Adapting and transforming critical workflows and digital task management: an ITSD case study

Dr. Lewis J. McGibbney, Enterprise Search Program Technologist
Web and Mobile Application Development Group (172B)
Application, Consulting, Development and Engineering Section (1722)

Jet Propulsion Laboratory
California Institute of Technology

- JPL's Enterprise Search Program Technologist (Data Scientist III).
- Information retrieval, Web search, databases, natural language processing, semantic technologies, software and data engineering.
- Apache Software Foundation Member – Open Source Software practitioner.
- History of participating and leading standardization efforts across W3C, OGC, NASA's ESDSWG, OASIS, ESIP

Part 1: Legacy data engineering

Overview of problems

Legacy data engineering platform purpose

a.k.a *the connector framework*

1. **Acquire (pull model) heterogeneous data from heterogeneous data systems**: several flavors of RDBMS, content/archival management platforms i.e. Docushare, Alfresco, arbitrary remote file systems, etc.

2. **Execute predetermined, static data engineering pipelines:** i.e. Extract, Transform and Load (ETL) data(sets) before passing to an indexing engine.

Legacy data engineering platform source code

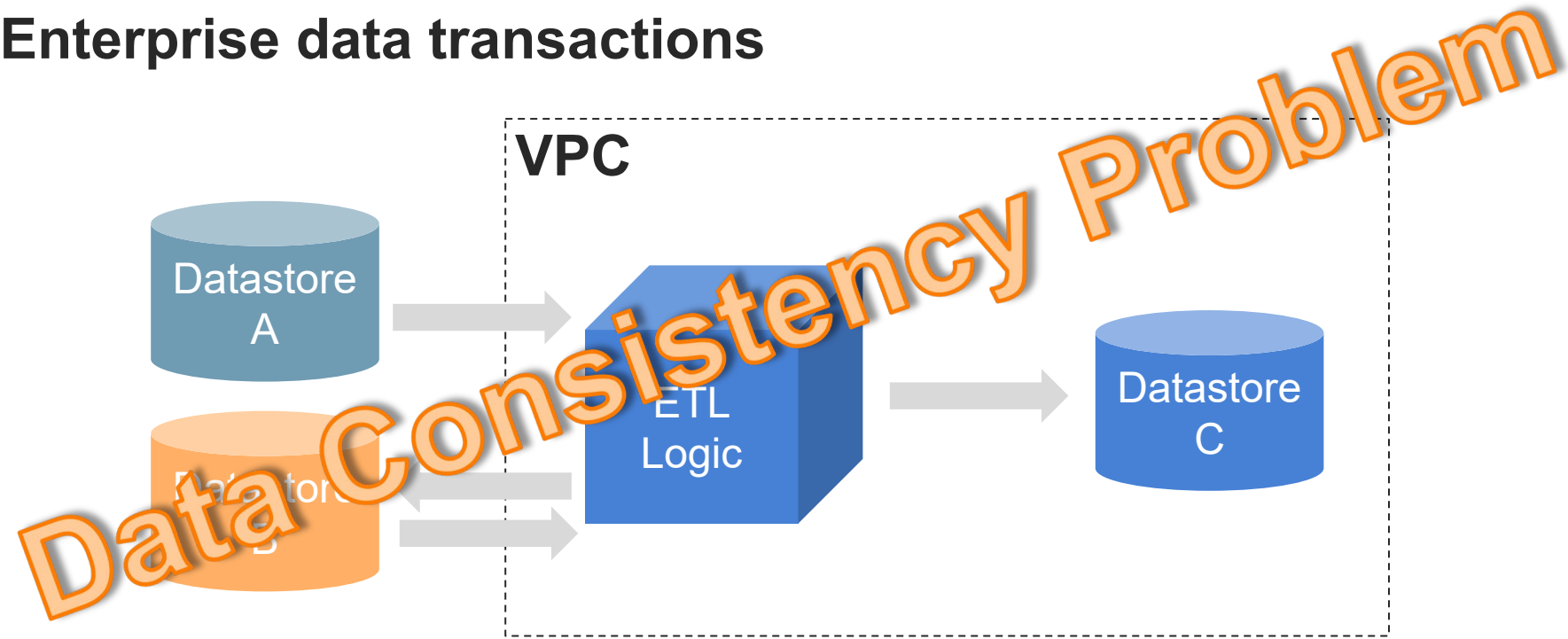
```
      62 text files.
      62 unique files.
      12 files ignored.

github.com/AlDanial/cloc v 1.86  T=0.11 s (520.9 files/s, 78523.8 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Python                          44           1468           1518           4974
YAML                             3              0              0            169
SQL                              7             21            126            137
Markdown                         1             17              0             66
Dockerfile                       1             12              2             58
make                             1              7              0             17
-------------------------------------------------------------------------------
SUM:                            57           1525           1646           5421
-------------------------------------------------------------------------------
```

