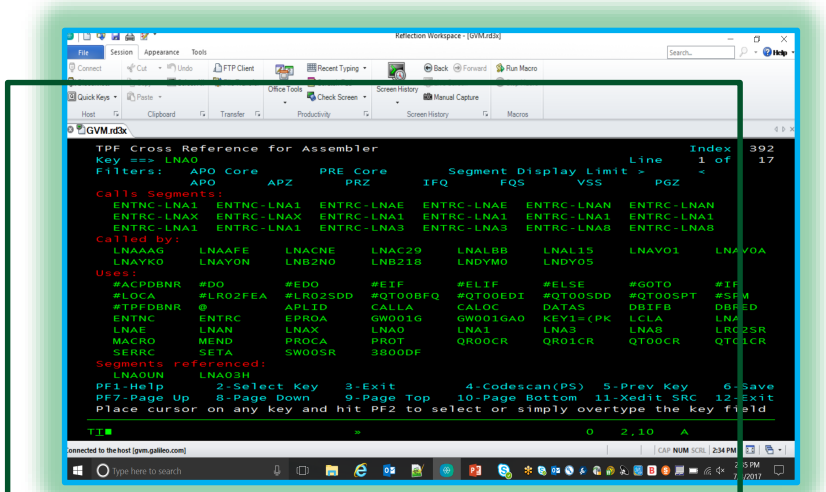
The first version of this zTPF DevOps toolchain which covers the end to end process was deployed in Production on 22nd March and has been rolled out to Travelport's Systems team, File Support team. The Minimum Viable Product(MVP) that is currently in production has the following value adds:

- State-of-art SCM in GIT
- Build using Jenkins
- Positioned for Continuous Integration
- Source Contention Management
- Divergence and Convergence
- File Sync
- Workflow using Activiti BPM
- Peer Review
- Production build and load improvements
- Robust Dependency Analysis from code check-out until loaded to production
- Production online status notification

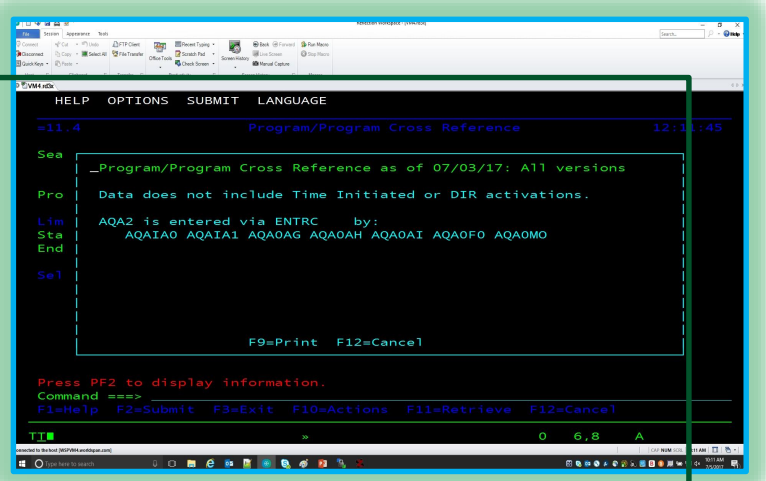
SDLC Gates for a project developed and deployed using the zTPF DevOps Toolchain



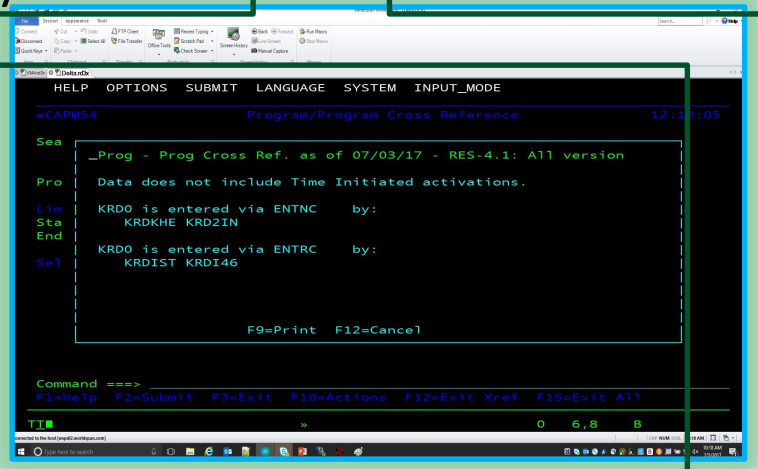
Impact Analysis – current VM based tools (hyperlinks in images)



MX tool used in 1st system



Workbench used in 2nd system

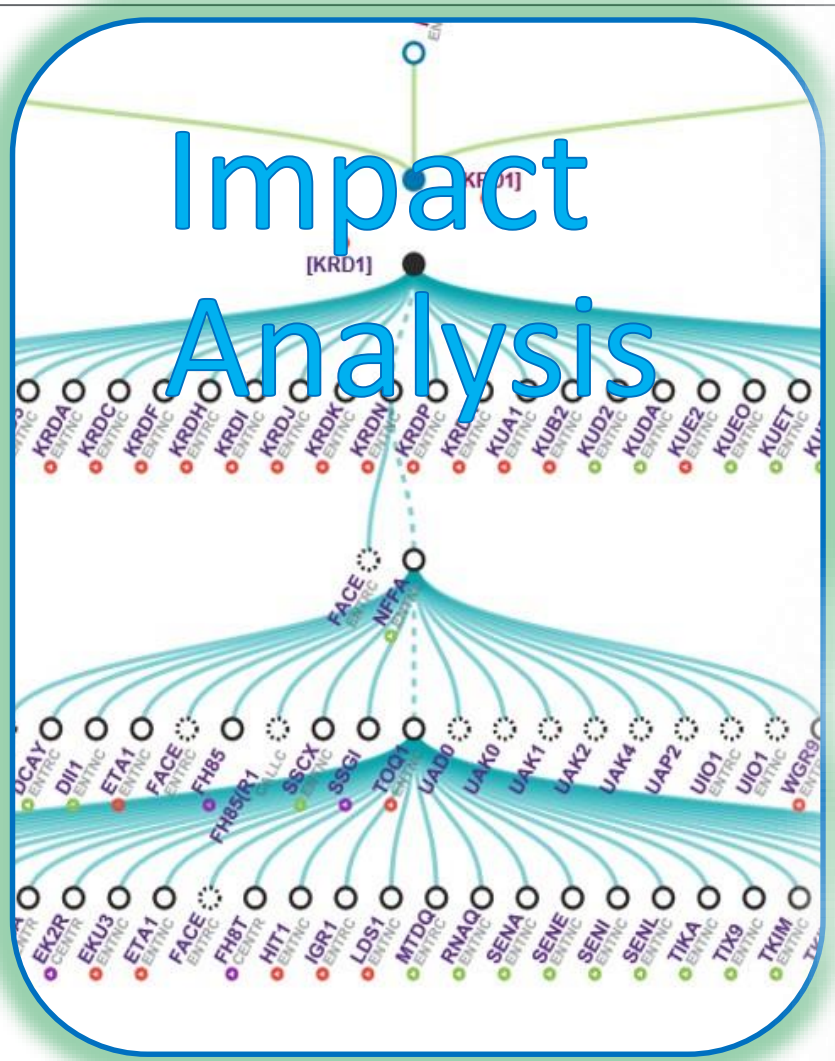


CAPM54 used in 3rd system

TPF Software

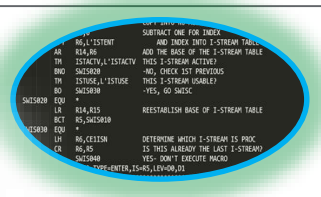
Travelport

Impact Analysis Tool – Developed in 2016 (hyperlinks in images)



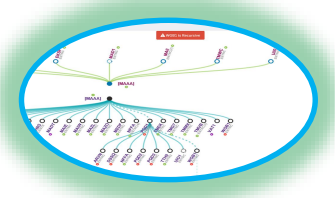
Displays Source

- .asm, .c/.cpp, .sbt
- .mac, .h/.hpp, include
- .mak



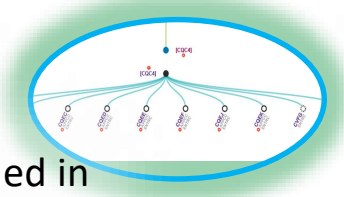
Recursive Calls

- Self referenced nodes
- Repeated nodes



Executable Macros

- Segment calls embedded in macros

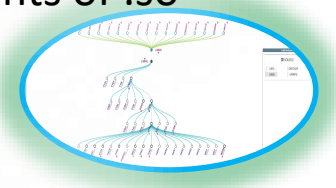


Export Results

- Exports details of analysis
- .pdf, .xls

CSO and BSO xref

- Components of .so



Build Index offline

- Complete Index rebuild
- Refresh index after every deployment

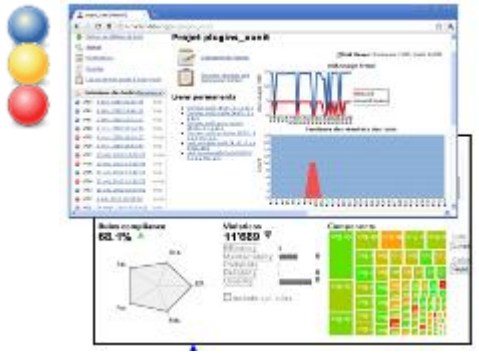
Sabretalk

- .sbt xref
- Includes and includeafs xref

TPF Software

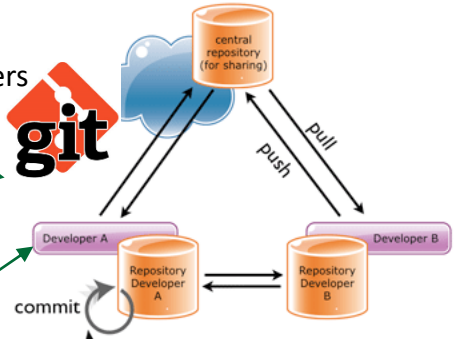
Travelport

The end to end Process using Git, Jenkins, Activiti BPM – Delivered in 2018 (hyperlinks in images)

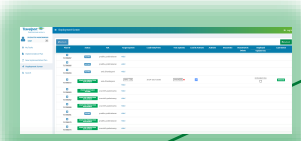


Jenkins

Git Push operation triggers the Jenkins CI process



Continuous Delivery (fire ZOLDR commands via YODA / TOS interface)



Continuous Integration (maketpf build)

Pull source from Git

Push derived binaries to GitLfs

Event – Git Push to Master

Developer VPARS

QA VPARS Functional , Regression

Production zTPF

Oldr file (ftp)

Z/LINUX

Local Workspace

QA Workspace

Deploy Workspace

Config files `maketpf.cfg`
Environment files
Makefiles
Control files `tpf.cntl` & `usr.cntl`
Bldtpf/maketpf `assemble, compile, link`
Loadtpf `.oldr file`

TPF Software

Travelport

zTPF DevOps Tool Chain - Configured Open Source Tools for zTPF

- Built the solution using Open Source Technology
- Impact Analysis tool was built in-house
- Used Industry Standard Practices to solution the problem
- Customized these Open Source tools to address that which makes zTPF Special
  - Build Dependency: large code base
  - Single Source, multiple objects
  - Deployed to multiple target systems
  - Dependencies, Date Audits
- Configured Open Source Tools for zTPF
- Appropriate design patterns to enable appropriate usage in zTPF

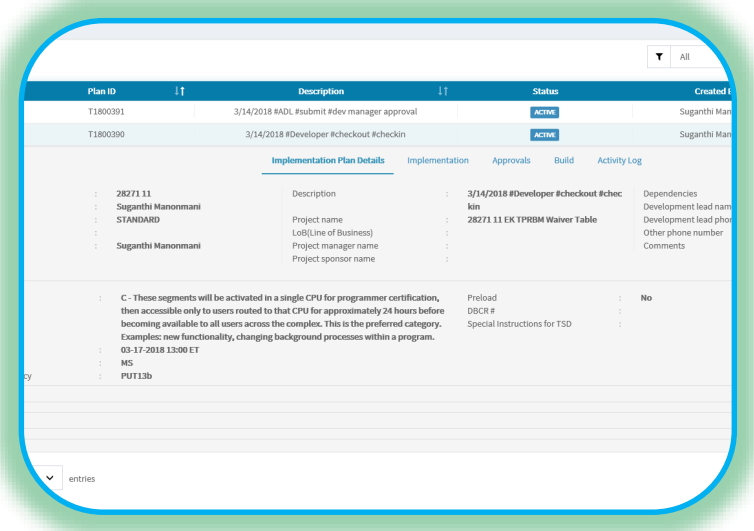
Jenkins
Server

TPF Software

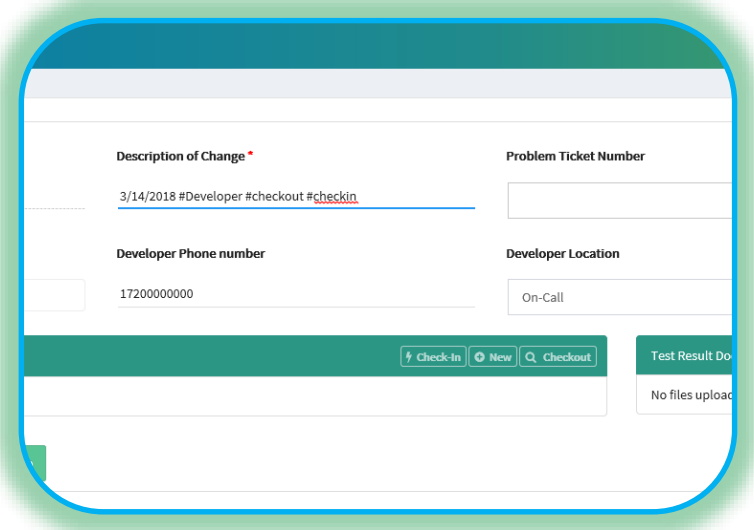
Travelport

Product Demo and Snapshots

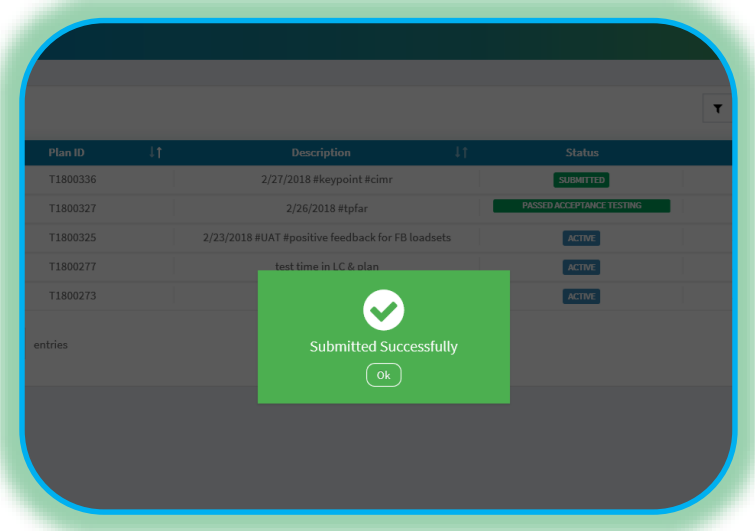
Application Developer Lead(ADL) Role (hyperlinks in images)



Creates Project

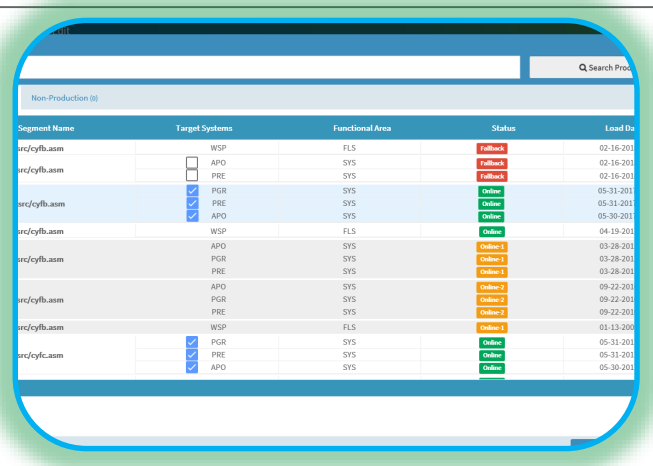


Creates Features – Assign Developers

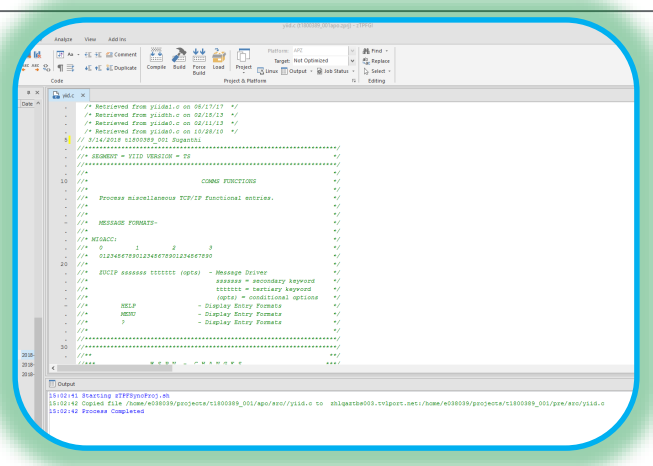


Submits Project Plan (secures)

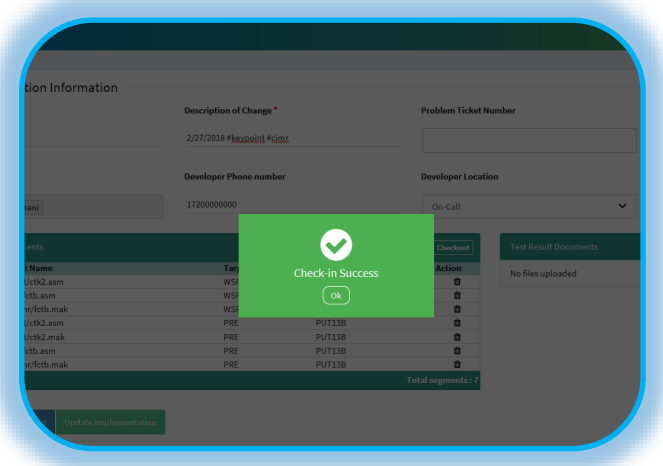
Developer Role (hyperlinks in images)



