Meeting Minutes: MuleSoft (Private Sector)

Date: 07/17/2018

Time: 9:00 to 10:00am EDT

Attendance:

Nial Darby = Client Architect
Beju Ekperigin = Customer Success Director
Bob Craig = Director of Global Customer Success Architecture
Alexis McBride = Regional Director of Customer Success
Joe Cosentino = Senior Leadership/Business Partner, Sprezzatura
Shane Johnson = Management Analyst, Sprezzatura
Nitin Sahai = Enterprise Cloud Architect, Sprezzatura

Interview Agenda Items/Questions:

General:

- What is your role in your organization?
  - Mr. Nial Darby (Client Architect), and Mr. Bob Craig (Director of Global Customer Success Architecture). Primary feedback provided from these two interviewees most of the meeting.
- How many API's do you build each year? How many are for external consumption? How many for Internal consumption? What are the security concerns for internal versus external consumption? Firewall relaxation process and procedures.
  - N/A
- Do you utilize external industry publications such as Gartner?
  - N/A
- Website Link: https://www.mulesoft.com/

Prioritizing API's to Build:

- How do you prioritize the API's you're building? Do you use an Agile Backlog, or other method? Do you evaluate based on Level of Effort and ROI, or use other metrics? Do you prioritize APIs as to their eventual utilization across h enterprise versus a single project?
  - Question raised regarding how to reconcile with scaled Agile. Answer provided is embed and add a different perspective, speed (I.e. look outside for re-use).
  - Current API's - Consumer - Came after decision to build. Fundamental conviction = "Build it and they will come." Patient centered care, negotiating with consumer afterwards. We will see two things. Decision should be application based. Basic information for patient (reference re-use).

    - Cannot let teams go off on their own (I.e. be Agile on their own), then come in behind them, then do a review, then change direction; this goes against the current of everything.
- How much of your API backlog is defined by a Consumer's request? How much do you attempt to anticipate the needs of Consumers and build API's speculatively? Should speculative API development be more or less constrained by defined governance specifications?
    - Speed in mockout of APIs is critical to not frustrating consumers. Whoever is iterating with them should not be the developer of the process service. It should be someone who is interested in the consumer requirement (I.e. a specialized team).
- How do you make choices between building Experience layer API's vs Process layer vs System layer API's? Experience layer API's vs Process layer development could conceivably be performed by different development team – how will the governance model be enforced when there is a multi-team development environment?
    - Commingle publishing versus access. Separate the contract from the access (this eliminates re-use).
    - Cannot see the API definition; Specificity; too many layers, too much abstraction.
    - Mechanism (referenced here; more below?)
- Do you allow Experience layer requirements to drive build prioritization on Process and System layer APIs? If not, why not? Or is the option of the building of “mocked” lower level API implementations (Process and System)?
    - What is the mechanism that you use to make sure nothing overlaps with different clients?
        - Call failed during this answer (technical difficulties during the call/conversation).
    - Two areas that need to be digitized for this technology to work (unsure which ones; not noted here).
    - If you come in at the early stage over the top of the API saying something is wrong, you risk losing the momentum of development. You should encourage them to continue to build. However, if they are intentionally duplicating APIs rather than enhancing their own, then you must tell them to stop. If those APIs are not visible, then this is not the client’s fault (NOTE: Nial wasn’t saying this).
    - Speaking generically, HSBC (example company?) moved away from heavy handed governance, facilitation good principles in service design; not at low level rest. Moreover, what is the purpose of a microservice? Freedom to teams to execute. However, with technology, enforcing security policies across the enterprise. Have witnessed/observed governance teams get too heavy handed here; choice of mechanisms promoted in a perspective way.
    - (API) governance laying out the work for DevOps.
    - Various levels of continuous delivery (I.e. is automation fully in place, tooling for sign off [automation of this workflow])? Spirit of which is the question of if there is continuous delivery.

