BACK TO BASICS
Child Welfare Information Technology Systems Managers and Staff Webinar Series

Impact of Development Methodologies
Software Frameworks and Platform Choices on Release Management

Thursday, August 29, 2013
2:30 – 3:30 p.m. EDT

Joyce Rose, ICF
Tom Kine, MN
Iroabuchi Arum, OH
Fred Crawley, OH
Thomas Hammons, OH

Presented by ICF International under contract with the U.S. Department of Health and Human Services Administration for Children and Families, Children’s Bureau
Back to Basics Series

- One webinar per month between April and September, 2013
- Target audience
  - Child welfare IT systems managers and staff
  - New and experienced
- Recorded webinars
Back to Basics Series (continued)

- Webinar 1: What is a Child Welfare Information Technology Systems Manager (April)
- Webinar 3: The Child Welfare Information Technology System Project Lifecycle, Part 2 (June)
- Webinar 4: Common Pitfalls and How to Avoid Them (July)
- Webinars 5: The Impact of Development Methodologies, Software Frameworks and Platform Choices on Release Management (August)
- Webinar 6: To be determined (September)
Participating in Today’s Webinar

• Questions and comments by telephone
• Questions via chat
• If you have additional questions, please contact Joyce Rose after the webinar

Joyce@kassets.com
Attendee Poll

Who is attending today’s webinar?

- State Child Welfare Information System (CWIS) Project Manager
- State Child Welfare Information System (CWIS) Program Manager
- State Child Welfare Information System (CWIS) Technical Manager
- State Child Welfare Information System (CWIS) Project Staff
- ACF/Children’s Bureau Personnel
Today’s Agenda

• Format
• Introductions
• State participant discussion
• Attendee Q & A
• Wrap up
Introductions

• Tom Kine, MN
• Fred Crawley, OH
• Thomas Hammons, OH
• Iroabuchi Arum, OH
• Joyce Rose, ICF International
I'd like to start with a cartoon.

It's about a guy who shows a cartoon before giving a boring presentation.
State Presentation: Minnesota

- State background
- Development methodology
- Software framework
- System platform
- Impacts upon release management
- Lessons learned
- If we had to do it all over again, we would (fill in the blank)
State Background

Child Protection
Foster Care
Adoption
Children’s Mental Health
Other Child Welfare Programs
Adult Maltreatment Reporting
Waiver Claiming
Other Adult Services
State Background (cont.)

Minnesota is state-supervised and county-administered

The Minnesota SACWIS (Social Services Information System – SSIS) is used by 80 individual counties, 7 additional counties organized into 2 regions, and 2 Indian tribes

Over 6,000 agency social workers, case aides and DHS staff use SSIS to track cases involving 272,000 individuals annually
SSIS started planning in 1995; no statewide social services system existed

Years of negotiation with counties

Counties wanted to keep their data separate from each other
Development Methodology

Waterfall for fiscal-related development

Rapid Application Development (RAD) for social-services-related development

Agile for managing bug fixes and minor enhancement requests
**Development Methodology (cont.)**

- **Waterfall for fiscal-related development**
  - **Pros**
    - Good with well-defined and understood requirements
    - High degree of control for financial systems considerations
    - Allows creation of detailed testing plans based on the design documentation
  - **Cons**
    - Rigid, hard to adjust for changes in requirements
Development Methodology (cont.)

- RAD for social–services–related development
  - Pros
    - Good for development focuses on user–interaction
    - Fast iterations focused on incremental improvements
  - Cons
    - Testing can be a challenge
Development Methodology (cont.)

Agile for managing bug fixes and minor enhancement requests

- Pros
  - Flexible as compared to waterfall
  - Quick turn–around through iterations
  - QA through–out process – fewer defects

- Cons
  - Hard to get/maintain documentation
Software Framework

Windows Application (Delphi)

Oracle Database

Windows Server Application (Delphi)

User Interface
Connectivity
Application Logic
Database Connectivity

Client

Application Server

RDBMS
Data
Software Framework (cont.)

Typical County/Tribe Configuration

- County Financial System Host
  - AS/400
  - IBM 3090
  - HP3000
  - SQL Server etc.