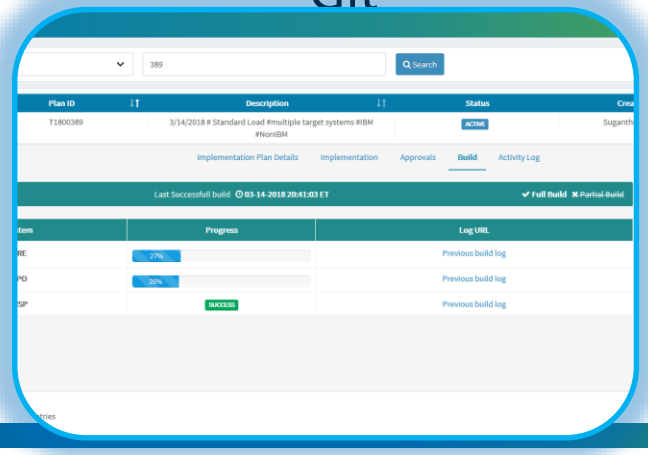
Search and Checkout Source artifact from Git



Update code in IDE, File Sync, Local Build – Performs unit testing



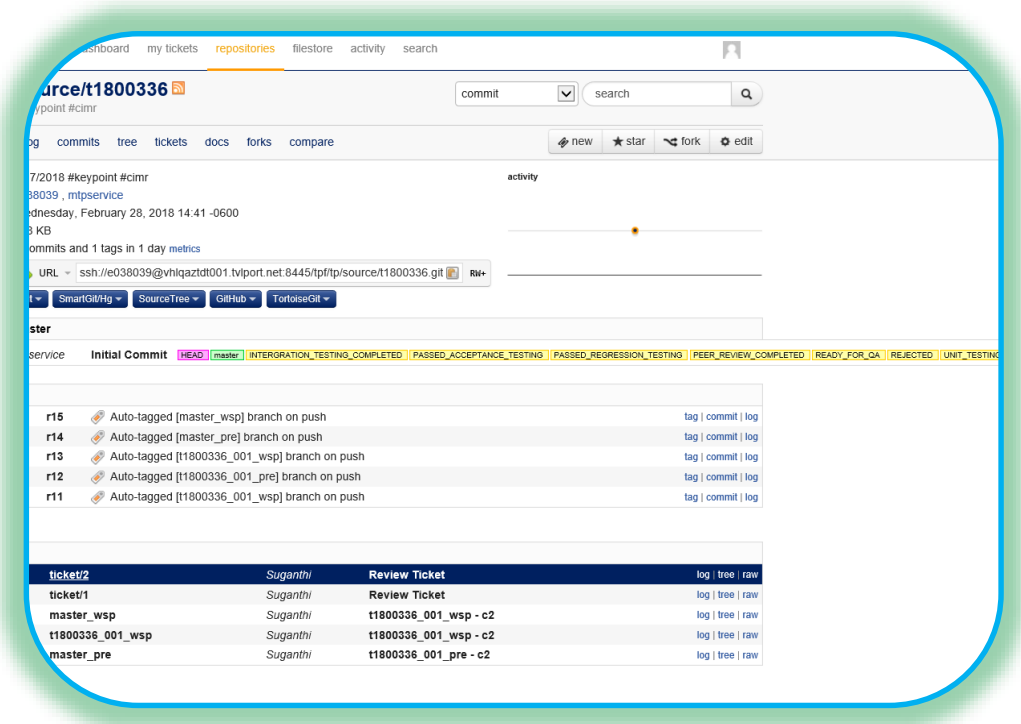
Commit, Checkin Source artifact



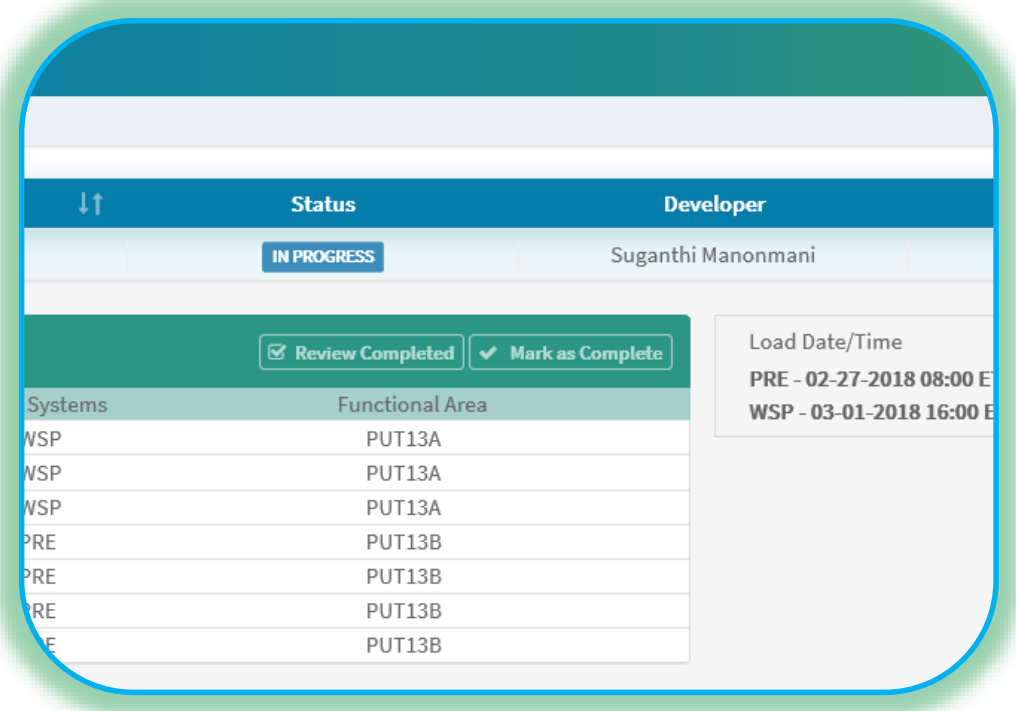
Devl Build –Performs Integration testing

Reviewer Role (hyperlinks in images)

Travelport
Redefining travel commerce

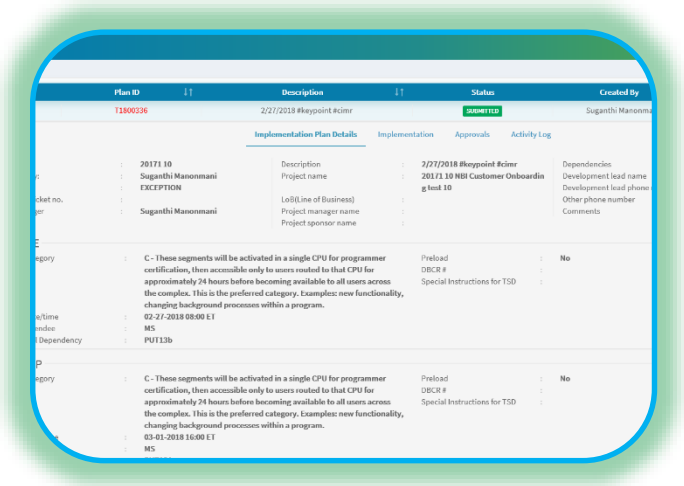


Reviews the source artifacts and documents the comments in Gitblit tickets

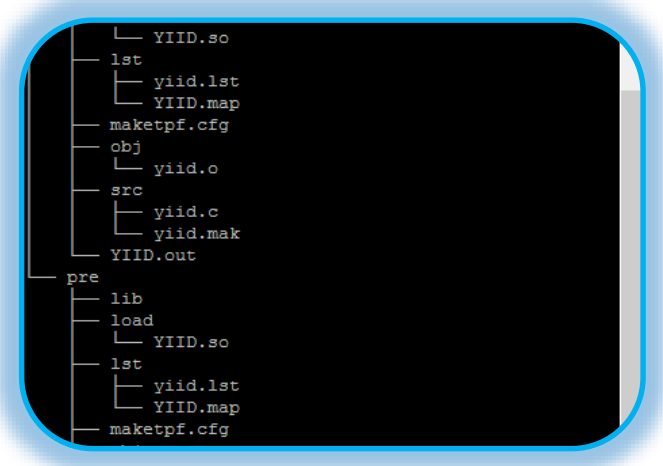


Marks the Feature as Peer Review Completed

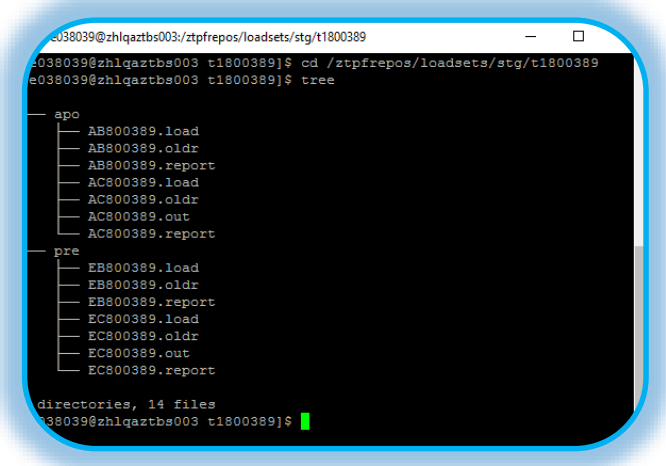
Dev Manager Role (hyperlinks in images)



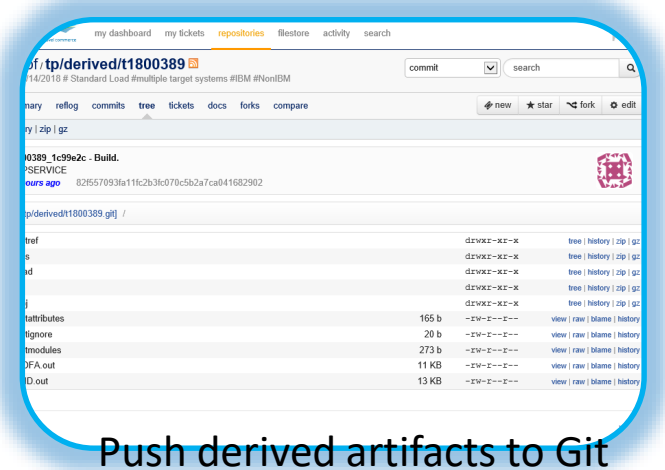
Approves Project



Staging Build done



Staging loadset & fallback loadset generated(same loadset used for QA, Pre-Prod and Production deployment)

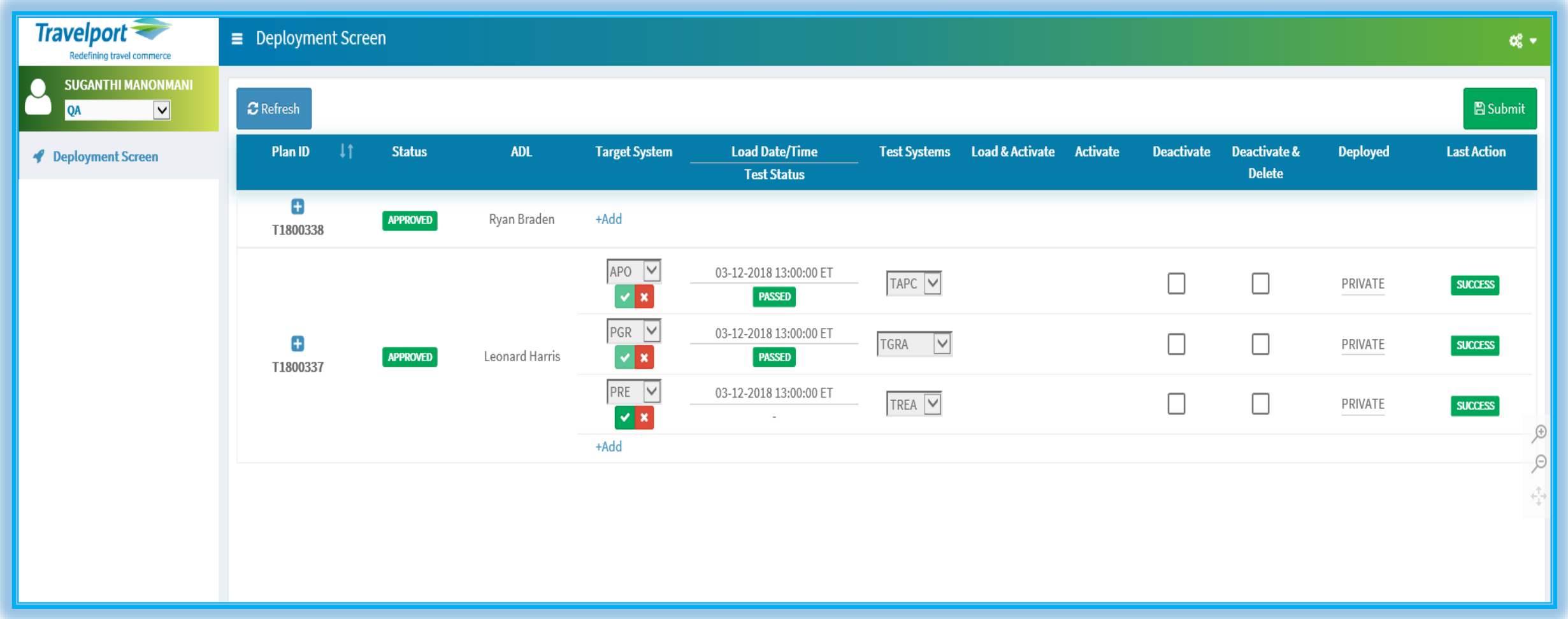


Push derived artifacts to Git Derived repository

QA Role
(hyperlinks in images)

Travelport
Redefining travel commerce

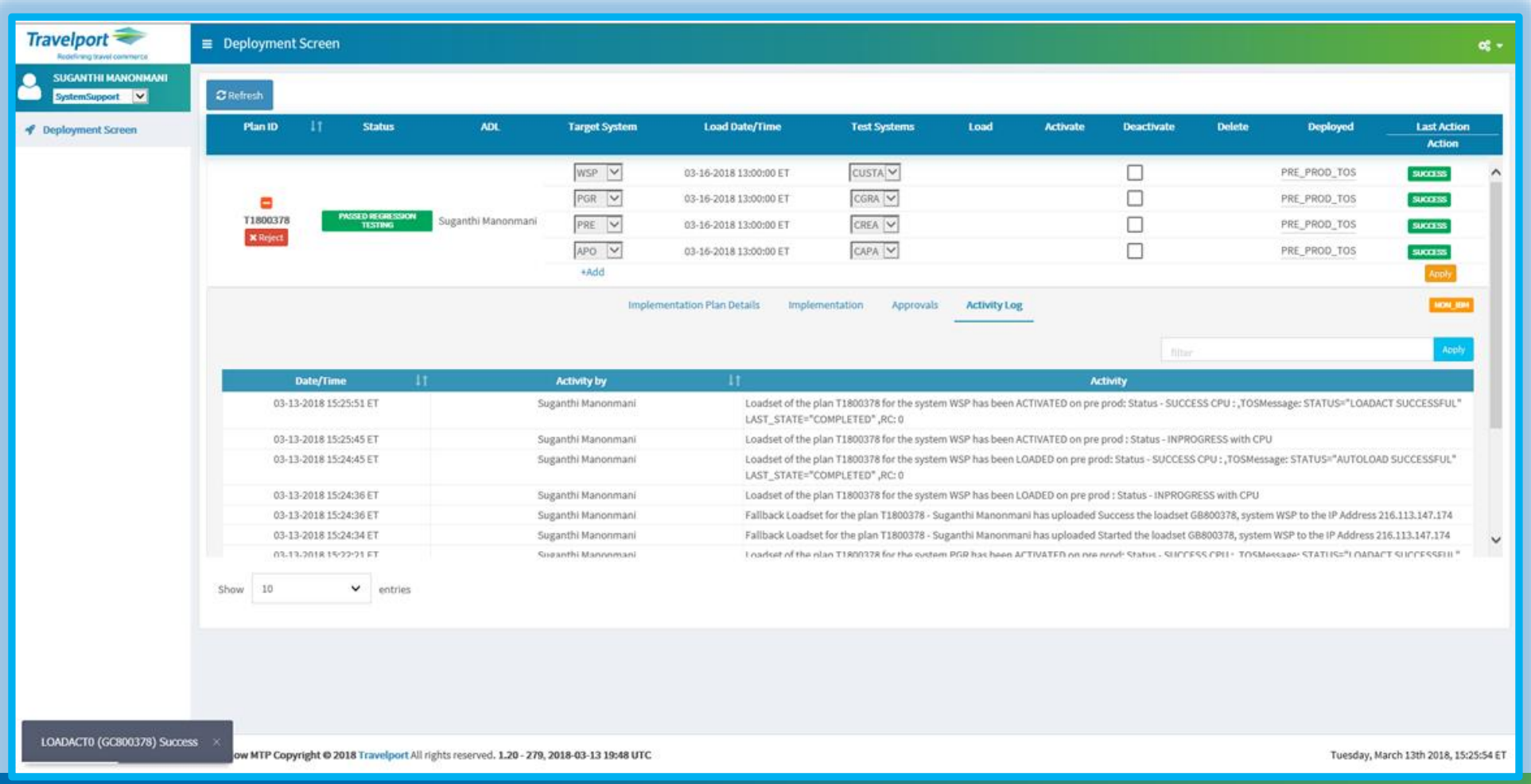
Staging loadsets deployed to QA Regression test systems



Test System Support Role (hyperlinks in images)