Legacy data engineering platform shortcomings

1. **Deprecation** – Python 2.7
2. **No concept of a unit of work i.e. task** – very difficult to debug
3. **Overreliance on CRON -** ignorance of differences between ***task automation*** and ***workflow management***
4. **No tests** – unit, integration, mock, smoke, regressions
5. **User unfriendly –** no user interface… period
6. **Unable to scale** – no standard application programming interface
7. **No documentation** - difficult and time consuming to teach new engineering staff

- **Failures -** retry upon failure (how many times? when/how often?)
- **Monitoring -** workflow (or even task) status. How long does each process (workflow or task) take to run? What can we learn about the workflow execution?
- **Dependencies**
  - **Data dependencies**: upstream data is missing, connection to Oracle fails, etc.
  - **Execution dependencies**: job 2 is meant to run after job 1 finishes
- **Scalability -** there is no centralized scheduler between different CRON environments
- **Deployment -** How do we deploy new changes constantly without breaking some workflow?
- **Processing historical data -** backfill/re-run historical workflow(s)

Enterprise data transactions



CRON limitations

1. **Error handling** - If a job fails, what should happen?

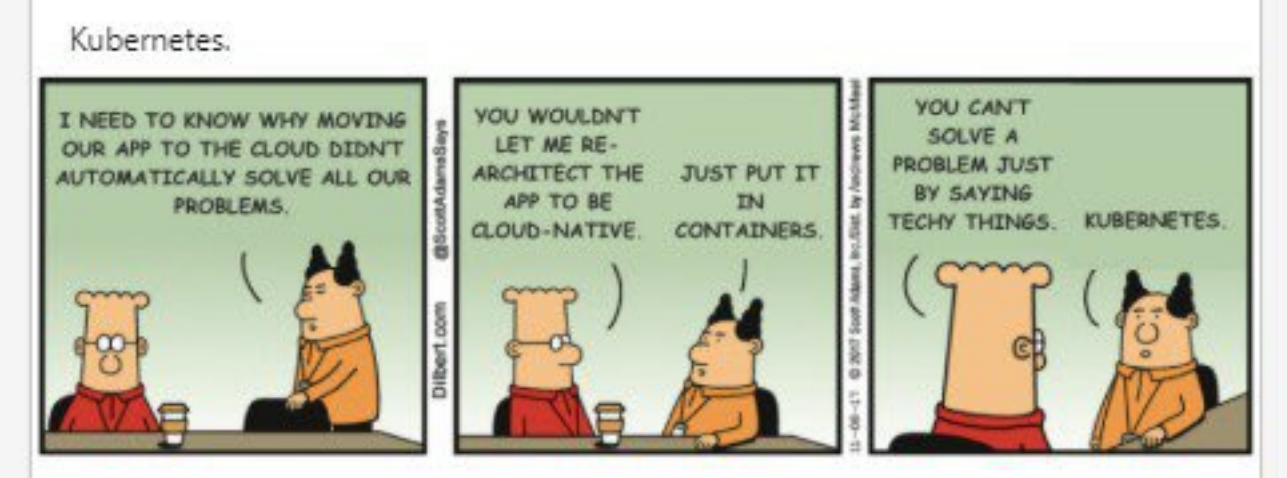
2. **Logging** - CRON doesn't log, unless you tell it too.