- If you have API's/services on legacy platforms, how do you decide when port those over to a new platform? Additionally, how these legacy applications will be "ported" – re-hosted, re-factored, re-built?
    - When do your clients usually port over their process API's over to a new platform?
        - Mr. Nial Darby is working with an agency that has an old platform that is too heavy weighted, and they would like to migrate over to MuleSoft. Their old platform is not allowing them to do what they would like to do in the tech world. Mr. Bob Craig said leave them until the business has no other choice but to take those process APIs off the legacy platform.
    - They (MuleSoft) have a team called "See free". This team is dedicated to education and training of API development and governance. They will hold the clients hand through the process in order to provide a smooth project creation and install. Change management is a huge focus at MuleSoft.
    - The banks (for example) have hundreds of teams developing onto the same platform. The C3 team spends a lot of time on education. Anyone who comes on(to MuleSoft?) gets educated on their platform. Project teams have autonomy to make decisions. Large organizations have the flexibility to promote that approach. Evangelizing the benefits of the new platform (I.e. showing them the resources that are already available , and that you would have to build if you go it alone; then focus on training). There needs to be an internal training mechanism, not off on your own. Do not assume that developers will go figure it out. Then keep driving the flywheel. Stay close to the team, and get them use to sharing APIs from day one (I.e. getting value, then contribute back). Building out automation that is necessary; deployment stuff is for free. Service catalog available; have to make an investment on how to coach them. It is change management with developer. My value is re-using and creating what is not there. There is a concerted effort that is needed there. Referenced banks, insurance companies (etc).

Standards to Which to Build

- Do you have multiple, unrelated teams, delivering API's into the same environment? If so, what lessons have you learned about configuration management across the environment/teams? What level of maturity are your Agile, CI/CD, Dev/Ops capabilities?
    - They have multiple teams.
        - The lab team; governance team owns this. Ensure every API goes into there and in-line with a sense of quality - it only gets published at a level of quality.
- Does your organization maintain formal standards for: API Contracts, Naming Conventions; Version Control, Branching, and Merging; Testing Requirements; Exception Handling; Logging; Security? Would you be able to share any documentation? How is this documentation presented – web pages, WIKIs, Playbook SharePoint sites etc.?
    - Agency outside VA; platform abandon; too heavy weight. Speed is the thing, migrate to a lightweight. The primary reason is to invest the money; We are on XYZ, license fee. Short of ROI, recommendation is to leave, if not changing. Whether the business process

    is changing, incremental cost to move, then do it. Do it because business had needs to be addressed. Business has a need to limit; facading to make it easier to move.

- Do you experience a need to keep documentation and configuration control minimal/light? How do you ensure you're minimizing paperwork/bureaucracy and maximizing code delivery? Again Agile, CI/CD, DevOps capability levels.
    - Not addressed specifically with these questions.
- How do you enforce standards across multiple development teams?
    - Not addressed specifically with this question.

Consumer Experience

- How many customers external to your organization consume your APIs?
    - How do they deal with Consumer frustrations? Mr. Nial Darby said they show them proof on how they test and build their APIs before implementation to reduce friction during conversations. Mr. Bob Craig said there is only friction and frustration when the provider does not ask how the client wants their API and simply tells them, “no you cannot do that, your getting this.” They then try their best to bend and mold their API and API governance to what they want out of their technology.
- What have you learned about development, testing, deployment that you apply to maximizing consumer experience?
    - Not all APIs are fit for consumption from the outside yet.
    - The governance team should establish a negotiation between VA and a new consumer.
    - Experienced API should be tailored to exact needs of consumer. Related to this, some data may need to be filtered. Security blend, that needs work too. Facade, gateway.
    - Frustration on the consumer side, and push back from process team. Customer Success Engineers are focused on enabling consumers. Niche for people who are good for working with consumers, then figuring out how to adapt.
        - Do these engineers build? No, coaching and guiding.
    - Consumer designs = Who builds MuleSoft app that consumes? Adapting existing APIs … That is the experience API. Experience consumes APIs (lightweight effort). Who does this? The consumer? Is this a Junior Developer dedicated to this? This has to be identified.

Closing Questions

- Are there any internal documents (not published on your webpage) that you could share that would show how you apply governance internally?
    - How much governance do you place on your large enterprise customers?
        - Not allowing them to simply go off on their own during development helps to stop error. Focus on having the governance team there looking at an API blueprint before anything is worked on is the best way to move forward. The worst practice is letting them develop the API and then coming behind them and telling them to turn back and fix it.
- To whom do you look to for API governance best practices, with who else should we speak?

    - What are some of the best practices to make these APIs easy to consume?
        - MuleSoft creates a lab by the governance team that they can run, test and tailor the APIs to the client's business needs before deploying anything. They bring the two teams together and show them an experienced API as an example to build towards what they want.
- May we please circle back with any follow-up questions?
    - Yes. The individuals interviewed today welcome follow-on dialogue and conversations.