Travelport
Redefining travel commerce

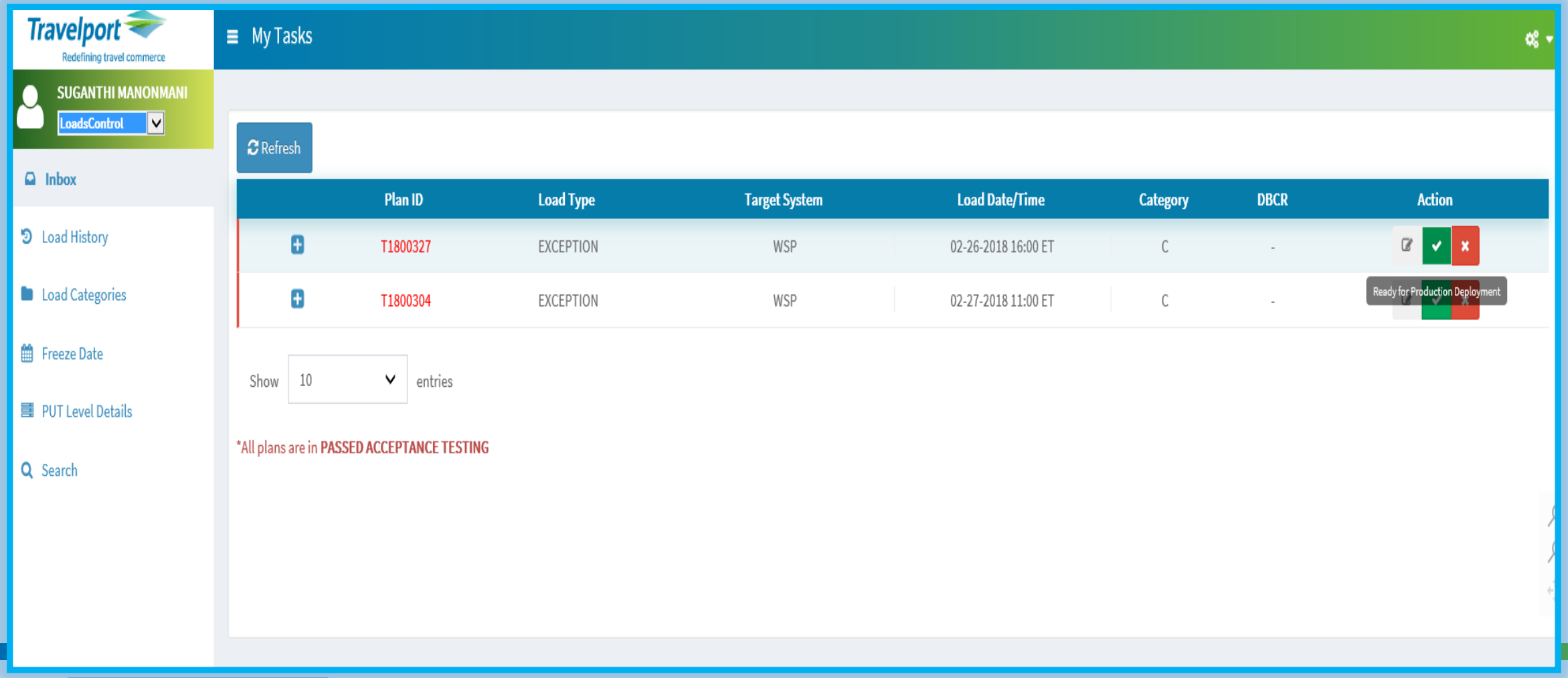
Staging loadsets deployed to Pre-Prod COPY systems



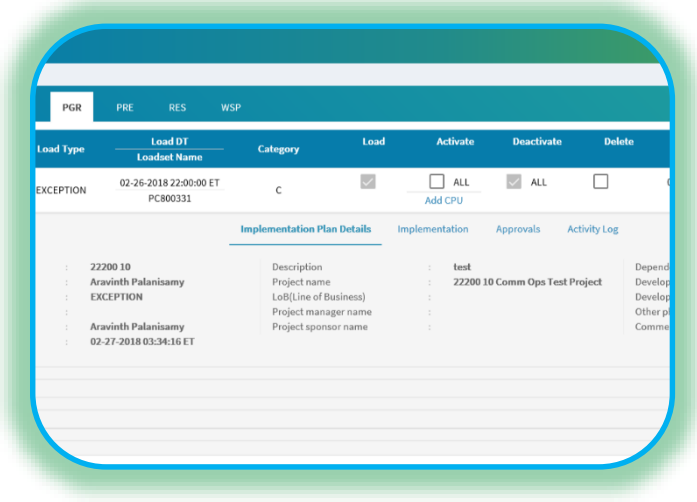
Loads Control Role (hyperlinks in images)

Travelport
Redefining travel commerce

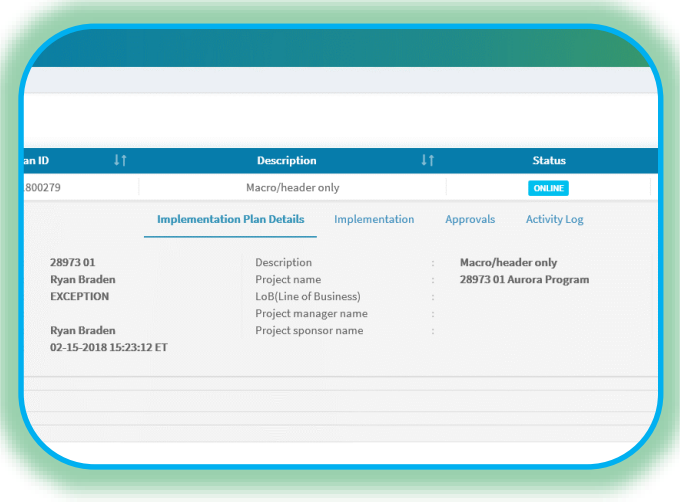
Staging loadsets ftp'ed to zTPF Production System



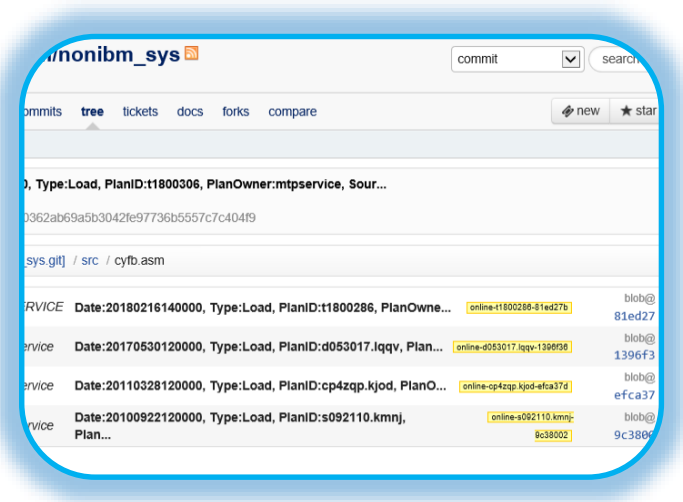
Technical Service Desk(TSD) Role (hyperlinks in images)



Zoldr actions – Staging loadsets deployed in zTPF Production system

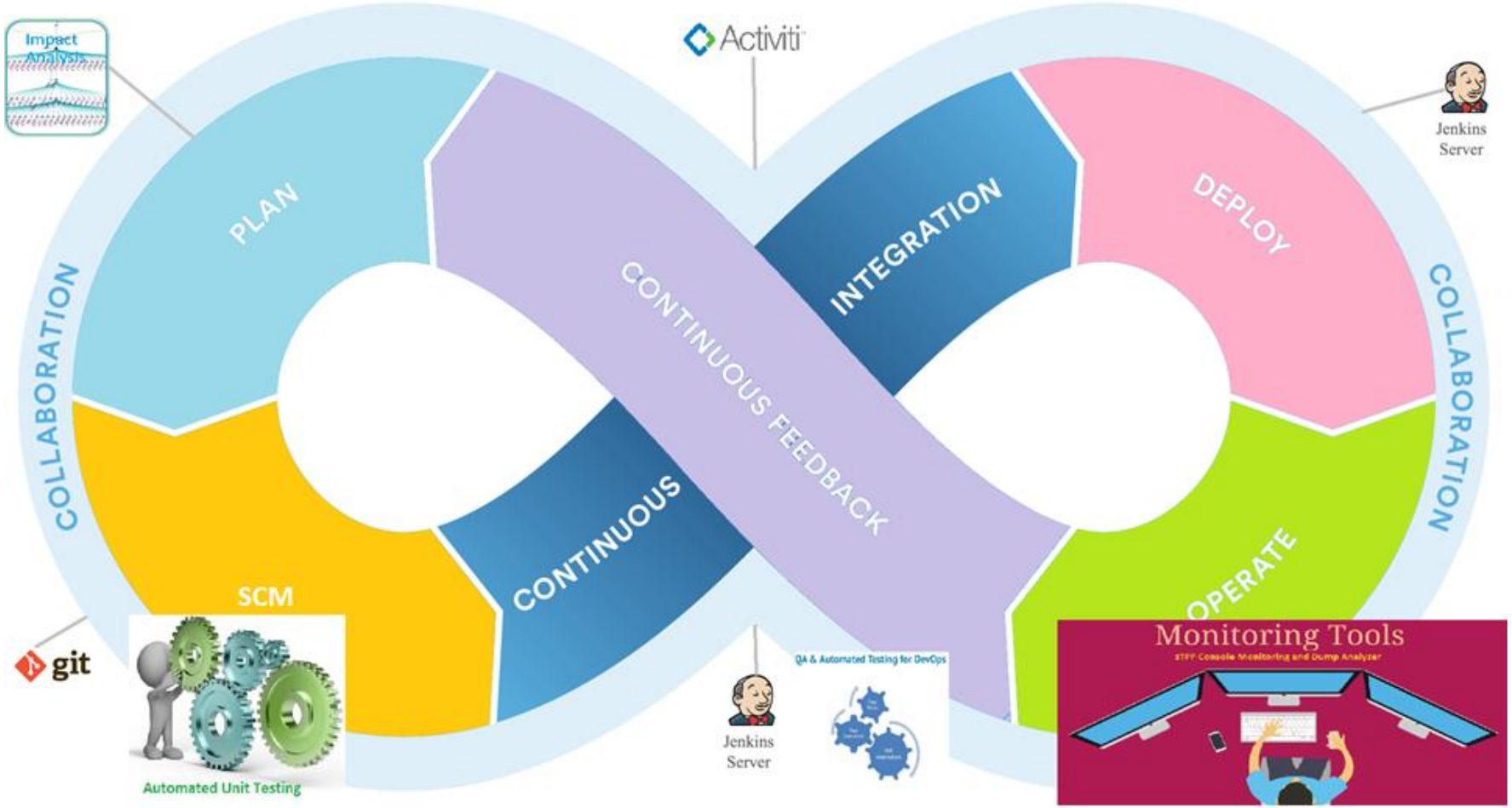


zTPF Online feedback – Project marked as Online.



Artifacts merged to Production Git Repository

zTPF DevOps Tool Chain – The journey continues…



TPF Software

Travelport

Key Take-Aways and Questions

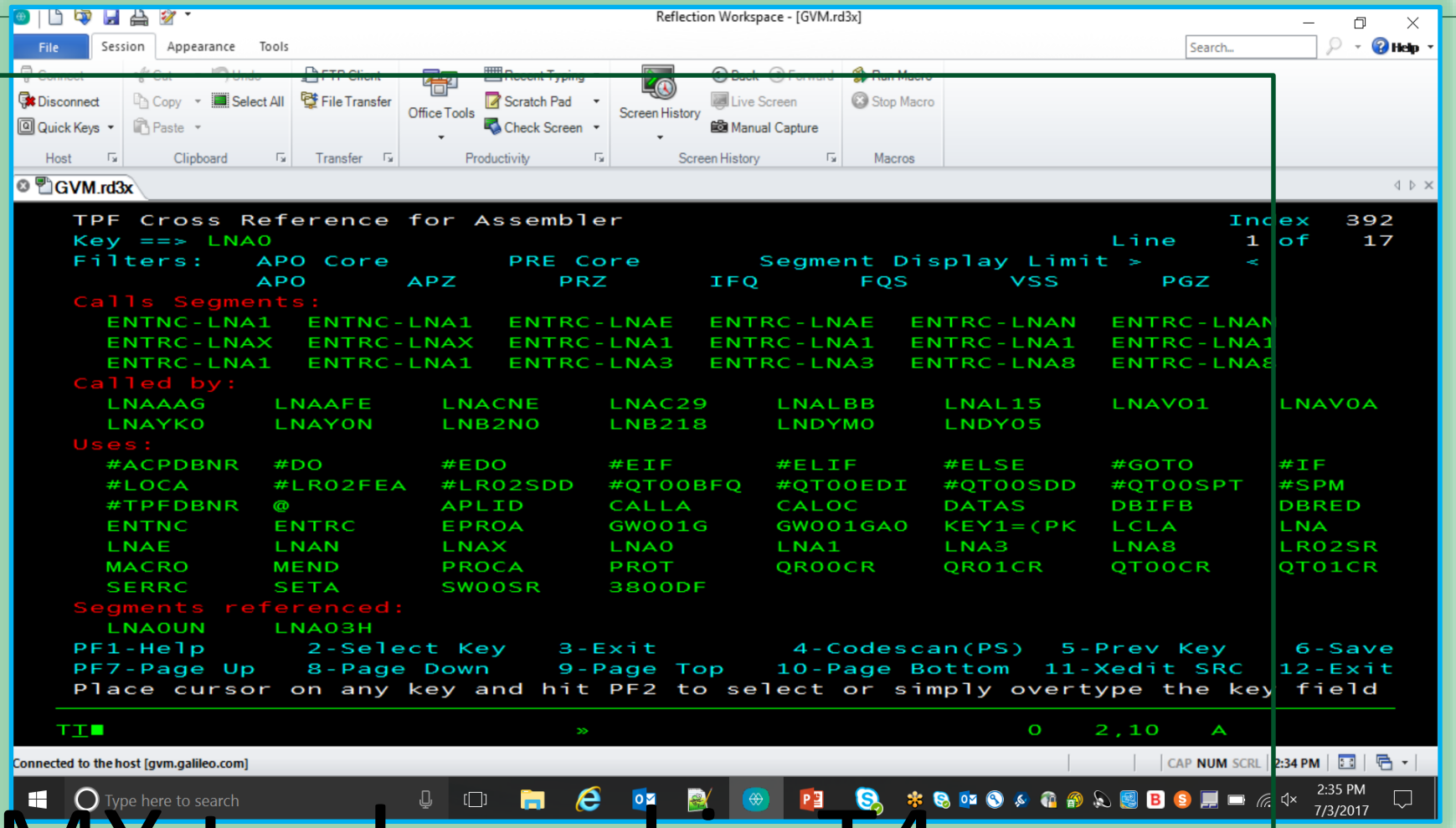
Key Take-Aways

- Modern automated software engineering practices are equally applicable to TPF and mainframe in general.
  - Consistent and connected software engineering across the organization
  - Productivity improvements in TPF development through implementing common processes and modern tools.
  - Learning curve for new hires shortened by leveraging Open Source Tools that are taught in colleges today.
  - Enforce business processes and development best practices through automation of repetitive tasks in a more consistent manner.
- Engage subject matter experts early on and gain their trust to discussion and demonstration.
- Separate needs from habits.
- Build incrementally and show value.
- Gain support of leadership.

Thanks!

Appendix – Enlarged images

MX tool used in T4 (hyperlinks in images)

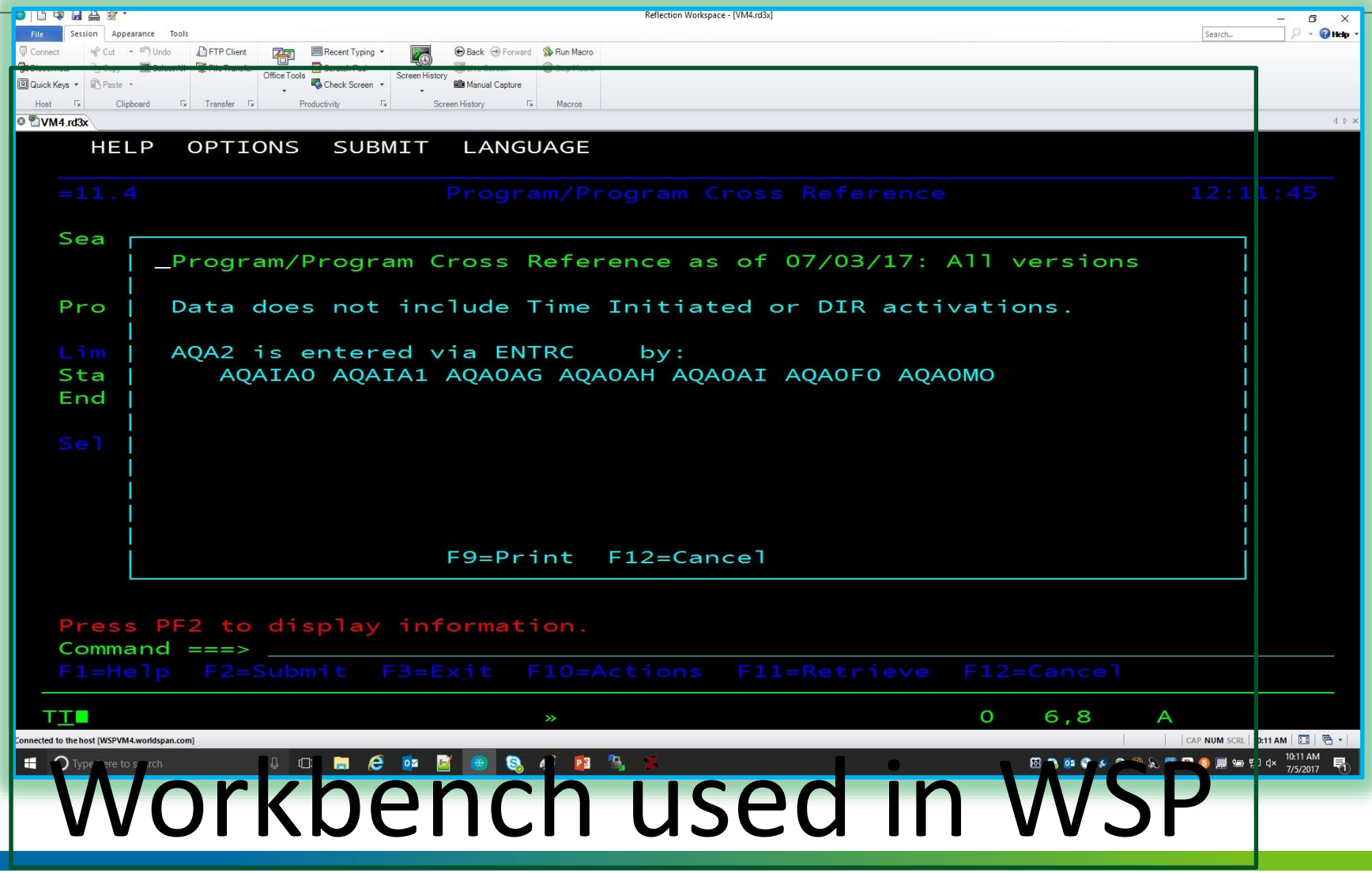


MX tool used in T4

TPF Software

Travelport

Workbench used in WSP (hyperlinks in images)