3. **Smallest resolution is 1 minute** - If a task needs to run every 30 seconds (or based on event sourcing), you can't do it with CRON.

4. **Cron pulls you out of the application** - CRON is a system level process. Not an application process.

Cron limitations cont'd

But it's OK… we run CRON on Kubernetes…



There are literally blocking reasons why you may NOT wish to use CRON on K8s –

https://kubernetes.io/docs/concepts/workloads/controllers/cron-jobs/#cron-job-limitations

Research motivation

1. Understand past and current practices related to task automation and workflow management
   - Why? To identify areas for improvement

2. Gauge whether (from within ITSD) there is a misunderstanding between task automation (CRON) and workflow management systems
   - Why? Because task automation is only part of digital transformation. Task management (and subsequently workflow management) are the next steps.

Part 2: Case study

How we were able to understand what needed to be done

… well we decided to put together a survey. Let's begin with some lessons learned from that experience.

1. Establish simple goals for the research study.
2. Always ask people what they have done, not what they would like to do.
3. State that the results will be anonymous.
4. The entire effort evolved from being a ‘questionnaire’ to being a ‘survey’.
5. A simple survey pitch should span no more than a few sentences.
6. Keep any question scales simple e.g., Love it, note sure, hate it.
7. Keep the questionnaire introduction simple.

ITSD | Info & Engineering Technology Planning & Development (172) JPL

To: JPL.Org.1722
From:172 Management

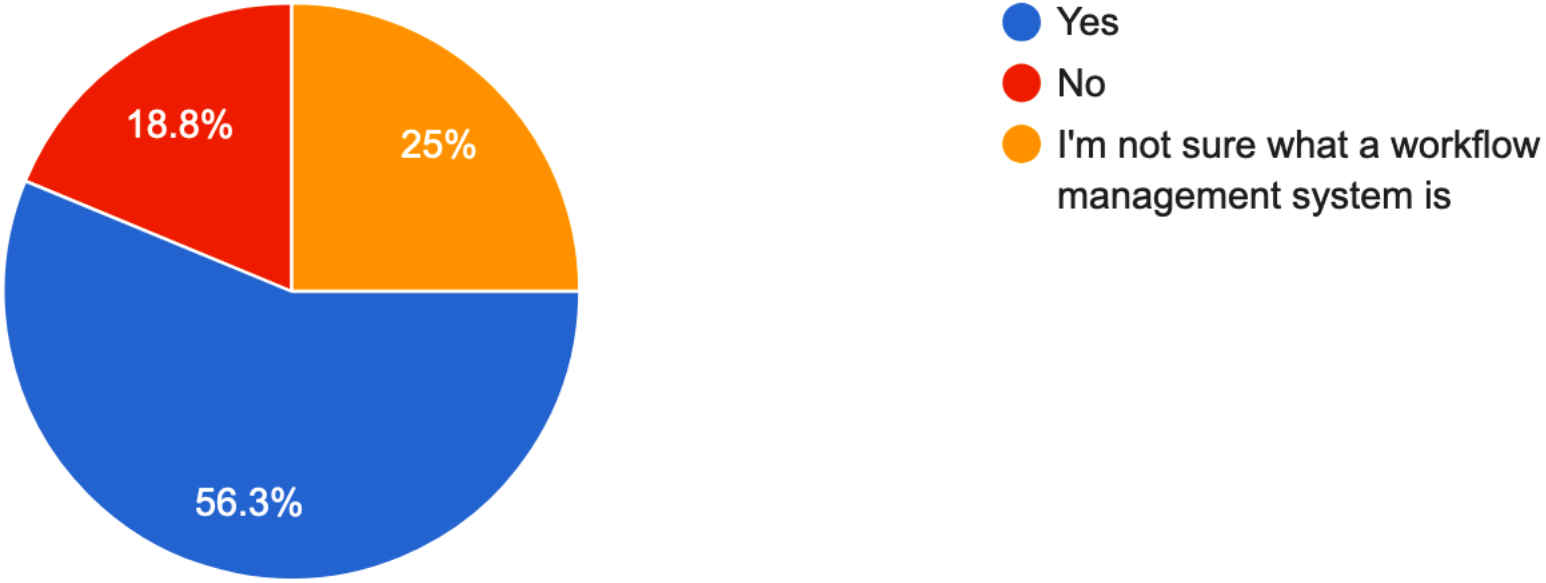
Thank you for your continued efforts in supporting ITSD. We ask that you please click the link below to complete an 11 question **Section 1722 Workflow Management Survey**, which should take no longer than 5–10-minutes. Your answers are anonymous and will be instrumental in understanding past and current practices related to task automation and workflow management, and will help determine next steps in supporting productivity within the section. We are grateful for your time and feedback.