- Social Worker Desktop

DHS

Centralized Databases
- SWNDX
- SSIS Repository
- State Adoptions

100+ Servers
- Database Management & Support
- Server Management & Support
- QA/Test
- Development

40+ Interfaces
- MAXIS
- MMIS
- MN-ITS
- PRISM
- EDMS
- Data Warehouse
- Licensing
- Dept. of Health
- Dept. of Finance

86 county and tribal application and database servers
Software Framework (cont.)
Impacts on Release Management

- Distributing software to 6,000+ desktops is a big deal
- Historically, about 3 releases every 2 years
- SSIS uses a custom-written application launcher
  - Launcher runs, checks server for updates, if update found, patches or replaces existing .exe, launches application
  - Patch technology minimizes impact of pulling down entire .exe over county networks
Impacts on Release Management (cont.)

Agile project management has had a huge impact on release management

Converted to a quarterly release cycle for bug fixes and minor enhancements

Develop new functionality in it’s own branch and merge in when ready

- Releases were delayed in the past due to unpredictability of new development
- The later they were, the more behind we got – bug fixes continued to be pushed into release
Agile Project Management

• Background on its introductions into SSIS
• 2009 – Attended state IT Symposium with SSIS GUI Development Supervisor
• Started Agile planning
  o Joined state self–study group for Scrum–master
  o Started educating project staff on Agile
  o Developed plan to implement Agile for bug reports
• 2011 – Started using Agile
• 2013 – 3\textsuperscript{rd} or 4\textsuperscript{th} release in a row delivered on time
Process for Managing Problem Reports

<table>
<thead>
<tr>
<th>April</th>
<th>May</th>
<th>June</th>
<th>July</th>
<th>August</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
</tr>
<tr>
<td>34</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Managers
- Prioritize

Functional Analysts
- Generate PRs

Systems Analysts
- Review PRs

Developers
- Develop, code review, unit test

Quality Assurance
- Generate PRs
- System test

Enterprise Operations Team
- Distribution
- Install

DBA
- Help Desk
- Generate PRs

Trainers
- Update training curriculum

Production Control
- QA Server Mgt
- Build Mgt
- Pilot Coord.
- Release Mgt - Schedule, Release Notes, etc.

Mentor Coordinators
- <2 weeks
- New Development Merge Points
- For 3rd Sprint
- Could shorten or eliminate this sprint based on the needs of new development.

Release Cycle
- Sprint Planning Start
- Development Start
- Sprint Planning Start
- Pilot Candidate
- Pilot
- Statewide Release
### Process for Managing New Development

<table>
<thead>
<tr>
<th>April</th>
<th>May</th>
<th>June</th>
<th>July</th>
<th>August</th>
<th>September</th>
<th>October</th>
<th>November</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
</tbody>
</table>

- **Sprint Planning**
- **Development Start**
- **New Development Merge Points**
- **For 3rd Sprint**
- **Problem Reports from Planning Session**
- **New PRs Added**

#### Roles and Responsibilities

- **Managers**: Approve project
- **Functional Analysts**: Specify requirements
- **Systems Analysts**: Design and write specification (indefinite time)
- **Developers**: Develop from specifications
- **Quality Assurance**: Test per design
- **Enterprise Operations Team**: Regression test
- **DBA**: Distribution
- **Help Desk**: Install
- **Trainers**: User support
- Revise training curriculum, schedule & deliver training

---

Release 11.2

<table>
<thead>
<tr>
<th>April</th>
<th>May</th>
<th>June</th>
<th>July</th>
<th>August</th>
<th>September</th>
<th>October</th>
<th>November</th>
</tr>
</thead>
</table>

Release 11.3

<table>
<thead>
<tr>
<th>April</th>
<th>May</th>
<th>June</th>
<th>July</th>
<th>August</th>
<th>September</th>
<th>October</th>
<th>November</th>
</tr>
</thead>
</table>

- Could shorten or eliminate this sprint based on the needs of new development.
## Planned vs. Actual Release Dates

<table>
<thead>
<tr>
<th>Release</th>
<th>Planned Date</th>
<th>Actual Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.4</td>
<td>November</td>
<td>November 2011</td>
</tr>
<tr>
<td>12.1</td>
<td>February</td>
<td>February 2012</td>
</tr>
<tr>
<td>12.2</td>
<td>May</td>
<td>May 2012</td>
</tr>
<tr>
<td>12.3</td>
<td>August</td>
<td>October 2012</td>
</tr>
<tr>
<td>12.4</td>
<td>November</td>
<td>December 2012</td>
</tr>
<tr>
<td>13.1</td>
<td>February</td>
<td>February 2013</td>
</tr>
<tr>
<td>13.2</td>
<td>May</td>
<td>May 28, 2013</td>
</tr>
<tr>
<td>13.3</td>
<td>August</td>
<td></td>
</tr>
</tbody>
</table>
Story Points by Release