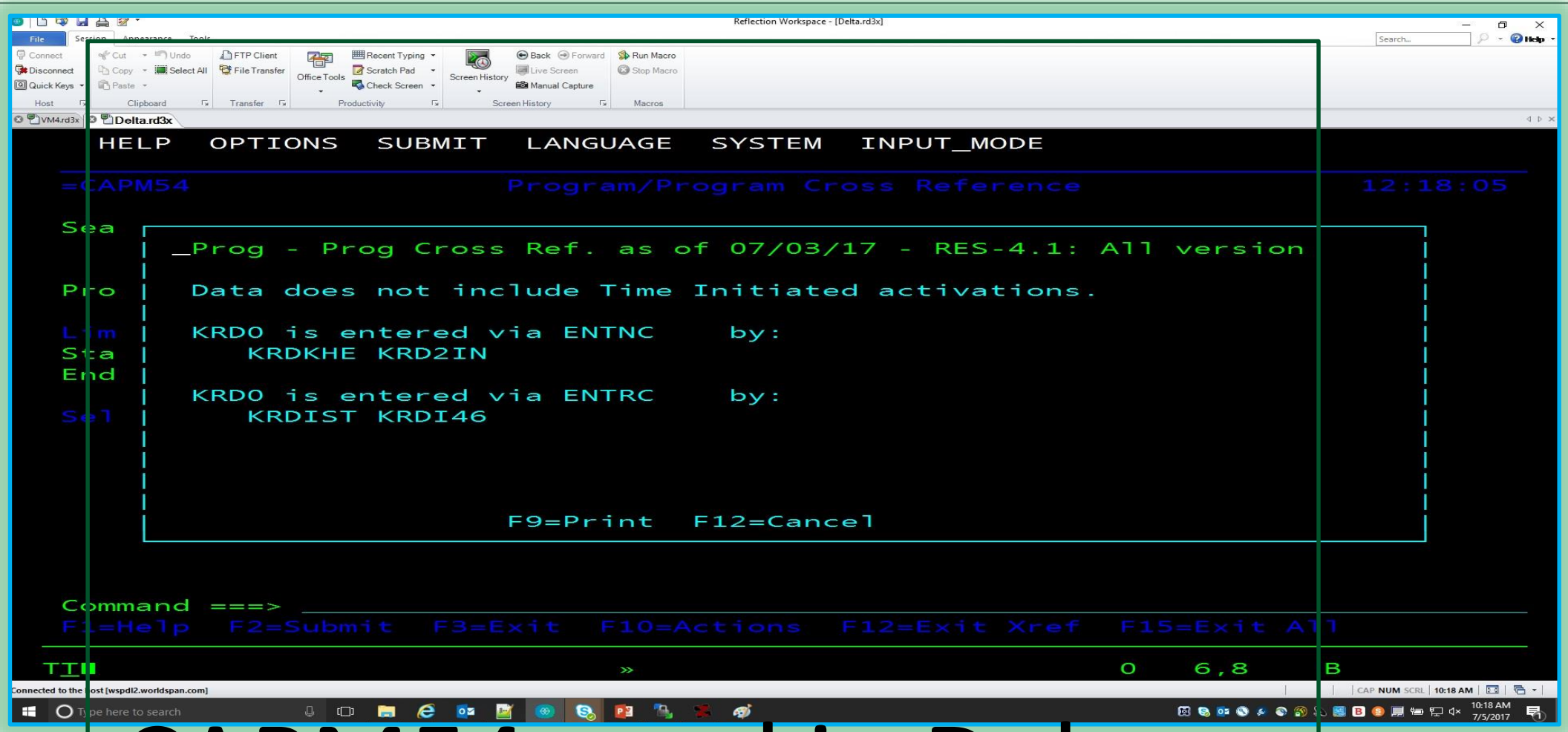


Workbench used in WSP

TPF Software

Travelport

CAPM54 used in Delta (hyperlinks in images)



CAPM54 used in Delta

TPF Software

Travelport

Displays Source (hyperlinks in images)

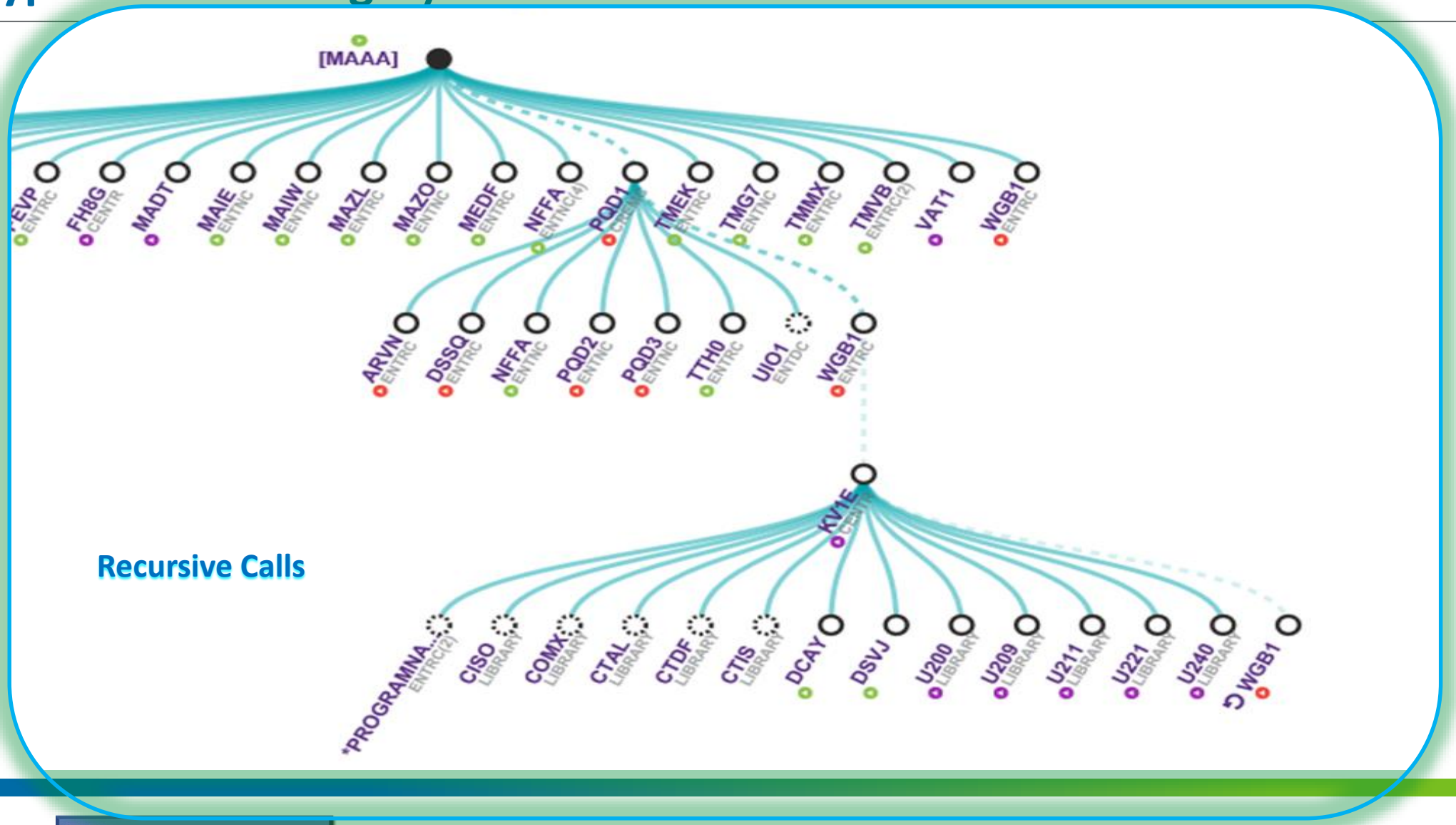
```
         LR    R6,R5                COPY INTO R6 ALSO
         BCTR  R6,0                 SUBTRACT ONE FOR INDEX
         MPY   R6,L'ISTENT              AND INDEX INTO I-STREAM TABLE
         AR    R14,R6               ADD THE BASE OF THE I-STREAM TABLE
         TM    ISTACTV,L'ISTACTV    THIS I-STREAM ACTIVE?
         BNO   SWIS020              -NO, CHECK 1ST PREVIOUS
         TM    ISTUSE,L'ISTUSE      THIS I-STREAM USABLE?
         BO    SWIS030              -YES, GO SWISC
SWIS020  EQU   *
         LR    R14,R15              REESTABLISH BASE OF I-STREAM TABLE
         BCT   R5,SWIS010
SWIS030  EQU   *
         LH    R6,CE1ISN            DETERMINE WHICH I-STREAM IS PROC
         CR    R6,R5                IS THIS ALREADY THE LAST I-STREAM?
         BE    SWIS040              YES- DON'T EXECUTE MACRO
         SWISC KRD1,TYPE=ENTER,IS=R5,LEV=D0,D1
*****************************************************************
```

Displays Source

TPF Software

Travelport

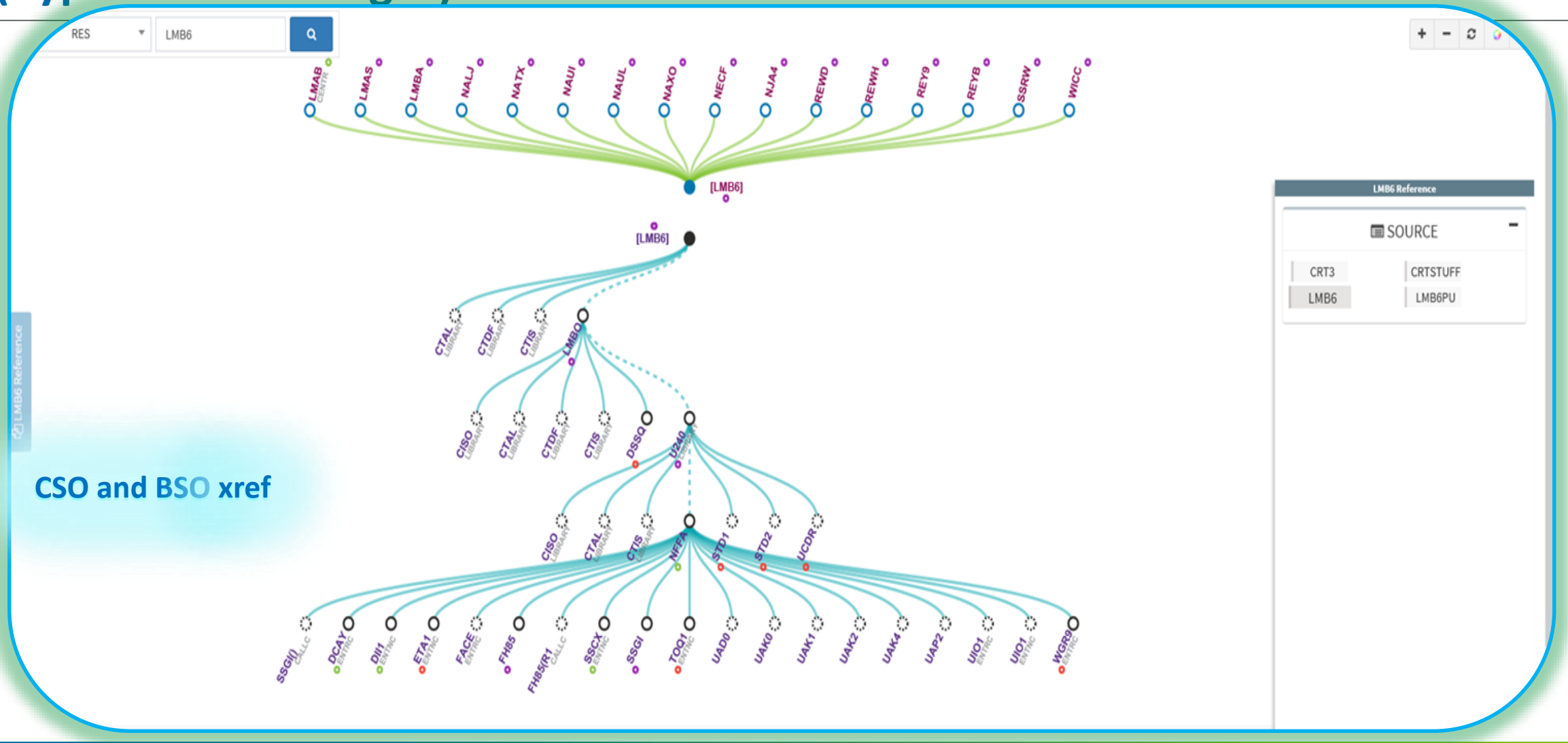
Recursive Calls
(hyperlinks in images)



TPF Software

Travelport

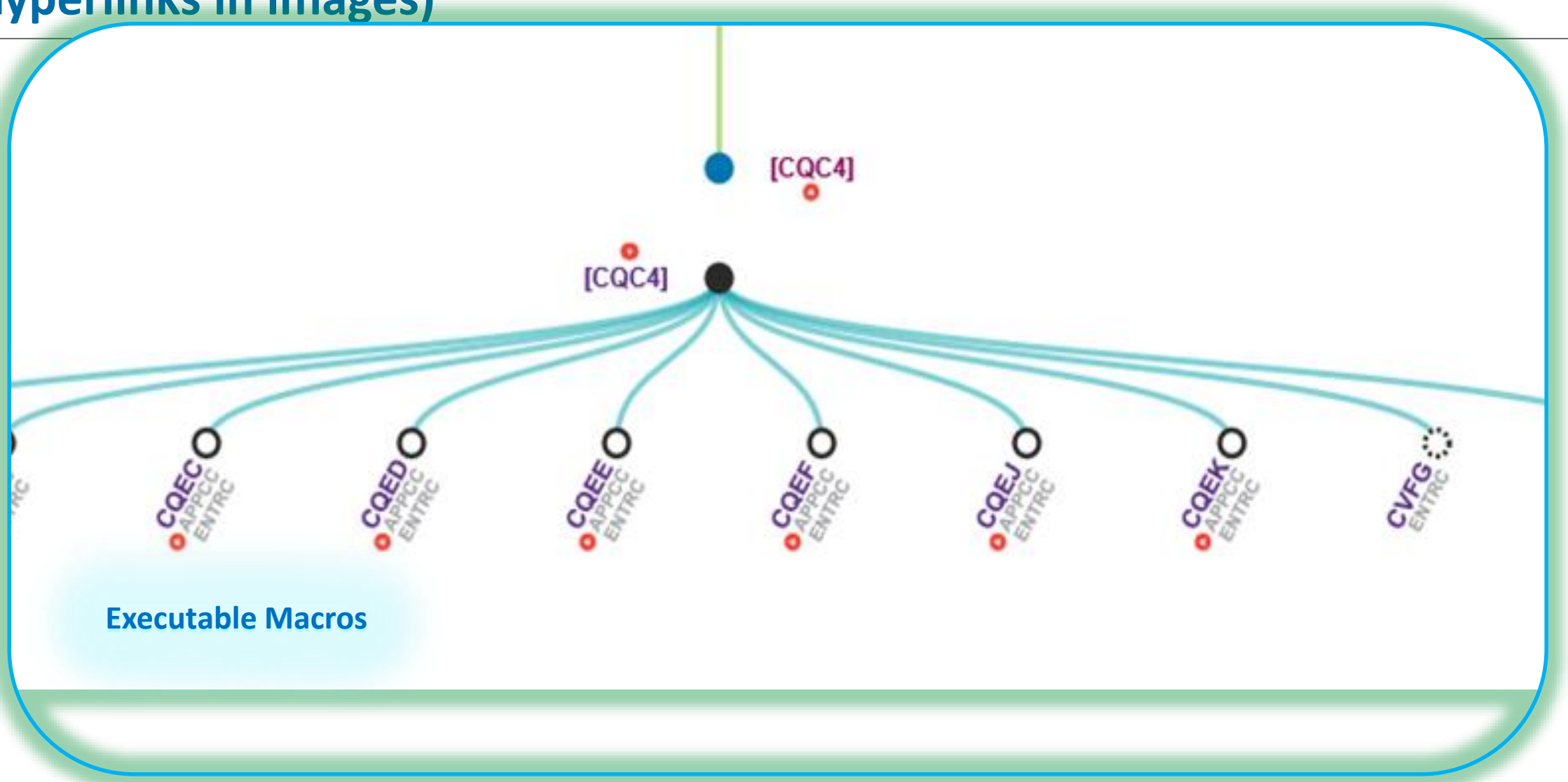
CSO and BSO xref (hyperlinks in images)



TPF Software

Travelport

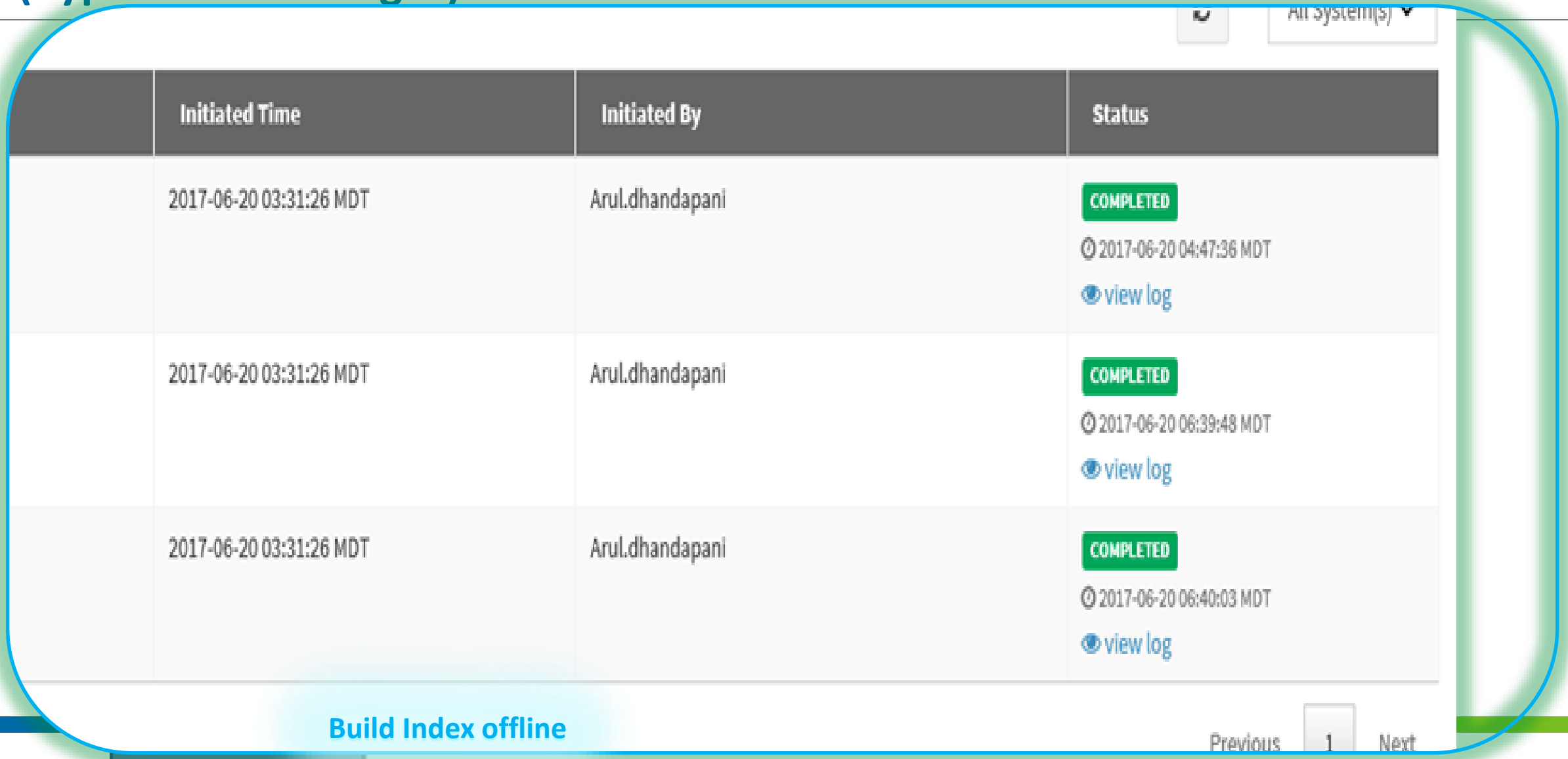
Executable Macros (hyperlinks in images)