1722 Workflow Management Survey: http://goto/workflow_questionnaire

Please Note: The survey is created in JPL Google Forms which is accessible to all JPL personnel. Should you encounter a screen directing you to request access, either sign out of your personal Gmail account and try again, or click on the button that shows your Gmail address to switch to your jpl.caltech.edu address, thereby logging you in via SSO.

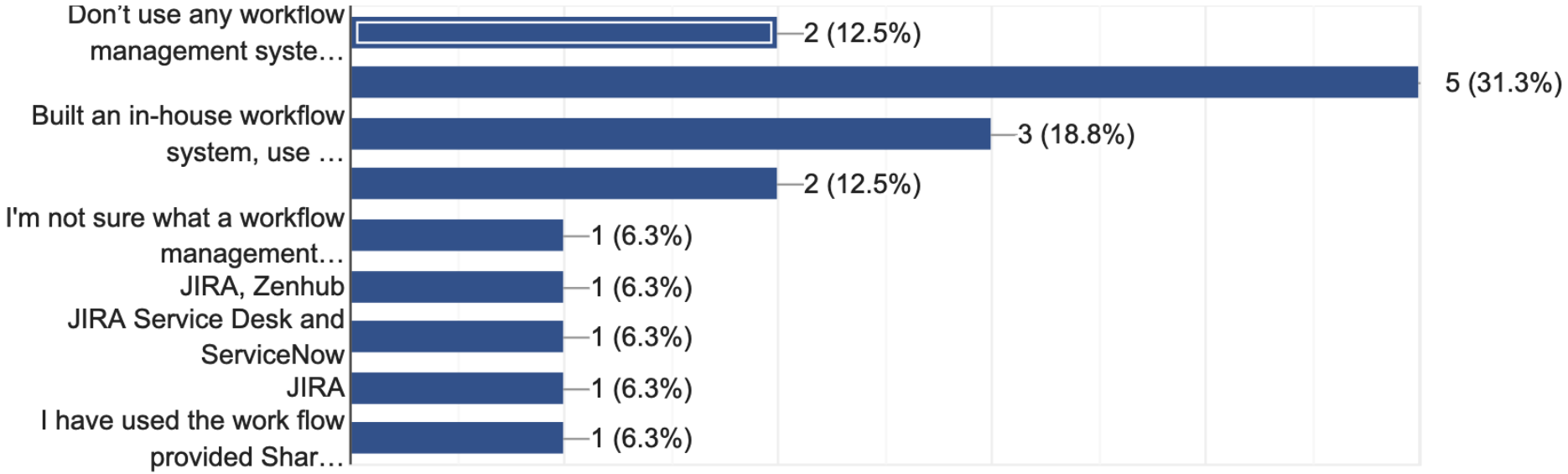
ITSD Information and Technology Solutions Directorate

Q: Have you ever used a workflow management system?


