Session #1851

Reinventing Allstate’s Mainframe: Our story so far...

Anthony Robinson
Billing Manager, Allstate Insurance

March 24, 2018 – IBM Think Conference
History of Mainframe Application Development

- **Traditional ‘green screen’ COBOL/PL1, CICS, Assembler, Db2 and IMS**
- **Utilized Toolbox (built on top of CA Endeavor) and SCLM for source code management**
- **Homegrown deployment process – Allstate Program Transmittal Facility ‘APTF’**
- **Software installed by production control team**
- **Manual coordination with dependent distributed-based application changes**
Driving factors for mainframe modernization

- **Speed to market – Agile**
- **Need for quality**
- **Company ‘High Priority Initiatives’ aka HPIs**
  - HPI#6 – ‘select common tools, platforms and processes (developer frameworks)’
  - HPI#7 – ‘transition to a Product mindset, including identifying and implementing the necessary roles, skills and behaviors, aligning support organizations and processes to enable this transition’
- **Cost**
- **Continuous integration and delivery**
- **Streamlined production deployments**
- **Billing is a critical business application @ Allstate**
- **Aging workforce**
- **Global developer community**
Driving factors for mainframe modernization

*DevOps efficiencies ...*

*Through modern tools and development environments, development efficiency is improved, aspects of the lifecycle are automated, steps within the lifecycle shrink and application quality is upgraded*
SCLM going off support in April 2017
Allstate had an existing Rational infrastructure, including DOORS, ClearCase and Team Concert
Team Concert was already in use at Allstate for Java-based development but company direction was for them to move to git
We evaluated z/OS SCM products but settled on Team Concert
  - Established migration plan from SCLM to RTC
  - IBM RTC project team had previously installed SCLM at Allstate and had performed significant customization work
RDz (Rational Developer for Z) provided an Eclipse-based IDE enabling a common ‘look and feel’ for Allstate developers
Jon Sayles’ RDz workshops had helped us keep track on product evolution and best practices
Allstate’s Enterprise Tools team had RTC experience for distributed systems
Implementation approach for rollout of RTC and RDz

- Historical approach to mainframe lifecycle work had been for all teams to migrate together as part of a ‘big bang’
  - This led to frequent delays
  - Large complexity
  - Extended code freezes severely hampering enterprise development work

‘New’ approach
- ‘Pioneer team’ would build a Minimum Viable product (MVP) of RTC and RDz
- Remaining teams would then ‘reuse’ and build on the MVP
- Pioneer team = Billing
- This is the Allstate P&C Billing application and had high complexity and name recognition
Implementation approach for rollout of RTC and RDz

• Billing had tracked on Jon Sayles’ RDz workshops to keep current with IBM’s devops offerings and product evolution
• Billing had successfully recompiled entire codebase to V5.2 in 2016
• Billing team included mainframers who had significant experience with git, and TFS for distributed systems as well as SCLM and Endeavor
• Analysis tools such as EzSource/ADDI were not available at Allstate so codebase knowledge was based on SME experience, together with learnings from the recompiling work in 2016
Billing’s approach to migration

1. A successful minimum viable product (MVP) of RTC was key to buy-in from other teams
2. Seeing Billing running successfully on RTC and RDz would be an important confidence-building measure for enterprise adoption
3. We re-used the existing RTC infrastructure based on Windows Server and Oracle database as we didn’t have zLinux on mainframe
4. We committed to a cutover date to move ‘in flight’ work from SCLM to RTC
5. The 1 week code freeze allowed IBM to migrate code using zImport and minimized the impact/downtime on our development work
6. We aligned the migration date to the Allstate release calendar, enabling us to complete a number of significant projects in SCLM
7. ‘In flight’ work aka projects started in SCLM and finished in RTC was clearly identified and risk-assessed according to business need
Billing’s approach to migration

- Billing, like most mainframe teams uses Waterfall for software development, deploying six major and six minor releases each year.
- Large number of yearly release requires speed to market, agility and quality.
- Initially 24 streams were configured in RTC to accommodate the waterfall releases but quickly realized that 12 would suffice.
- The ability to change stream setup, together with the built-in flexibility of RTC, meant that Billing could easily move to Agile development (Product-based mindset) with a different stream configuration in future without being tied to the initial configuration.
Billing’s approach to migration

• It was decided not to use ISPF/green screen with RTC as Allstate saw ‘green screen’ replacement as transformational
• We evaluated both RTCz client (Eclipse) and RDz (Rational Developer for Z) as potential IDEs
• The rich tooling and feature set of RDz was ‘jaw dropping’ for Allstate mainframers and made it an easy sell.
• As we built out RTC environments we incrementally installed RTCz and RDz clients and demo’ed core functionality in both IDEs
• Rather than producing copious amounts of documentation, we focused instead on delivering targeted learning and curated content from the Jon Sayles’ workshops
• Co-locating our developers worked well with two developers with git and TFS knowledge in Belfast and one in Chicago
• Used company development/learning time for coaching
Billing’s approach to migration

- 3 Billing projects in SCLM viz SCBILL, SCOUTPUT (shared) and SCEDP would be moved to a single project in Team Concert
- SCOUTPUT – moved Billing-owned components to TCBILL
- SCOUTPUT – moved shared components needed by Billing to TCBILL
- SCOUTPUT – left the Customer Communications (CC) team-owned components and the shared ones needed by CC in SCOUTPUT
- Customer Communications will be moving to RTC in Q2, 2018
- When Billing started our migration, RDz was not available as an enterprise software drop so it had to be installed manually.
- This required significant experimentation and was time consuming
Key points for Allstate’s RTC enterprise migration