TPF Software

Travelport

Build Index Offline
(hyperlinks in images)

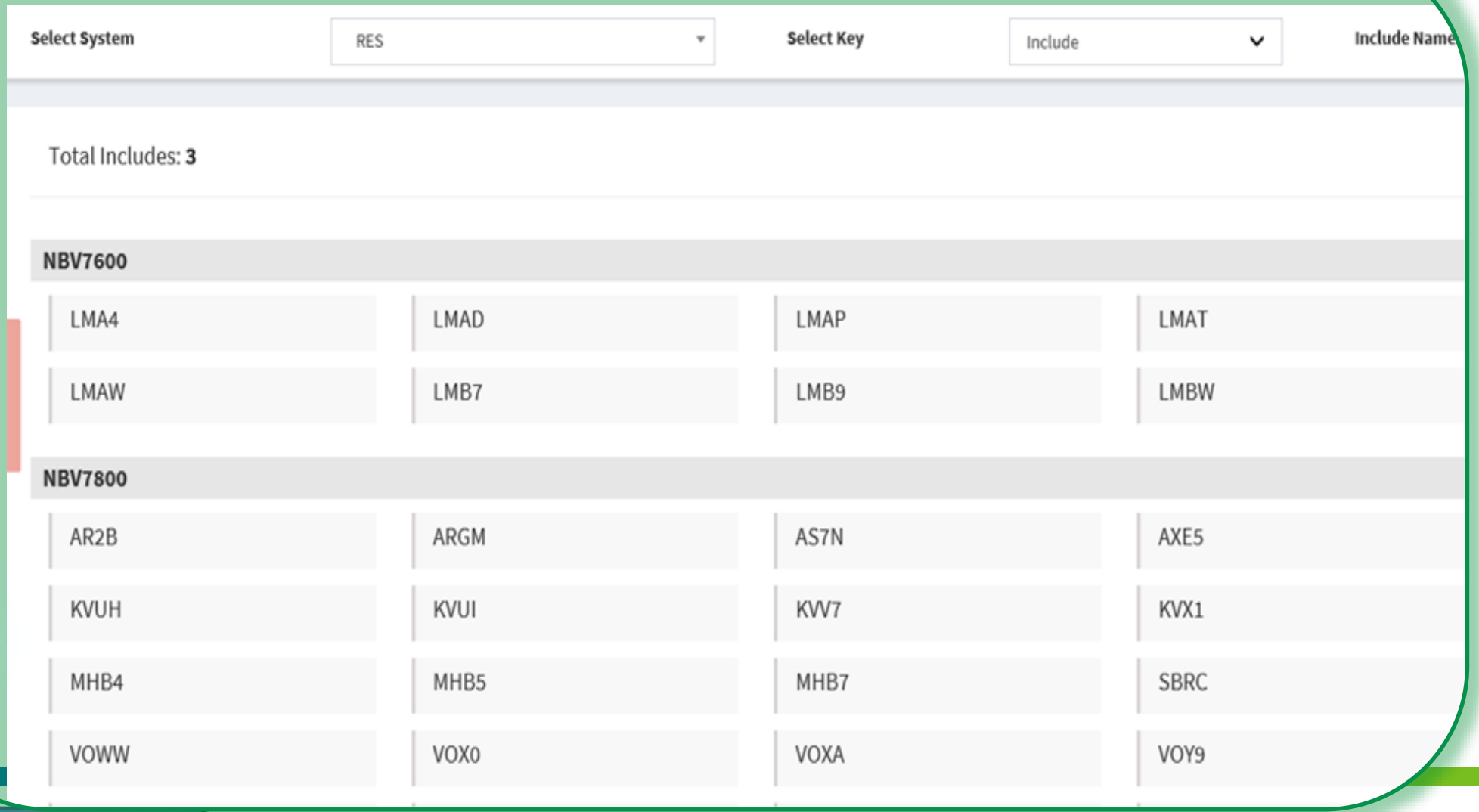


Build Index offline

TPF Software

Travelport

Sabretalk xref (hyperlinks in images)

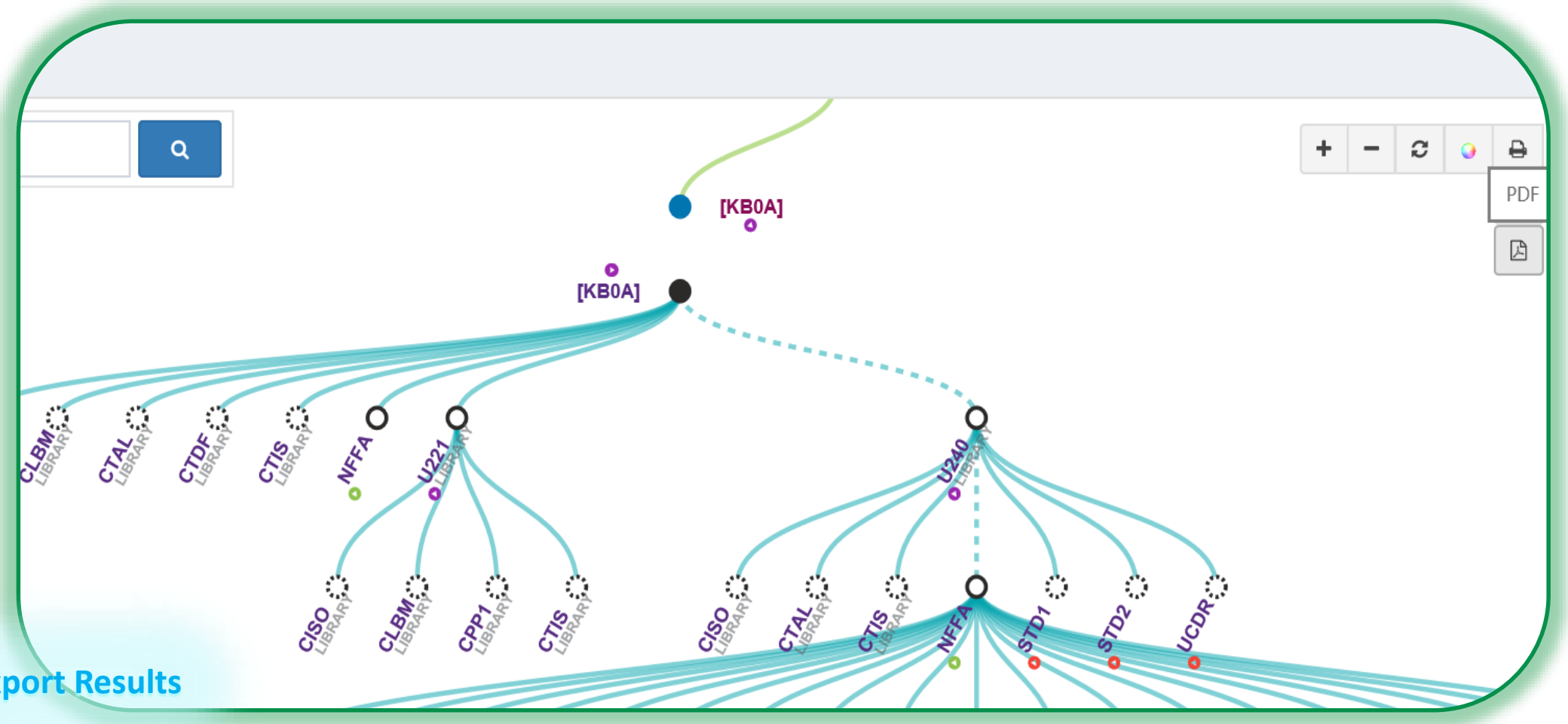


Sabretalk

TPF Software

Travelport

Export results – pdf, xls (hyperlinks in images)