18.8%
25%
56.3%
Yes
No
I'm not sure what a workflow management system is

Q: Please indicate your current approach to workflow management. If using a Commercial Off The Shelf (COTS) solution(s) then please use 'Other' and provide

- **a) which one(s) and**
- **b) what your satisfaction criteria is.**


Don’t use any workflow management syste…
2 (12.5%)
5 (31.3%)
Built an in-house workflow system, use …
3 (18.8%)
2 (12.5%)
I'm not sure what a workflow management…
1 (6.3%)
JIRA, Zenhub
1 (6.3%)
JIRA Service Desk and ServiceNow
1 (6.3%)
JIRA
1 (6.3%)
I have used the work flow provided Shar…
1 (6.3%)

Q: Do you use CRON to schedule and automate job execution? If so, how does CRON perform for you? What issues have you had or has it been great? Please explain.

I use cron to automate tasks that need to happen periodically. This can something as simple as sending emails to auditing or updating records in a database.

CRON works well.

We keep to CRON's strong suits of independent non-dependency based executions and it performs as we expect. I dont expect it to operate like a Workflow mgmt system.

Yes, Kubernetes cron and AWS Lambdas are more robust and reliable than standard Linux crond. For our current application we would like to standardize to Kubernetes cron scheduler.

Yes, we use it for triggering emails and database syncing. It works well.

I made use of CRON extensively and it is doing very well for the tasks I need.

Q: What would a developer-focused workflow management service need to get developers to use it?

Not to be snarky, but a solid definition of what workflow management software is would be a good start. As a developer, I use project management software like ZenHub and JIRA but I'm not sure if those are classified as workflow management software or not.

walk through

easy api, support for easy to complex scenarios, flexiability, integrations w/ rad, ldap, sso.

API driven for access/submission/status..etc. Also the ability to integrate into existing monitoring/alerting systems.

flexible enough to handle serial, parallel, or combination workflows; ability to add hooks.

Automation, resilience, and reliability.

Easy to use UI form for data entry, a dashboard to track progress of all team work

It would need to be excellent at supporting individual developers working in relative isolation. At JPL we do a poor job of using development teams to build products, so team-based collaboration workflow management is not useful to most. A service that could help improve individual productivity would have the highest benefit.

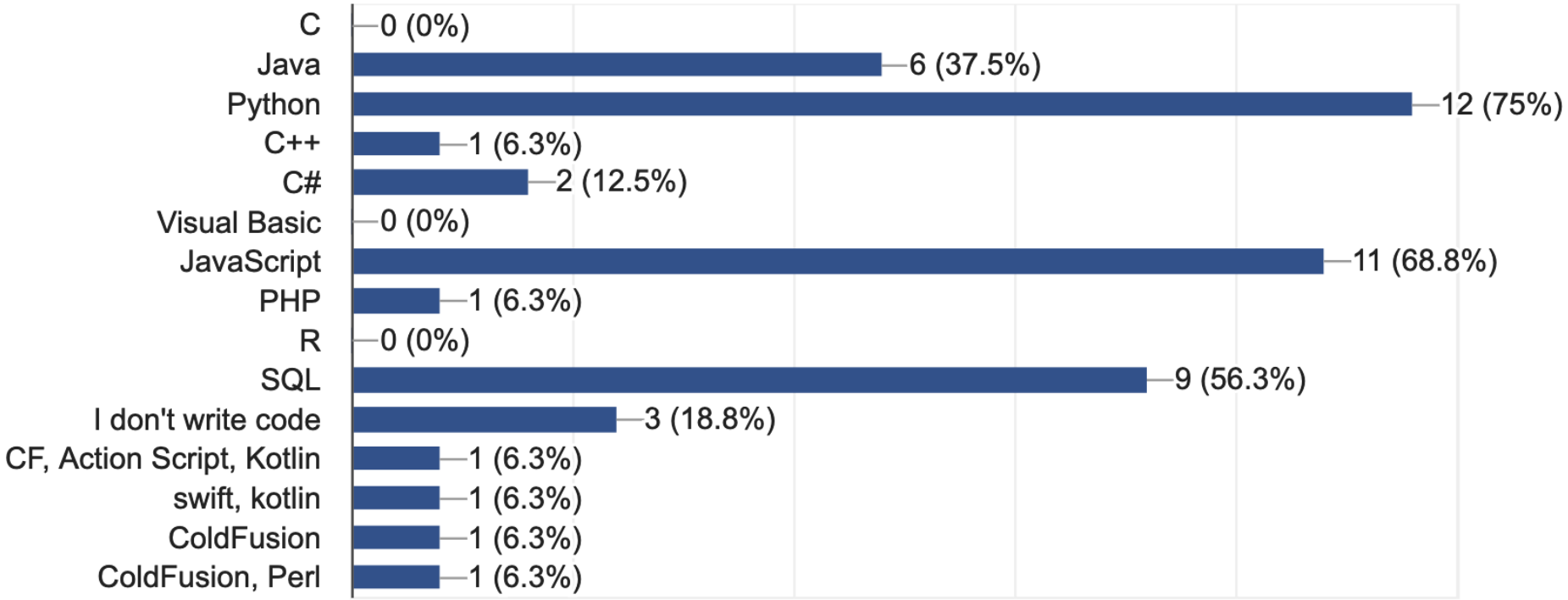
complete documentation, fast support response, easy to use, flexible, secure API design.