• 80+ SCLM projects would be migrated to RTC
• Over 200k members would be migrated
• 500+ developers would be impacted by the migration
• The Pioneer team (Billing) would migrate first, working with the IBM project and Allstate tools and environments teams
• After successfully building the MVP, the remaining Allstate teams would rollout in ‘waves’ based on business need, complexity and risk assessment
• An ‘Early Adopter’ initiative identified key resources in teams with complex SCLM projects, who would partner with Billing to get a ‘head start’ on the requirements gathering and understanding the setup process
Key points for Allstate’s RTC enterprise migration

- Around the same time as we rolled out RTC, Billing successfully completed a Proof of Concept for Urbancode Deploy (UCD) on z/OS
- The original vision had been to rollout RTC and UCD together, but funding was not available for UCD, and Allstate’s distributed systems had already adopted Jenkins, Octopus Deploy and Cloud Foundry
- The UCD PoC evaluation was written up along with findings and recommendations
- Allstate’s homegrown deployment system for z/OS (APTF) was well established and integrated with SCLM
- As part of the RTC migration, RTC was seamlessly integrated with no additional customization required
Billing rollout

- Billing successfully completed the April Major release using SCLM and APTF
- After completing User Acceptance (UAT), work items were created for the ‘in flight’ work
- IBM used zImport to move the code from SCLM to RTC
- IBM rolled out on-site training to Allstate teams in both Belfast and Chicago
- Billing completed first release out of RTC in May using APTF
- An enterprise telematics program with significant Billing involvement kicked off just as Billing moved to RTC
- Billing developers started using RTC and RDz immediately for this development work
Billing’s first 90 days using RTC and RDz

- Very steep learning curve for most developers, even after the pre-deployment coaching/mentoring and the on-site IBM training
- Ongoing ‘1 on 1’ in-person coaching and mentoring was vital to successful on-boarding and worked especially well in co-located teams in Belfast and Chicago
- ‘Learning at a distance’ via Skype, Webex and phone calls worked significantly less well
- ‘Self-directed’ learning using job aids and online courses again worked less well
- We set a team goal that each developer should successfully install production code, having written it in RTC – ACHIEVED
- Individual buy-in from developers really only occurred when they successfully delivered production code that ran as well as that written ‘old way’ on green screen
Billing’s first 90 days using RTC and RDz

- Alongside the steep learning curve for RTC and RDz, we enhanced the end user experience with usability improvements such as third party code compare tools (Beyond Compare), code snippets and debugging capabilities
- We also installed the IBM Data Studio plugin, extending RDz with additional tooling for Db2 work

Smart DB2 Development Tooling
Powerful, flexible graphical interface to DB2 and SQL tooling for:
- DB2 Table/View/Index Analysis, Design – for the DB2 Data Model and for test data management and manipulation:
  - Edit/View
  - Simple Table row/column sub-setting
- SQL code and test
  - Code embedded SQL directly within COBOL, PL/I and Assembler programs
  - Statement content assist from the DB2 Catalog
  - Test/Run/Tune SQL directly from COBOL or PL/I programs
  - Code SQL with graphical tooling
  - Run existing SPUFI files
  - Interactively Code/Test/Tune SQL statements
  - Export SQL statement results to:
    - Spreadsheets
    - HTML
    - XML
Billing’s first 90 days using RTC and RDz

- We used RDz code analysis features to rewrite Allstate’s legacy testing tools and have them reference RTC rather than SCLM

- Integration with APTF (homegrown deployment tool) was seamless; whilst not fully automated like Urbancode Deploy (UCD), the time savings for creating APTF installation packages were significant (estimated 75% time saving)
Billing’s first 90 days using RTC and RDz

- Developers liked the software analyzer feedback, showing severity of issues and the exact lines where issues arose

- They also liked the code review feedback, where RDz’s editors flagged statements in real time
What worked for Billing @ Allstate

• Leadership support

• Getting developers early and frequent access to RDz and RTC; choosing not to install ISPF/’green screen’ support for RTC

• A ‘lean startup’ mentality! Clear vision of the ‘big picture’, clear communication of the direction of travel and constant encouragement and bench marking against peers

• A Product Mindset – experimenting and testing assumptions along with a continuous feedback loop from end users of what’s working well and more importantly what’s not

• Similarly with RTC, building an MVP and then improving by simplifying it, removing streams we found we didn’t need
What worked for Billing @ Allstate

• Documentation combined with in-person 1-1 coaching and mentoring
• Setting up an easy to use repository (Sharepoint)
• Customizing the documentation for Allstate’s RTC environment (screenshots matching what was built out at Allstate)
• Role-based training
• Single Points of Contact (SPOCs)
• Mainframers with experience of git and TFS found on-boarding easier than most developers and helped share learning and findings
Current status with RTC @ Allstate

• Billing has successfully completed 10 months of successful releases with RTC and APTF

• All Billing developers are using RDz with Data Studio, Beyond Compare and other plugins installed

• Remaining Allstate mainframe teams are moving to RTC this year (2018) with next tranche at end of Q1

• Billing is building out CI/CD pipelines to deliver mainframe code to Sonarqube for static code analysis

• Billing is also building out CI/CD pipelines using Jenkins and Cloud Foundry to perform vulnerability scans with Veracode
Next steps for Allstate

- Expanded use of RTC and full adoption by all mainframe teams
- Further extending the usage of RDz features e.g. zunit and code coverage
- API-enabling the mainframe using z/OS Connect [successful Proof of Concept completed in Q4 2017]
- Sunsetting APTF and replacing with Urbancode Deploy (and Release?)