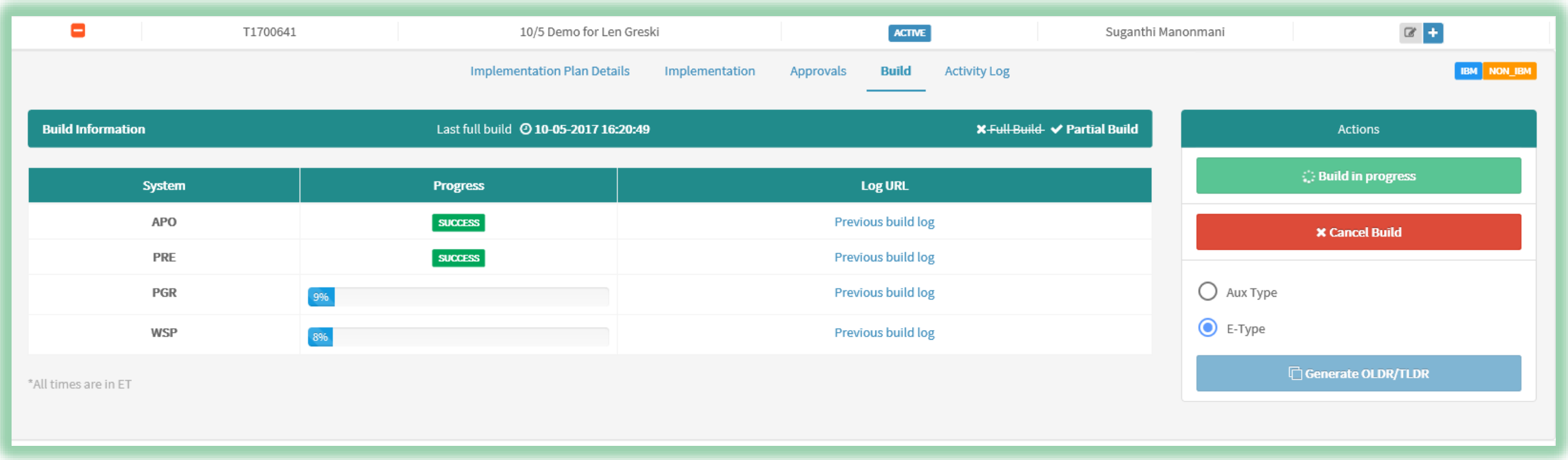


Export Results

TPF Software

Travelport

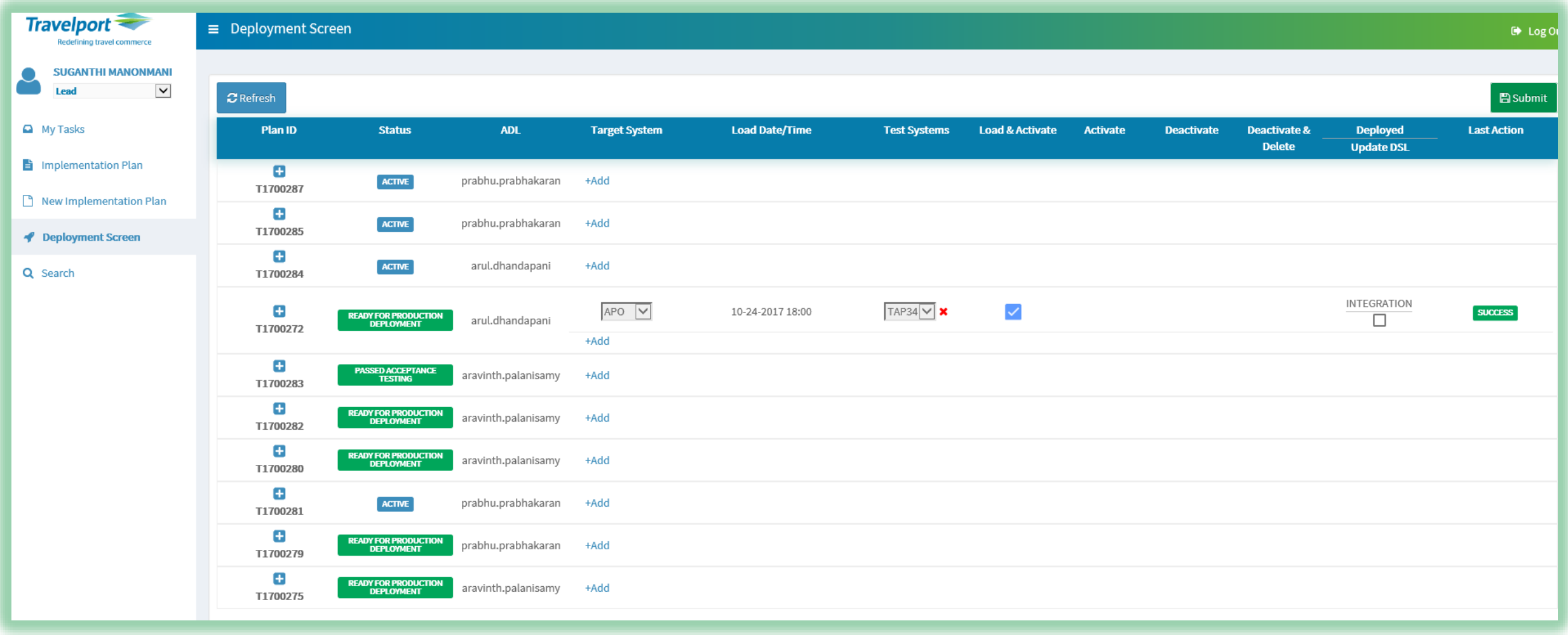
Build in devl workspace



TPF Software

Travelport

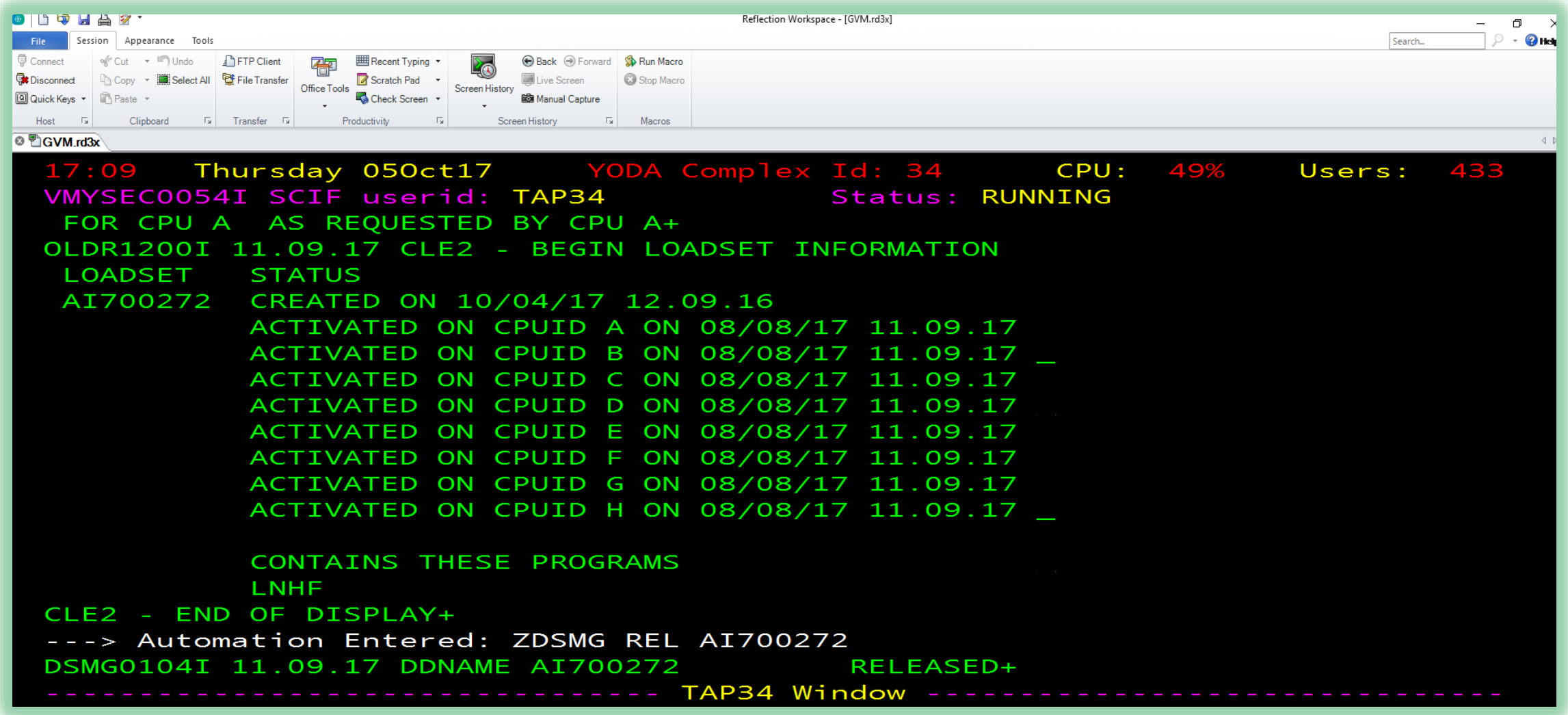
Deployment Screen for zTPF Test Systems



TPF Software

Travelport

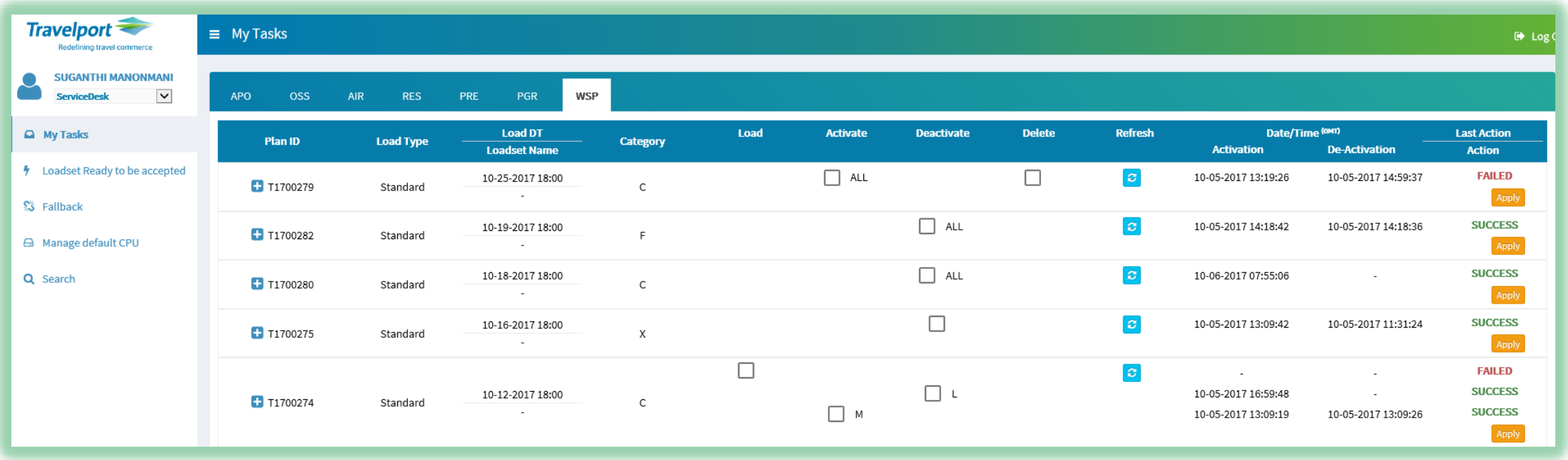
zTPF VPARS test systems



TPF Software

Travelport

TSD Screen for zTPF Production Deployment

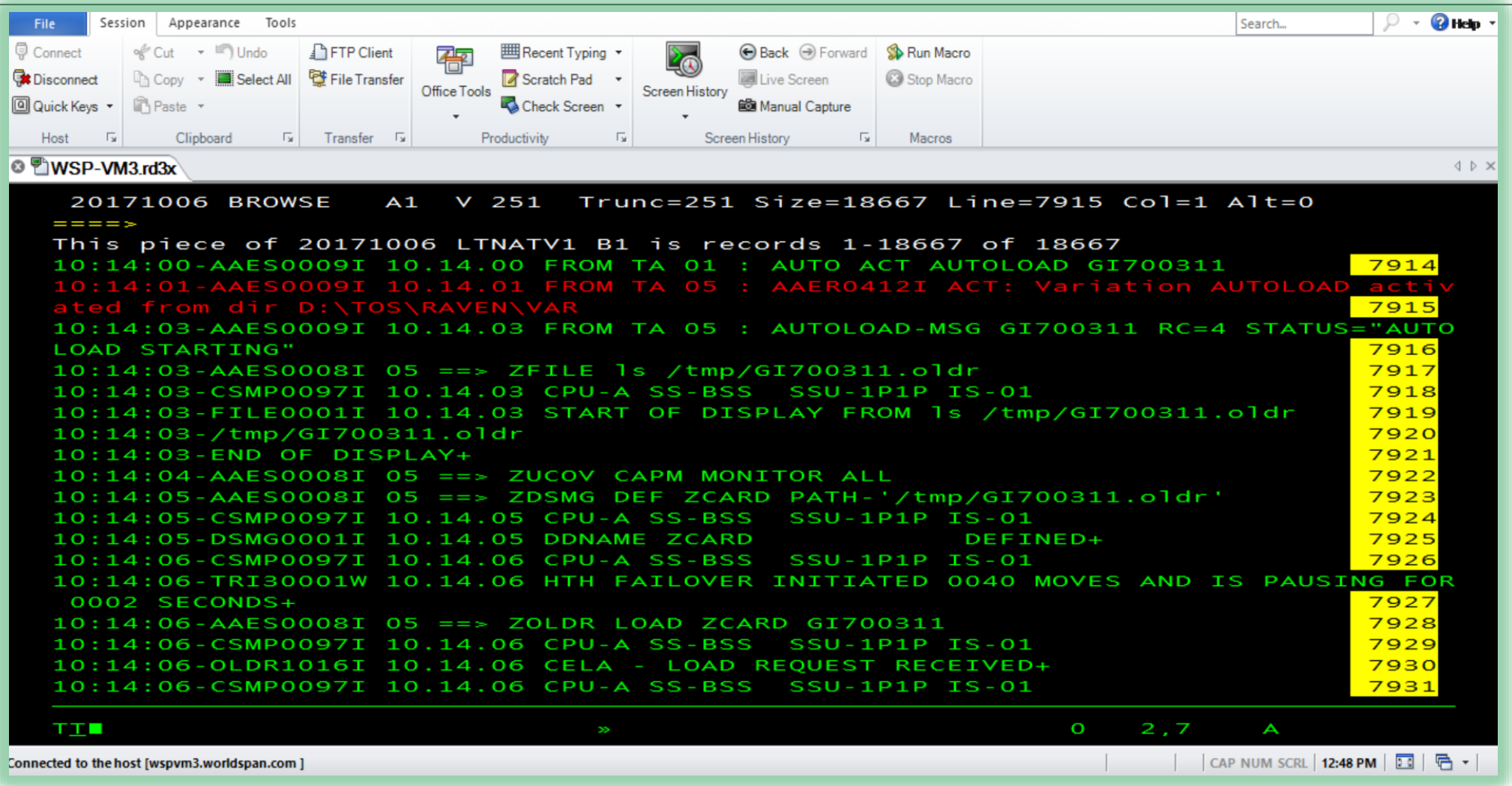


Plan ID	Load Type	Load DT / Loadset Name	Category	Load	Activate	Deactivate	Delete	Refresh	Date/Time (GMT) Activation	Date/Time (GMT) De-Activation	Last Action
T1700279	Standard	10-25-2017 18:00 -	C		ALL				10-05-2017 13:19:26	10-05-2017 14:59:37	FAILED Apply
T1700282	Standard	10-19-2017 18:00 -	F			ALL			10-05-2017 14:18:42	10-05-2017 14:18:36	SUCCESS Apply
T1700280	Standard	10-18-2017 18:00 -	C			ALL			10-06-2017 07:55:06	-	SUCCESS Apply
T1700275	Standard	10-16-2017 18:00 -	X						10-05-2017 13:09:42	10-05-2017 11:31:24	SUCCESS Apply
T1700274	Standard	10-12-2017 18:00 -	C		M	L			- 10-05-2017 16:59:48 10-05-2017 13:09:19	- - 10-05-2017 13:09:26	FAILED SUCCESS SUCCESS Apply