Good Documentation and socialization on benefits and real-world examples.

It must be easy to integrate with the application that is being developed.

Being intuitive and not extra work

Q: Which programming languages do you write your code in? It is important to know which languages you would like to use to author workflows. If you use more than one language please provide details.


C
0 (0%)
Java
6 (37.5%)
Python
12 (75%)
C++
1 (6.3%)
C#
2 (12.5%)
Visual Basic
0 (0%)
JavaScript
11 (68.8%)
PHP
1 (6.3%)
R
0 (0%)
SQL
9 (56.3%)
I don't write code
3 (18.8%)
CF, Action Script, Kotlin
1 (6.3%)
swift, kotlin
1 (6.3%)
ColdFusion
1 (6.3%)
ColdFusion, Perl
1 (6.3%)

Part 3: Evolving our (ongoing) workflow management strategy:

Architectural and design principles for workflow management systems

- **Dependency management** – resilient and flexible handling of upstream/downstream dependencies.
- **Historical reprocessing** - easy to reprocess by date, or re-run for specific intervals
- **Flexible, standardized, intuitive programming model** - tasks can pass parameters to other tasks downstream e.g. auth credentials
- **Error handling and failure recovery** - handle errors and failure gracefully and automatically retry when a task fails
- **Ease of deployment –** continuous integration tooling for automation
- **Rich 3rd party integration ecosystem** - hooks and operators for other enterprise systems
- **Logging and Interaction** - accessibility of log files and other metadata through the Web GUI
- **Real-time monitoring** - for all tasks' status in real time and send alerts to operations teams

Thank you to my colleagues

Specifically, in alphabetical order…

Aditi Shankar, Bill Seixias, Catherine Stringer, Christopher Berg, Cindy Trinh, Emily Tjaden, Eric Chiu, Ivonne Gonzales, Jeffrey Ma, Jennifer Yang, Jonathan Young, Manson Yew, Michael Milano, Paul Lumsdaine, Randy Moss, Roderick Enriquez, Todd Stoudnor and Yukio Sawada

Also, all of my colleagues in Section 1722 that populated the questionnaire and have shown an interest in this work.

Lewis John McGibbney Ph.D.
Enterprise Search Program Technologist
Web and Mobile Application Development Group (172B)
Application, Consulting, Development and Engineering Section (1722)
Information and Technology Solutions Directorate
O (818) 393-7402 | **M** (626) 487-3476
lewis.j.mcgibbney@jpl.nasa.gov
https://jpl.webex.com/meet/lmcgibbn

JPL | jpl.nasa.gov

ITSD	Future focused. Mission driven.	


JPL
Jet Propulsion Laboratory
California Institute of Technology