![Bar Chart: Story Points Completed by Release](image)

- Release 11.4: 237
- Release 12.1: 445
- Release 12.2: 358
- Release 12.3: 419
- Release 12.4: 431
- Release 13.1: 428
- Total: 237 + 445 + 358 + 419 + 431 + 428
### Developer Work Capacity by SSIS Release Cycle

<table>
<thead>
<tr>
<th>Developer Specialty Area</th>
<th>Number of Developers</th>
<th>Percentage of Time Available New Development</th>
<th>Percentage of Time Available Problem Reports</th>
<th>Functional Area Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUI Developers</td>
<td>7</td>
<td>50%</td>
<td>50%</td>
<td>40%</td>
</tr>
<tr>
<td>Database Developers</td>
<td>4</td>
<td>70%</td>
<td>30%</td>
<td>40%</td>
</tr>
<tr>
<td>Architecture Developers</td>
<td>2</td>
<td>90%</td>
<td>10%</td>
<td>40%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Functional Area Allocation</th>
<th>Fiscal</th>
<th>Worker</th>
<th>Adoptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>G11</td>
<td>40%</td>
<td>40%</td>
<td>20%</td>
</tr>
<tr>
<td>G22</td>
<td>40%</td>
<td>40%</td>
<td>20%</td>
</tr>
<tr>
<td>G33</td>
<td>40%</td>
<td>40%</td>
<td>20%</td>
</tr>
</tbody>
</table>

#### Weeks

<table>
<thead>
<tr>
<th>Release Cycle</th>
<th>Weeks</th>
<th>Hours by Functional Area*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sprint 1</td>
<td>4</td>
<td>Fiscal: 251, Worker: 251</td>
</tr>
<tr>
<td>Sprint 2</td>
<td>4</td>
<td>Fiscal: 251, Worker: 251</td>
</tr>
<tr>
<td>Sprint 3</td>
<td>5</td>
<td>Fiscal: 314, Worker: 314</td>
</tr>
</tbody>
</table>

Total working hours by release cycle by functional area: 816 Fiscal, 816 Worker, 407 Adoptions

**Story Points by Functional Area**

<table>
<thead>
<tr>
<th>Fiscal</th>
<th>Worker</th>
<th>Adoptions</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>25</td>
<td>125</td>
</tr>
<tr>
<td>163</td>
<td>163</td>
<td>81</td>
<td>407</td>
</tr>
</tbody>
</table>

*Formula for Hours by Functional Area = Number of developers * percentage of time available for PRs * functional area allocation * average working hours per week per developer * number of weeks in sprint

Average working hours per week per developer = 32
Number of hours in a story point = 5

### Story Points by Release by Functional Area

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1</td>
<td></td>
<td>214</td>
<td>176</td>
<td>30</td>
<td>3</td>
<td>20</td>
<td>2</td>
<td>445</td>
</tr>
<tr>
<td>12.2</td>
<td></td>
<td>165</td>
<td>120</td>
<td>49</td>
<td>24</td>
<td>24</td>
<td>358</td>
<td></td>
</tr>
<tr>
<td>12.3</td>
<td></td>
<td>274</td>
<td>99</td>
<td>31</td>
<td>24</td>
<td>15</td>
<td>410</td>
<td></td>
</tr>
<tr>
<td>12.4</td>
<td></td>
<td>208</td>
<td>130</td>
<td>37</td>
<td>24</td>
<td>32</td>
<td>431</td>
<td></td>
</tr>
<tr>
<td>13.1</td>
<td></td>
<td>269</td>
<td>94</td>
<td>41</td>
<td>27</td>
<td>24</td>
<td>455</td>
<td></td>
</tr>
<tr>
<td>Grand Total</td>
<td></td>
<td>1130</td>
<td>619</td>
<td>188</td>
<td>54</td>
<td>115</td>
<td>2</td>
<td>2108</td>
</tr>
<tr>
<td>Percent of Total</td>
<td></td>
<td>53.6%</td>
<td>29.4%</td>
<td>8.9%</td>
<td>2.6%</td>
<td>5.5%</td>
<td>0.1%</td>
<td>100.0%</td>
</tr>
</tbody>
</table>