TPF Software

Travelport

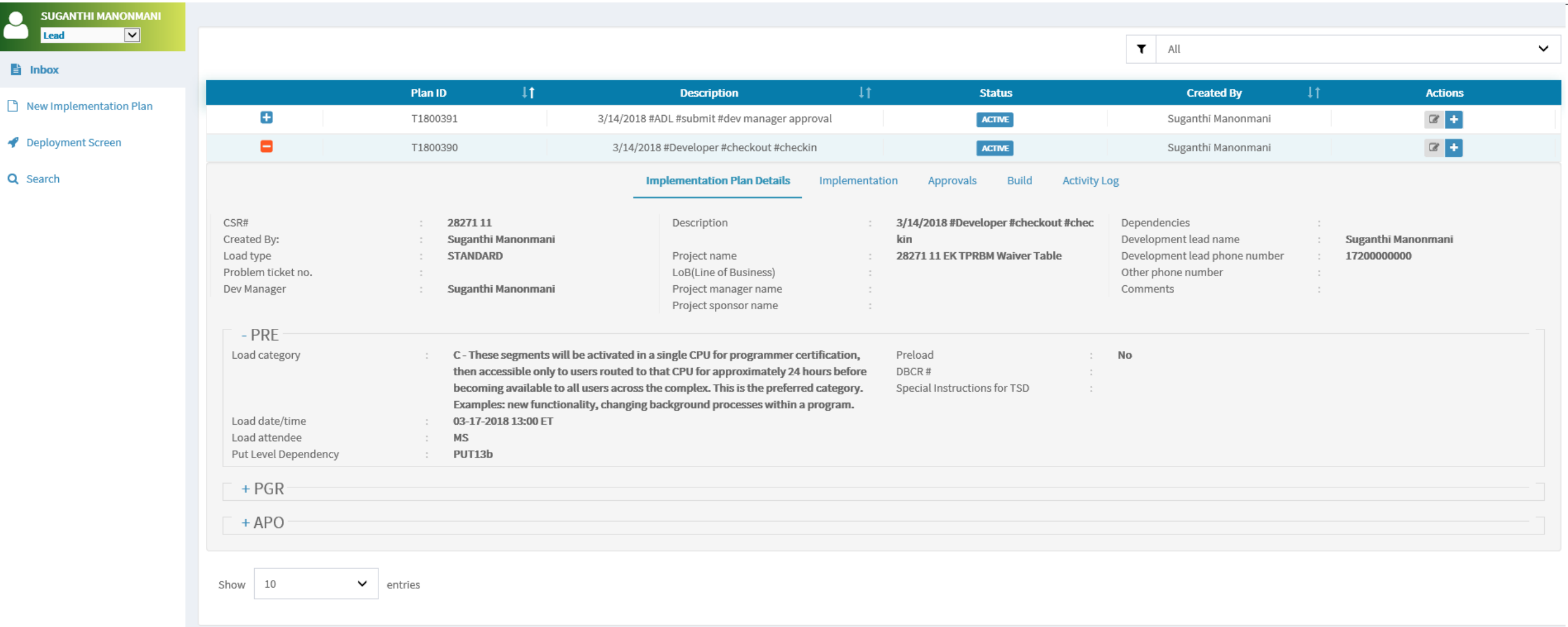
zTPF Production /Native Systems



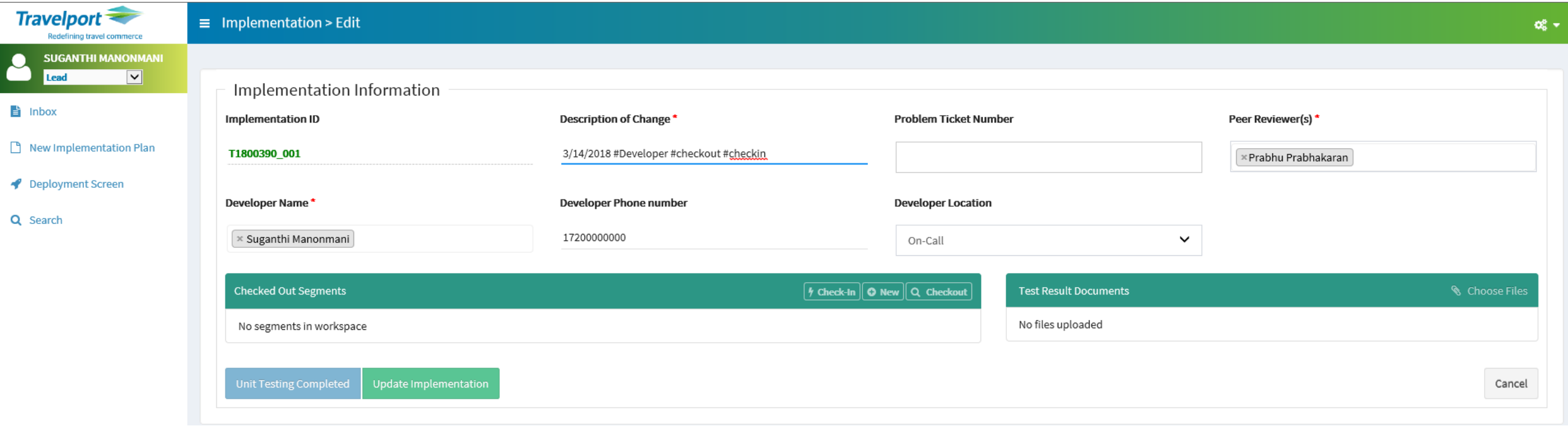
TPF Software

Travelport

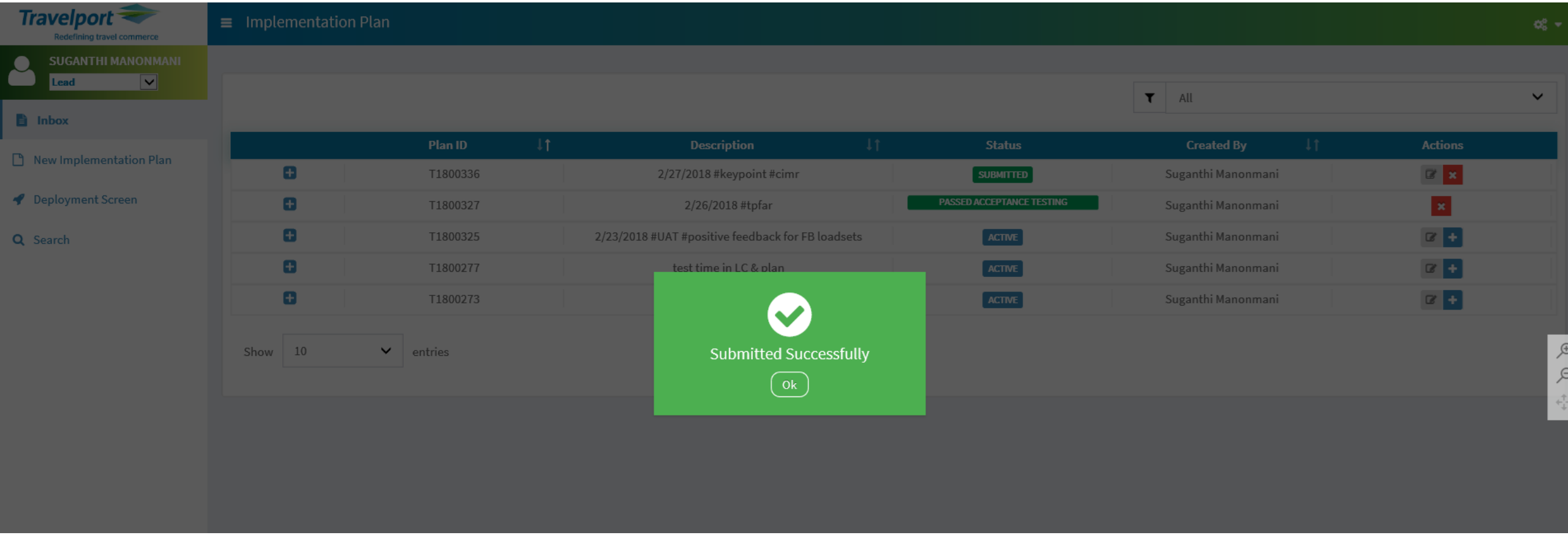
ADL – Creates Project



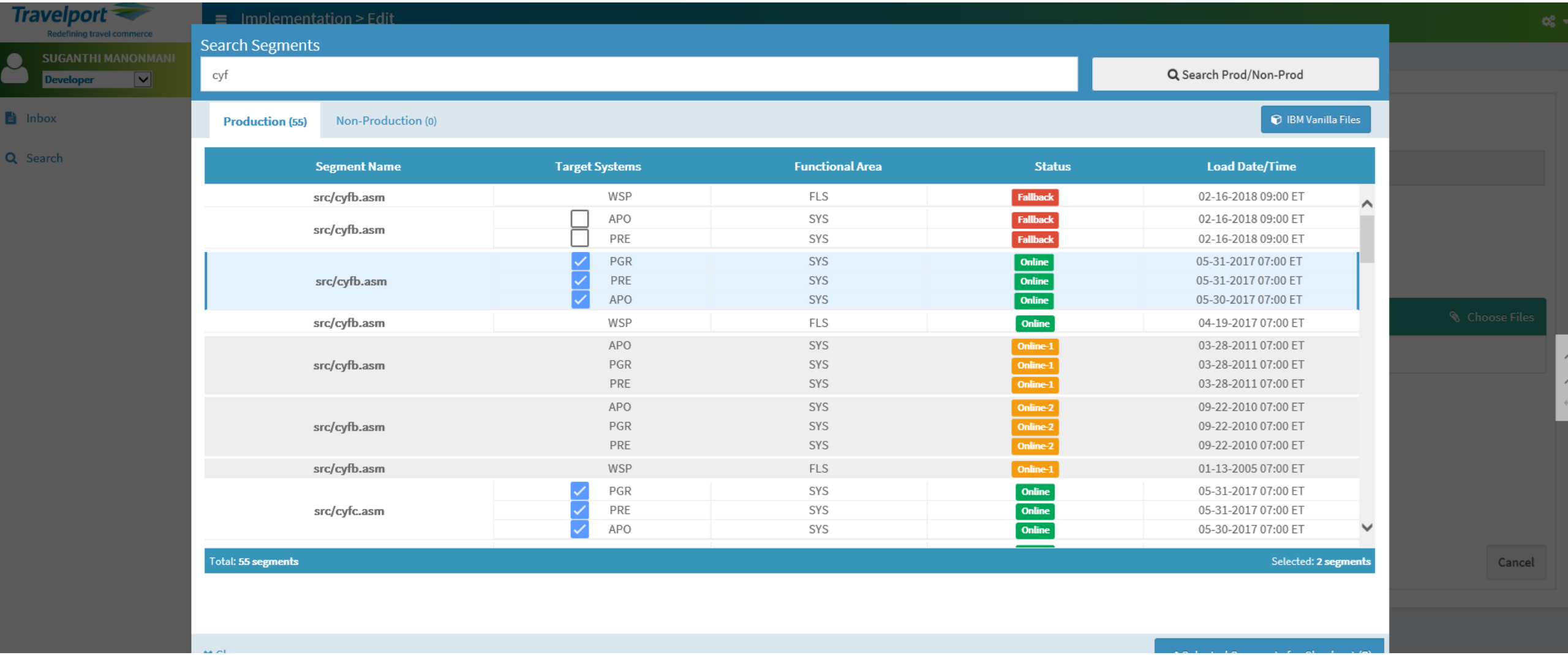
ADL – Creates Feature



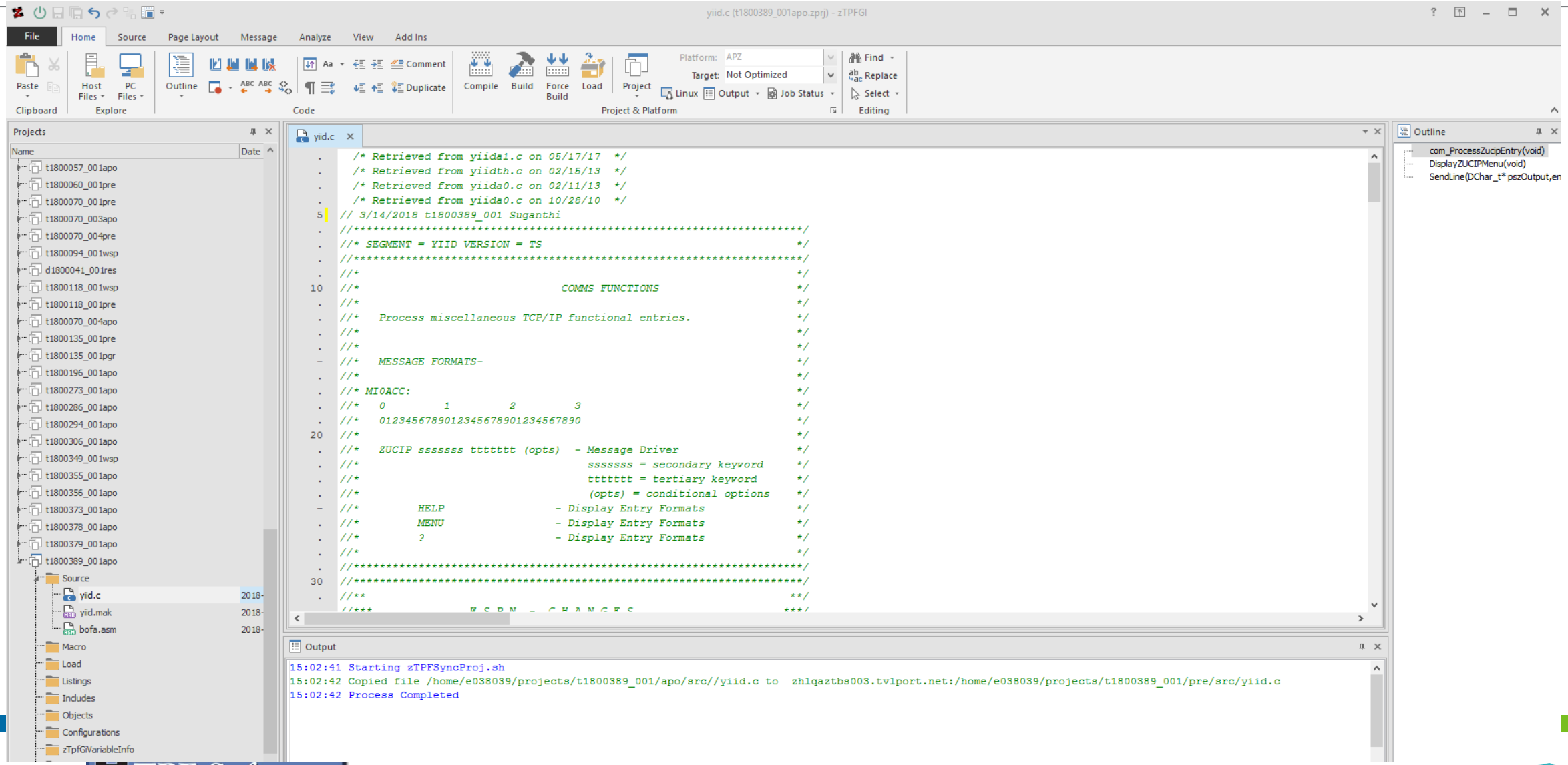
ADL – Submit Project



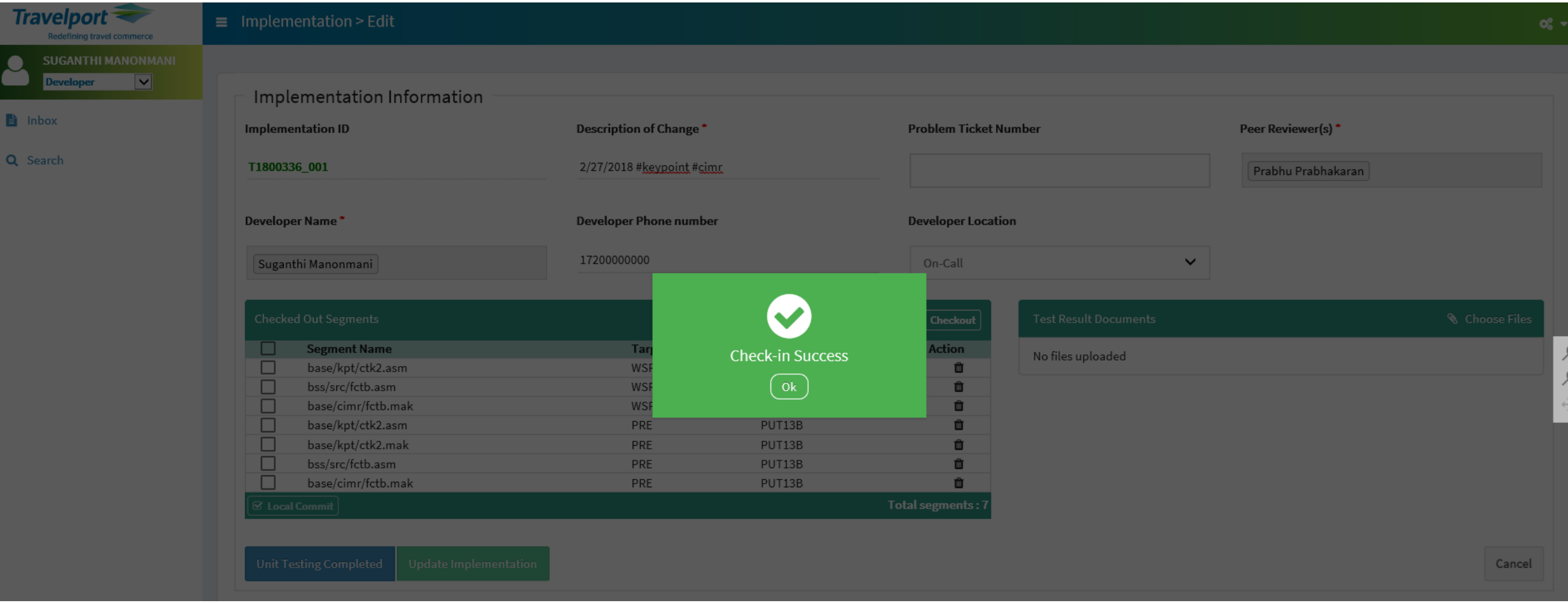
Developer – Source code search for Check-out