Closing Comments/Questions [Additional Notes]

- Who else are you consulting with right now that are similar to the VA system?
    - HSPC bank is similar. They have other examples, but are not liberty to discuss them. They promote good principles in service design across the board to keep things simple and easy. Another concept is enforcing security policies across the board, as well. Their choice of mechanism and service design work side-by-side to provide security and continuous delivery. If there was any automation that allows self-service, it would make development and installation a much smoother process.
- What information do you have on what the VA is building first for APIs?
    - The mentality was "build it and they will come." The decision of which APIs to build will be between the stakeholders of the VA "vets, VA employees, contractors." Another initiative would build for the internal and external user.
- If their are secure APIs and non-secure APIs, then there should be two contracts separating the two to create a wall between.
- Also, promotion of DevOps/promotion of self-service is relevant to VA.
- Should not dump into pipeline; DevOps produce artifacts to make deployment easy.
- Automation of various environments especially in cloud.
- Front-end for the team; easy to develop, hard to deploy.
- Fill out form for lower environment, then automated process for establishing the environment.
- Federating the effort (what does this reference/mean?).
- Architectural experience required; microservices have to be able to be reused. SOA was overly ambitious (I.e. too much time in canonical model; very hard to get consensus around the model).
- This will allow for a business entity (Veteran) demographics example:
    - VHA versus VBA way (maybe build to services); Double checking needs on consumer side (I.e. same set; stop if not fair enough). There can be multiple services. Two things that can be digitized; queries and execution of process. Different business entity's services could be reused
- Cannot have a lack of appreciation for the rest of the organization (could potentially happen).
- Governance team has to marry this together (I.e. service versus organization).
- If you come in at the early stages, and if you come in over the top, you risk losing momentum to move towards encouraging them to build the APIs. If it is a duplication, then go over the top; add to the (overall) API ecosystem and publishing (it all has to be visible).

    - SOA governance (focused on) at the beginning takes momentum away.
    - Come in at beginning afterwards, slows momentum.
- IRS (example): Mr. Bob Craig response = Critical and sensitive API publishing.
    - Commingle publishing versus access. Separate the contract from the access; this eliminates re-use (cannot see the API definition).

Additional Reference Information and Analysis

Microtask Requirement

As the VA Lighthouse (now API Platform) Product Owner seeking the appropriate Governance model, I would like to understand, with the intention to adopt, best practices from the private and public sector, specifically for prioritizing APIs to build, standards to which to build APIs, and making the APIs usable by external consumers. We would like the primary research performed to gather best practices around:

- Characteristics of effective models in the public and private sector, and who is successfully using them. Including the utilization of leading edge API development enabling technologies (CI/CD, Dev/OPS, Micro service, Containers etc.).
- Lessons learned from these organizations (both what is working and what isn't)
- Highlight strengths and weaknesses of selected governance models.

____

Interview Analysis

Here is analysis from the MuleSoft call.

Enterprise Architecture Engagement Focused on Canonical SOA Model Kill's Momentum - In its consulting engagements, MuleSoft observed that a heavy governance process that centrally manages what API's get built and how they fit into a canonical SOA model, kills developer team momentum.

> Analysis: MuleSoft recommends embedding enterprise level architects in individual development teams as collaborators. The goal is not to conduct a formal check of what API's get built at the enterprise level. Breaks should only be put on development teams when there is an obvious duplication of effort.

Federated Delivery Model Requires Education and Support of Developers using CI/CD Pipeline - MuleSoft recommended heavier governance in the area of CI/CD Pipelines support for development teams. They cited the adage that "it is easier to build then deploy." It is inefficient to have each development team built and manage its own pipeline. That can be done centrally. There is then a requirement for a *education* and documentation to support the development team's use of the Pipeline.

> Analysis: While management of a central CI/CD deployment pipeline is a governance issue, it may be better for VA to plan for resourcing such an effort first, and then make this a separate acquisition from either PMO services or developer services. This is a different model than currently pursued in VA acquisitions.
>
> Either in a CI/CD "as a service" acquisition, or in a governance support acquisition, we recommend including a developer education and documentation component beyond standard VIP documentation requirements.

Designate a Customer Success Engineer - MuleSoft discussed the importance of having a dedicated "Customer Success Engineer." This person would be responsible for direct liaison with Consumer API Developers when VA develops a new API for them and the associated documentation. This person would be on a VA team, but maintain the perspective of an outside developer consuming the VA API, focusing ease of use of the interface.

> Analysis: We recommend VA acquire services for such a roll. This would need to be placed as an explicit requirement in either a PMO contract or a API development contract. There should be governance over the full interaction with Consumer Developers (design, authorization, testing, etc) and such a contracted resource would need to have explicit responsibilities in each step of the process. The emphasis should be on minimizing effort by the third party developer and simplifying their experience.