**Average Story Points Per Release**: 422
Lessons Learned

• Establishing fixed release dates and hitting them builds credibility
• Fixed release dates facilitate work planning
• With credibility, users are more willing to negotiate schedule
• Project team functions more efficiently when a rhythm is established
• Software quality goes up as predictable release dates reduce pressure on the team
If we had to do it all over again, we would (fill in the blank)

- Establish a fixed release cycle as soon as possible
- Hit the published release dates – every time
  - Do not push last-minute fixes into the release
  - Negotiate scope at the front of each scheduled release
- Make it a priority to develop metrics measuring quantity and quality of the work
State Presentation: Ohio

- State background
- Development methodology
- Software framework
- System platform
- Impacts upon release management
- Lessons learned
- If we had to do it all over again, we would (fill in the blank)
State Background

- County administered (88 counties)
- Web–based
- Working on Federal compliance
- Mix of state and vendor development staff
- Integrated project team (Child Welfare and IT)
- Over 7,000 users
- New key initiatives
  - Private agencies Phase 2
  - IV–E court rollout
  - Field assistant mobile application
  - OptimalJ refactoring
Development Schedule
# IPT Schedule

<table>
<thead>
<tr>
<th>Week 1</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JAD Prep</td>
<td>JAD</td>
<td>JAD Prep / Design</td>
<td>JAD</td>
<td>Design</td>
</tr>
<tr>
<td></td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
</tr>
<tr>
<td></td>
<td>Create Test Scripts</td>
<td>Create Test Scripts</td>
<td>Create Test Scripts</td>
<td>Create Test Scripts</td>
<td>Create Test Scripts</td>
</tr>
<tr>
<td></td>
<td>Last Dev Day</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dev / Test Kickoff</td>
<td>Create Interim Build</td>
<td>Deploy Interim to SYSTEST and UAT</td>
<td>Test SYSTEST &amp; UAT</td>
<td>Test SYSTEST &amp; UAT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Week 2</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JAD Prep / Design</td>
<td>JAD</td>
<td>JAD Prep / Arch Review</td>
<td>JAD / BA Review</td>
<td>Design</td>
</tr>
<tr>
<td></td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
</tr>
<tr>
<td></td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST / BA Review</td>
<td>CGI SYSTEST</td>
</tr>
<tr>
<td></td>
<td>Test SYSTEST &amp; UAT</td>
<td>Test SYSTEST &amp; UAT</td>
<td>Test SYSTEST &amp; UAT</td>
<td>Test SYSTEST &amp; UAT</td>
<td>Test SYSTEST &amp; UAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status Meeting</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Week 3</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JAD Prep</td>
<td>JAD</td>
<td>JAD Prep / Design</td>
<td>JAD</td>
<td>Design</td>
</tr>
<tr>
<td></td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
</tr>
<tr>
<td></td>
<td>Create Test Cases</td>
<td>Create Test Cases</td>
<td>Create Test Data/Scripts</td>
<td>Create Test Data/Scripts</td>
<td>Create Test Data/Scripts</td>
</tr>
<tr>
<td></td>
<td>Last Fix Day</td>
<td>Create Final Build</td>
<td>Deploy to SYSTEST</td>
<td>Test SYSTEST</td>
<td>Deploy to UAT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Week 4</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JAD Prep / Design</td>
<td>JAD</td>
<td>JAD Prep / Arch Review</td>
<td>JAD / BA Review</td>
<td>Design</td>
</tr>
<tr>
<td></td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
<td>Construction</td>
</tr>
<tr>
<td></td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST</td>
<td>CGI SYSTEST / BA Review</td>
<td>CGI SYSTEST</td>
</tr>
<tr>
<td></td>
<td>Test UAT</td>
<td>Test UAT</td>
<td>Test UAT</td>
<td>Deploy to Production</td>
<td>Deploy to Staging</td>
</tr>
</tbody>
</table>
Software Framework & Platform

Legend

- Custom Code
- OJ Generated Code
- Third Party Frameworks
- OJ Generated Features

Presentation Tier

- JSP Files
- ActionServlet
  - ActionClasses
    - OJ ActionClasses
      - OJ Business Facades
        - Store, Clear, Create*, retrieveForUpdate*, and delete*, pruning
      - saveBean, loadBean, obtainBusinessFacade