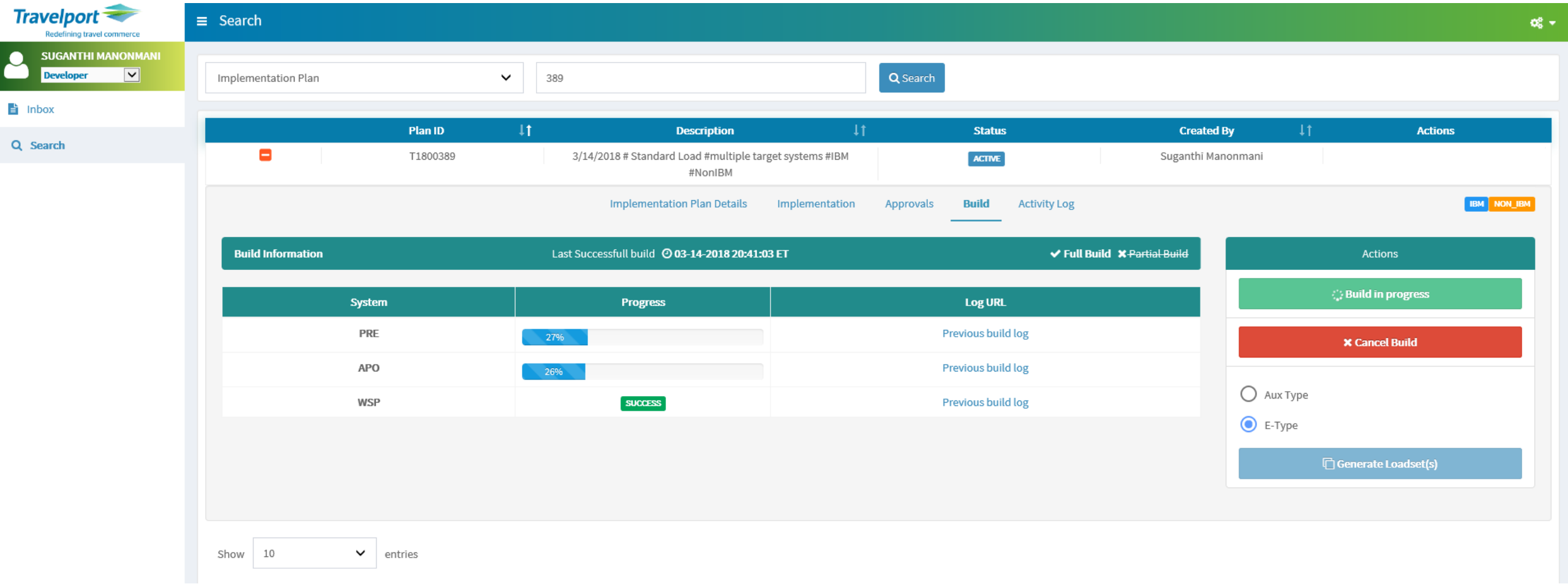
Developer – Edit, File Sync



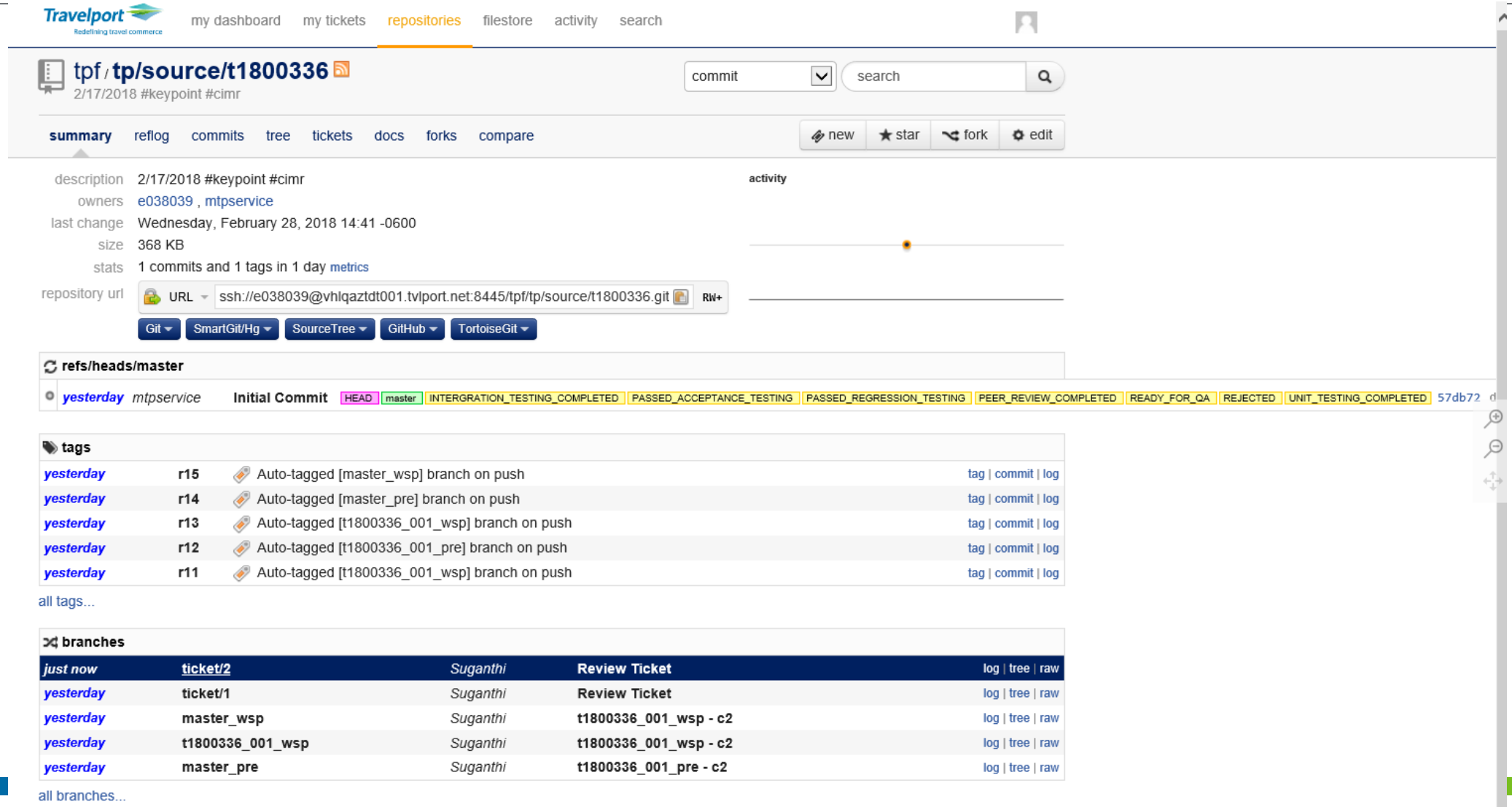
Developer – Check-in



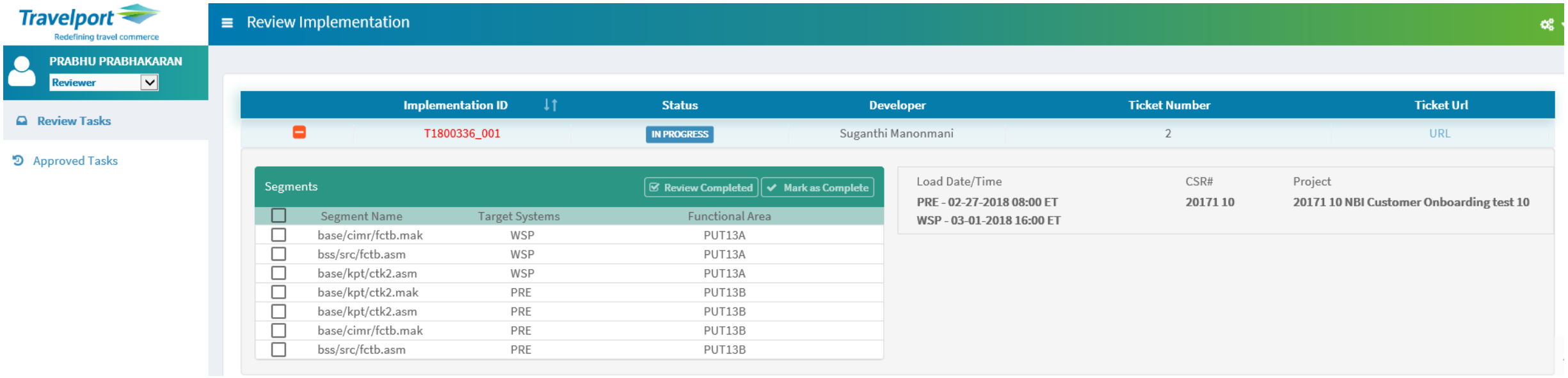
Developer – Devl Build



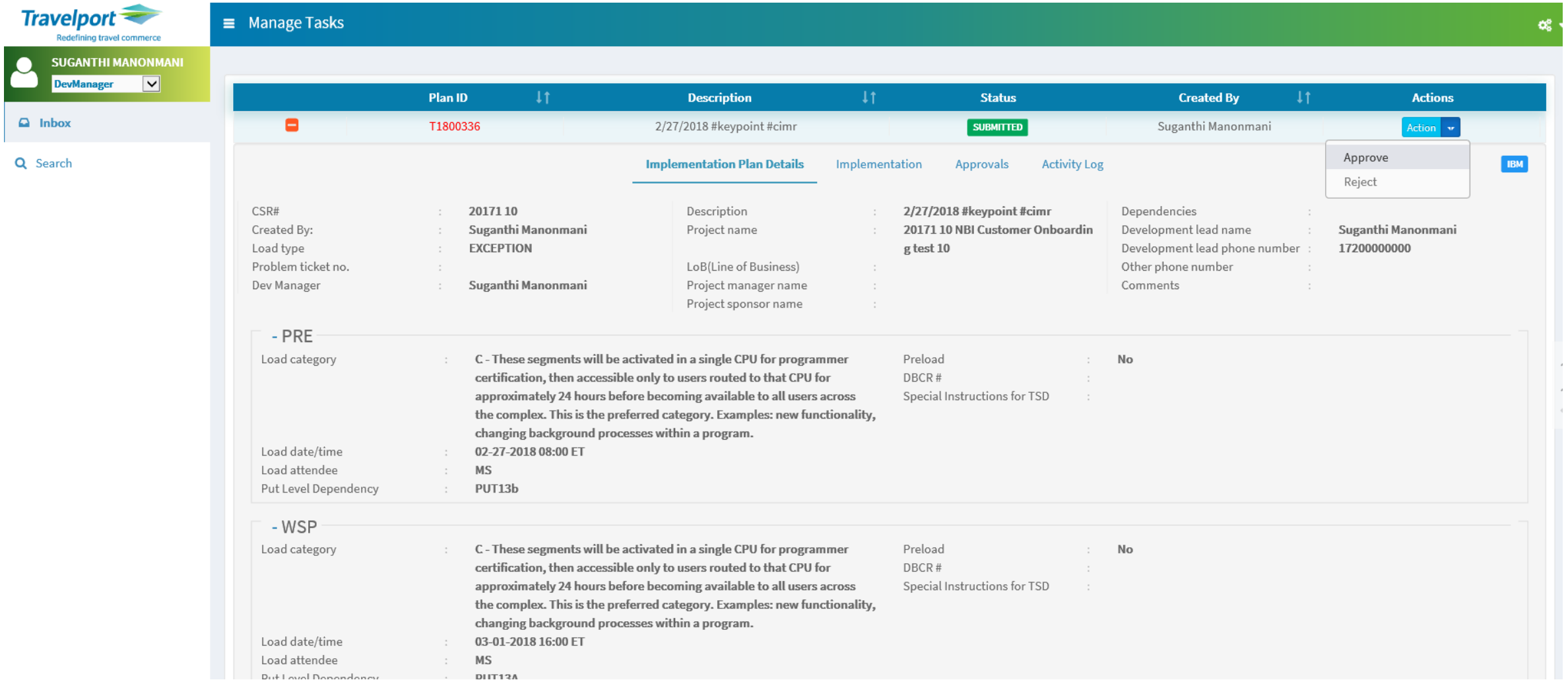
Reviewer – peer review comments



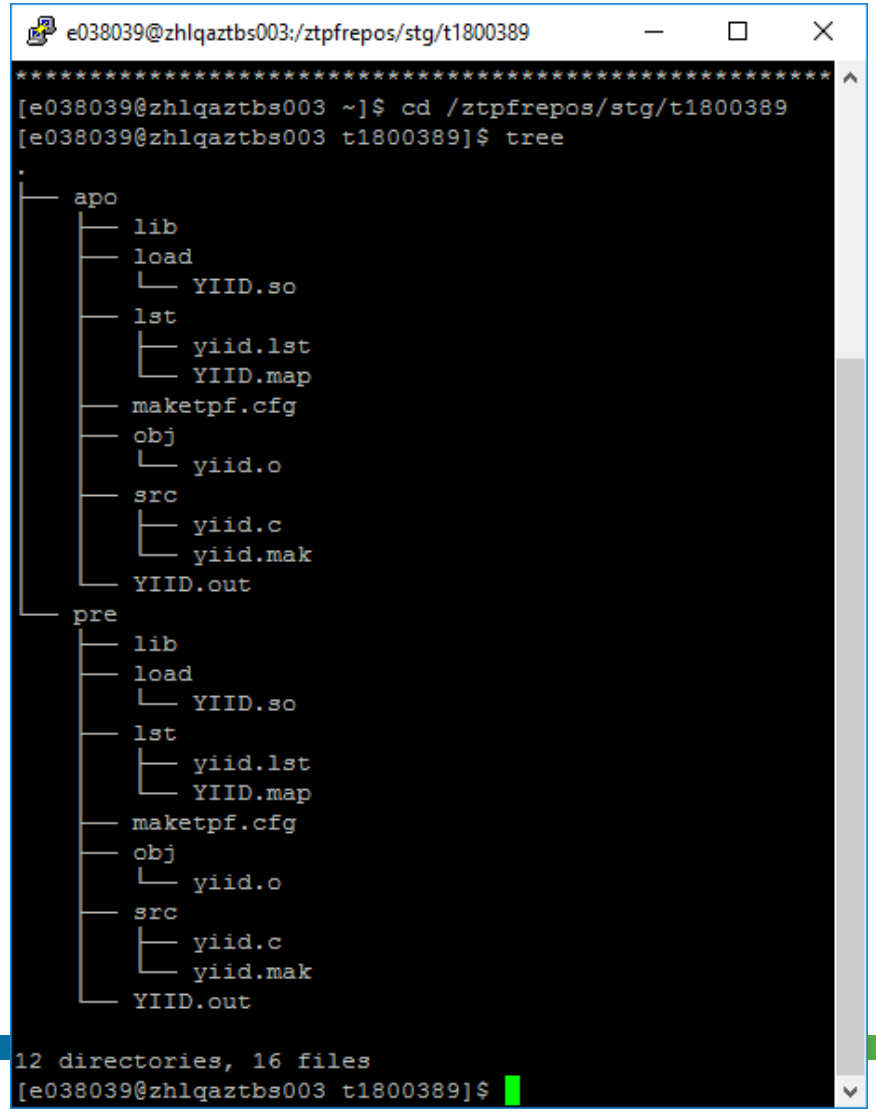
Reviewer – mark peer review completed



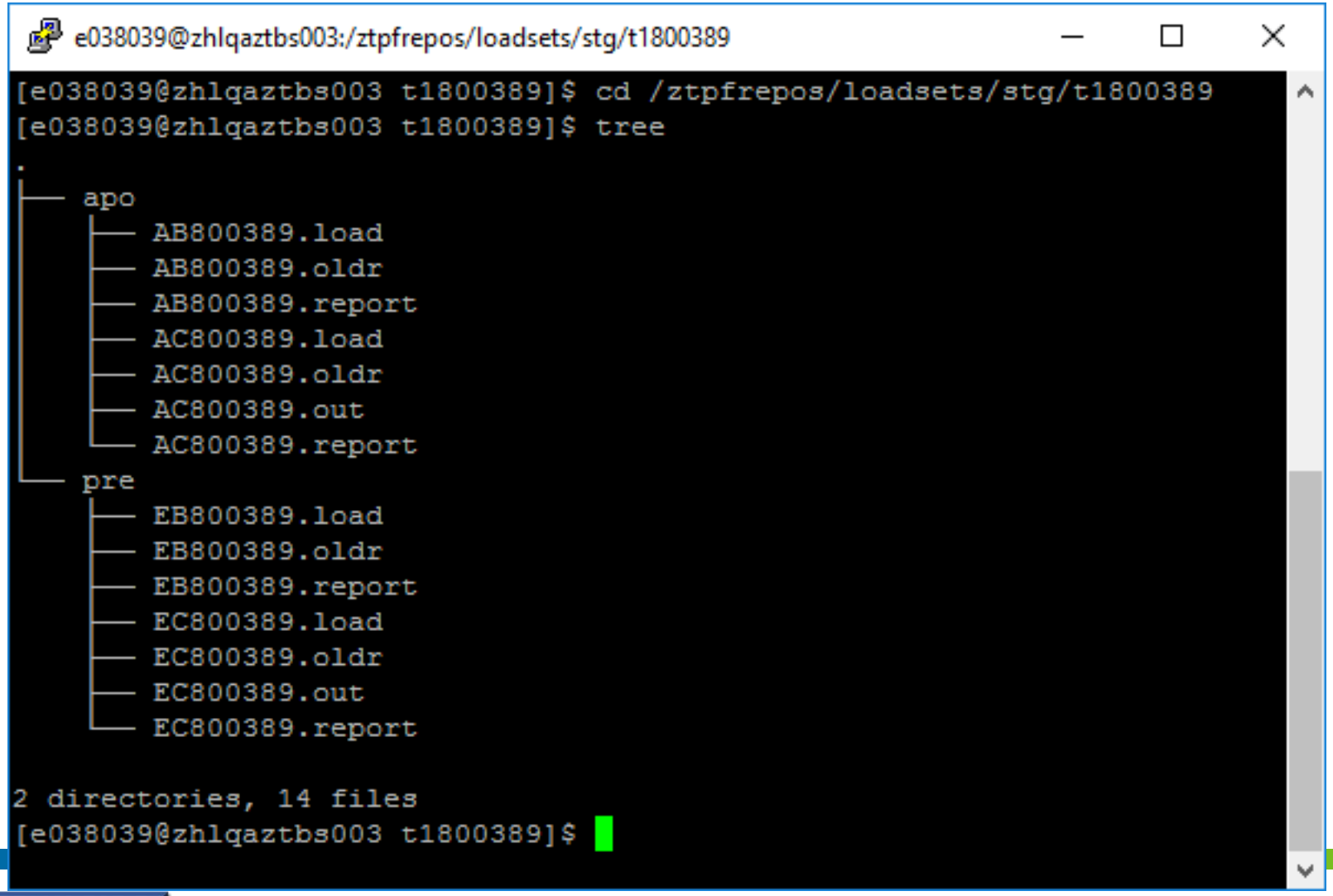
Dev Manager – Approve Project



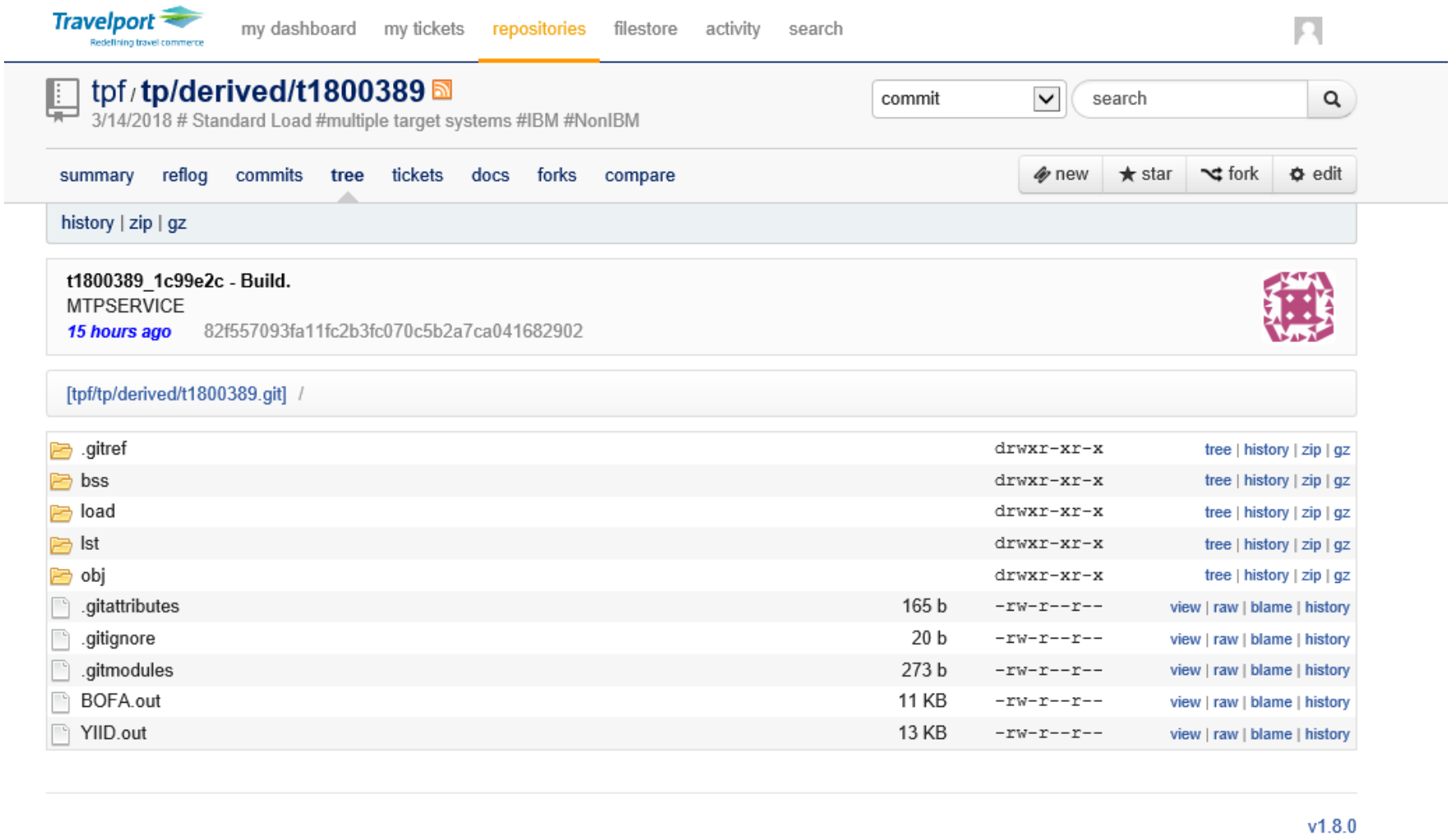
Dev Manager – Staging build



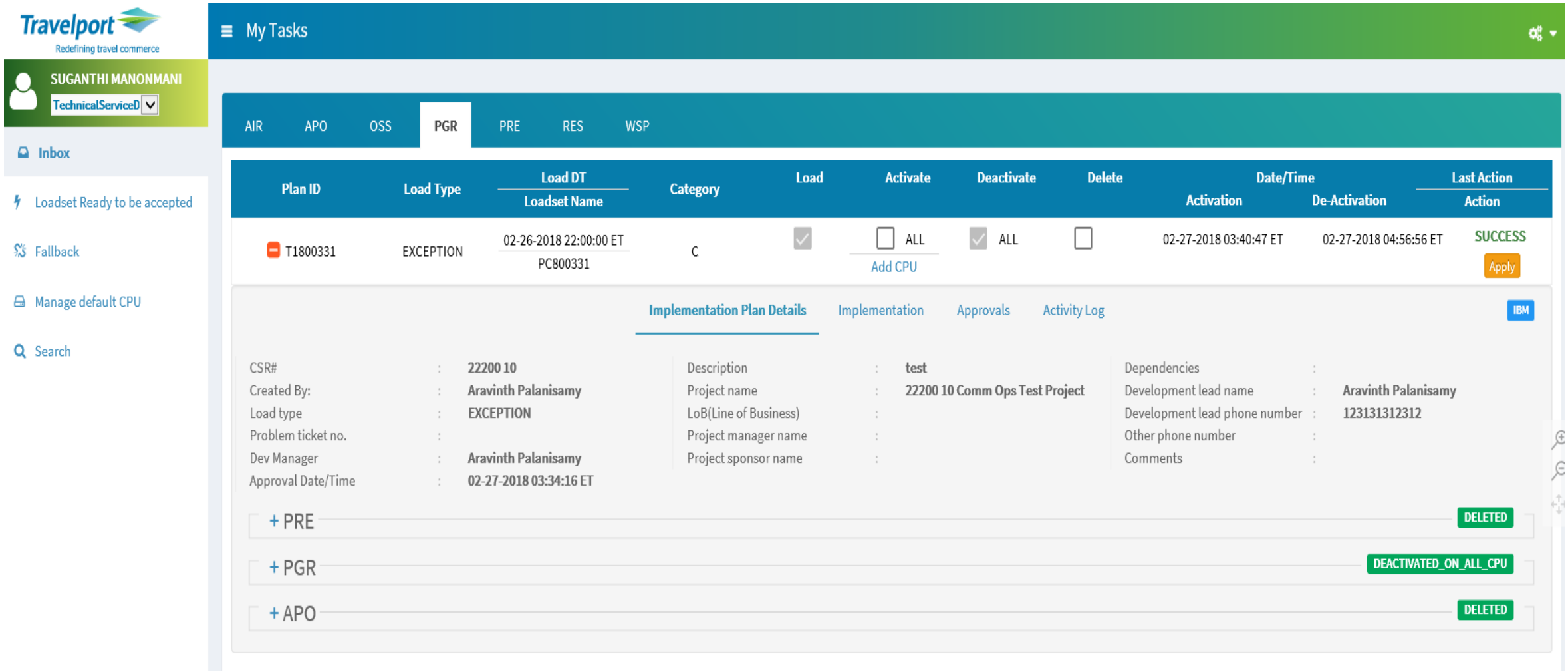
Dev Manager – Staging loadset & fallback loadset



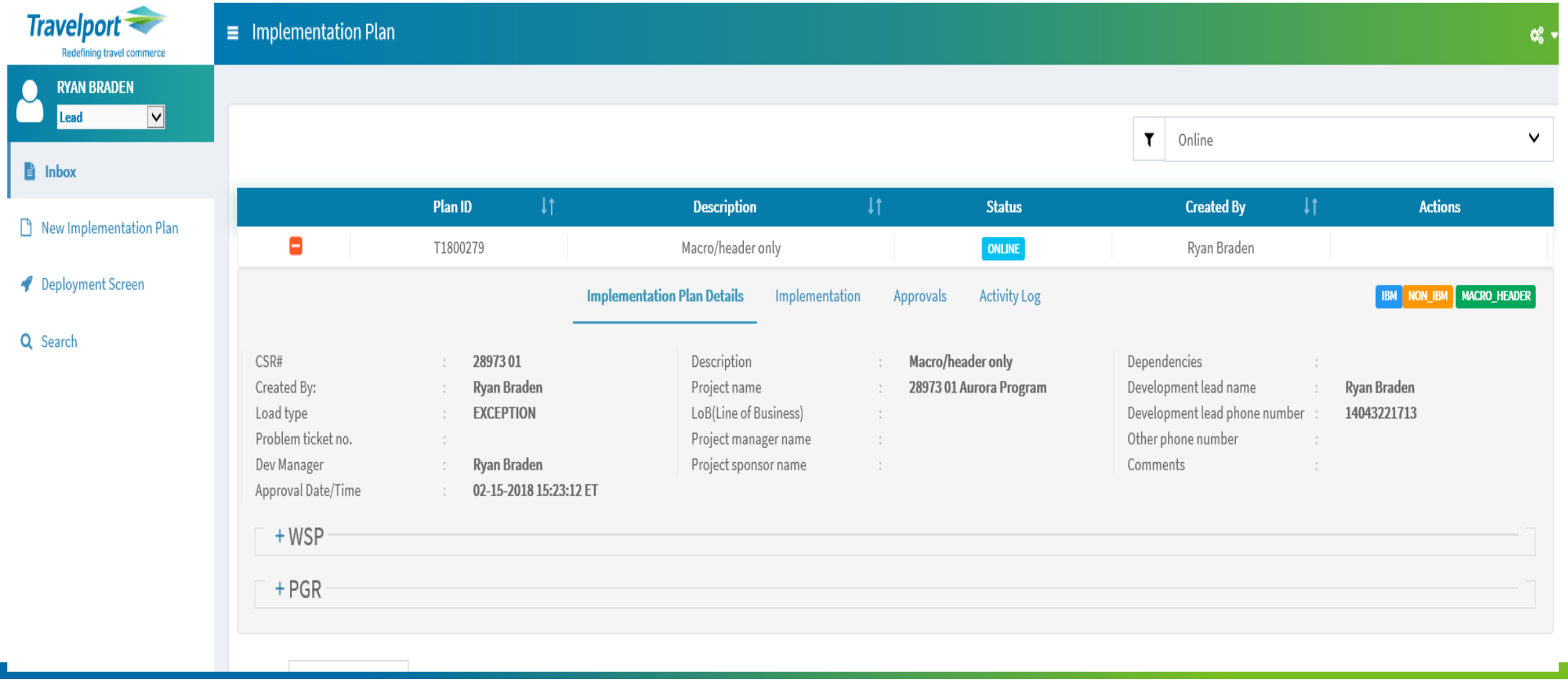
Dev Manager – Staging derived artifacts pushed to derived repository



Technical Service Desk – zoldr actions



Technical Service Desk – zTPF Online feedback



Technical Service Desk – Merged to Production Git Repository