Business Tier

- Validate
- OJ FormBeans extends FormBeans
  - OJ Session Beans
    - OJ Entity Beans
    - Framework Session Beans (ReadOnly, RDA etc.)
    - DAOs
- Custom Code

Database Tier

- Transactional Database

* indicates other API variations
Impacts on Release Management

- Integrated team approach and Agile inspired practices offer flexibility in development process and schedule
- Upfront planning to incrementally introduce functionality ensures greater buy in and success for larger initiatives/complex functionality
- Release schedule milestones are checked regularly; methods for delaying releases and implementing hot fixes are in place in the event they are needed
- Release progress communicated to users in multiple venues: release note and database change documentation are provided to users in advance
- Significant outreach to county users to participate in ongoing requirements documentation, design and testing process
Lessons Learned

- Collaboration is “KEY”
- Users need mechanisms to provide constant feedback on functionality (user group meetings, testing build calls, help desk, survey data, focus groups, etc.)
- Self-managing teams meet daily to review progress and resolve issues
- A combination of manual and automated testing with collaboration of county, state and development staff
- Risk management identification, communication and mitigation on an ad hoc, daily and milestone basis
- If functionality is not being used or meeting needs, team stays open to “get it right”
- Team needs protocols for reducing scope or moving/adding resources to support critical priorities when issues arise
If we had to do it all over again, we would (fill in the blank)

- Streamline the front end to ease use for caseworkers in the field
- Plan for incremental roll out of modules (rather than entire system at once)
- Formalize coordination of dependencies between business functions and user groups during planning phase
# IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Prepare for JAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>Finalize the pre-JAD Screen Specs</td>
</tr>
<tr>
<td>Participants/Role</td>
<td>IPT Leads</td>
</tr>
</tbody>
</table>
| Tasks       | Review Existing Materials  
|             | Conduct Preliminary Analysis  
|             | Develop Clarification Questions  
|             | Prioritize Screen Specs and/or Quality Center Items |
| Input(s)    | Business Priorities  
|             | Quality Center Items  
|             | Preliminary Screen Specifications  
|             | Workflows |
| Output(s)   | JAD Schedule  
|             | JAD Agenda  
|             | List of Questions |
## IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Conduct JAD Session/Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>To obtain consensus that each screen spec covered is clear, complete and ready for design</td>
</tr>
<tr>
<td>Participants/Role</td>
<td></td>
</tr>
<tr>
<td></td>
<td>➢ IPT Leads</td>
</tr>
<tr>
<td></td>
<td>➢ Business/Technical SMEs as required</td>
</tr>
<tr>
<td>Tasks</td>
<td>➢ Conduct Walkthroughs of Screen Specifications and Report Mockups</td>
</tr>
<tr>
<td></td>
<td>➢ Call for Consensus on Screen Spec Completeness</td>
</tr>
<tr>
<td></td>
<td>➢ Update Screen Specs in Place (optional)</td>
</tr>
<tr>
<td>Input(s)</td>
<td>➢ JAD Schedule</td>
</tr>
<tr>
<td></td>
<td>➢ JAD Agenda</td>
</tr>
<tr>
<td></td>
<td>➢ List of Questions</td>
</tr>
<tr>
<td></td>
<td>➢ Preliminary Screen Specifications</td>
</tr>
<tr>
<td>Output(s)</td>
<td>➢ Create / Update Quality Center items</td>
</tr>
<tr>
<td></td>
<td>➢ Updated Screen Specifications</td>
</tr>
<tr>
<td></td>
<td>➢ Use Case Consensus</td>
</tr>
</tbody>
</table>
## IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Conduct JAD/BA Review</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>To obtain consensus that each screen specs design meets the intent of the requirements and is ready for construction</td>
</tr>
</tbody>
</table>
| **Participants/Role** | - IPT Lead  
                          - Business/Technical SMEs as required |
| **Tasks**    | - Conduct Walkthroughs of Screen Specs Realizations  
                          - Review Screen flows to confirm all possible paths and correct as required  
                          - Walkthrough Report flow and Content  
                          - Review Boundary Scenarios to Make Sure the Requirements and Design Handle Them Correctly |
| **Input(s)** | - Updated Screen Specifications  
                          - Updated Report Specifications |
| **Output(s)**| - Updated Quality Center items  
                          - Updated Screen Specifications  
                          - Updated Report Specifications  
                          - Optimal/J Artifacts |
## IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Conduct Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>To review requirement documents and develop a technical design that meets the intent of the requirements</td>
</tr>
</tbody>
</table>
| Participants/Role | ➢ IPT Leads  
➢ Business/Technical SMEs as required |
| Tasks | ➢ Review Screen Specs / Quality Center Items  
➢ Compare Screen Flows to Requirements  
➢ Review Screen and Report Specifications for Each Screen and Report Related to the Screen Spec  
➢ Review Boundary Scenarios  
➢ Create Optimal/J Artifacts |
| Input(s) | ➢ Update Quality Center Items  
➢ Updated Screen Specifications  
➢ Use Case Consensus |
| Output(s) | ➢ Optimal/J Artifacts |
## IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Conduct Architecture / Design Round Table</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>To obtain consensus that technical design meets the SACWIS architecture guidelines and module inter-dependencies are resolved</td>
</tr>
</tbody>
</table>
| **Participants/Role** | ➢ IPT Leads  
➢ Additional SACWIS Team Leads |
| **Tasks**      | ➢ Share and Review Technical Design  
➢ Discuss Inter-dependencies Across Modules |
| **Input(s)**   | ➢ Optimal/J Artifacts  
➢ Updated Quality Center items  
➢ Updated Screen Specifications |
| **Output(s)**  | ➢ Design Consensus  
➢ Updates to Optimal/J Artifacts  
➢ New Requirements / Updates to Frameworks |
# IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Construction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>To develop detailed design and code artifacts that meets the intent of the requirements, high level design and is ready for UAT</td>
</tr>
</tbody>
</table>
| **Participants/Role** | ➢ IPT Development Team  
➢ IPT Leads |
| **Tasks**      | ➢ Review Requirements and High Level Design Documents  
➢ Develop Detailed Design / Implementation Approach  
➢ Review Implementation Approach with IPT Leads  
➢ Develop Code Artifacts and Complete Unit Testing  
➢ Review Code Artifacts with IPT Leads as Required |
| **Input(s)**   | ➢ Updated Optimal/J Artifacts  
➢ Updated Quality Center Items  
➢ Updated Screen Specifications  
➢ Update Report Specifications |
| **Output(s)**  | ➢ Code Artifacts  
➢ Update Quality Center Items |
# IPT Process

<table>
<thead>
<tr>
<th>Activity</th>
<th>Internal System Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>To develop test scripts and data population scripts that verifies whether the code artifacts meets the intent of the requirements</td>
</tr>
<tr>
<td><strong>Participants/Role</strong></td>
<td>IPT Test Team, IPT Business Lead</td>
</tr>
<tr>
<td><strong>Tasks</strong></td>
<td>Review Requirements and High Level Design Documents, Develop Test Scripts, Develop Data Population Scripts, Review Test Scripts with IPT Business Lead</td>
</tr>
<tr>
<td><strong>Input(s)</strong></td>
<td>Quality Center items, Updated Quality Center items, Updated Screen Specifications, Update Report Specifications</td>
</tr>
<tr>
<td><strong>Output(s)</strong></td>
<td>Test Scripts (uploaded to Quality Center), Data Population Scripts</td>
</tr>
</tbody>
</table>
Supporting Information

• **Classics**
  - [http://www.projectsmart.co.uk/docs/chaos-report.pdf](http://www.projectsmart.co.uk/docs/chaos-report.pdf)
    - A copy of the classic 1995 report on SW project failure by the Standish Group

• **Agile**
  - [http://www.agilealliance.org](http://www.agilealliance.org)
    - The home of the Agile Alliance, with a great library of Agile articles
    - Effective user stories by Mike Cohen
    - A Retrospective: Managing Agile Transition in Government Bureaucracy by Brandon Raines and Judy Wankerl
Supporting Information

• Agile (cont.)
  • http://www.poppendieck.com
    • Mary and Tom Poppendieck’s home website, excellent material on Lean
  • http://xpday3.xpday.org/slides/LeanTutorial.pdf
    • Overview and Tutorial on Lean by Mary Poppendieck
  • http://leansoftwareengineering.com
    • Good articles on Lean from multiple authors
  • http://leansoftwareengineering.com/ksse/scrum-ban/
    • On Kanban workflow by Corey Ladas, author of Scrumban
Attendee Discussion

Q&A
Wrap Up

• What was accomplished today?

• What’s next?

• Reminder: Recorded versions of each of the six webinars are being made available at: