

Table of contents

Potential Risks and Problems

Introducing SIAM

The Scenario

Step 1: Identify the Business Pain Points

- The Scenario

Step 2: Uncover the Business Requirements

- The Scenario

Step 3: Involve the Right People

- The Scenario

Step 4: Design and Engineer System Requirements

- The Scenario
- User Stories
- Use Cases
- Data Management
- Data Security
- Managing Different Versions of Systems



- Customizing the Core Setup
- Workflow Design/ Automation/ Orchestration
- QA/ QC Process
- Rollback/ Deployment Plan
- End-to-End Service Levels
- Knowledge Base
- Training

Step 5: Know the Right Solution for Your Customer

Step 6: Implement Iteratively

- The Scenario

Step 7: Deploy to Production

- The Scenario

Step 8: Operation and Support

- The Scenario

Conclusion



IT professionals such as Solutions Architects, Project Managers, DevOps Engineers, Sys Admins, Service Delivery Managers, and the like, all play an important role in every organization. They have their own contribution to a software project delivery lifecycle, right from understanding customer requirements to delivery of services to operations and support.

Though they have distinct responsibilities, they all need to work in tandem for the success of the end-to-end delivery of IT projects.

In this blog post, these different roles will be addressed as IT professionals, recognizing the fact that their expertise is highly valuable and relevant for IT projects. So, here's how we acknowledge the current situation of these IT professionals.

The organizations they are a part of, often employ work management systems or issue tracking systems to manage their day-to-day operations effectively. However, to fulfill varying requirements, such organizations often use best-of-breed systems from separate vendors.

For instance, Project Managers, who handle the project and product deliveries, might use Jira for their tasks, while Solutions Architects, who oversee the design and architecture of technical solutions might use ServiceNow for managing their tasks. As a result, negative symptoms of working in an environment where internal teams use disparate applications within the company become visible.

These symptoms are further manifested in a multi-sourced environment where customers, suppliers, and partners of such organizations have their own applications and information needs to be exchanged between them.

This is because, when these disjunct organizations and teams try to exchange information through such applications, the lack of a standardized interface between them creates problems. Problems like inadvertent alteration of data, lost or irrelevant information, resulting in delayed responses further impacting customers and business. The only way to mitigate this is by integrating these applications/ toolsets so teams can seamlessly collaborate and communicate with each other.

Welcome to this ultimate guide that rolls out a cross-company integration approach. It employs SIAM and ITIL 4 best practices, and its goal is to illustrate steps IT professionals take to develop this capability within their organization.

Note: If you prefer an ebook, then go ahead and download [this free guide](#) along with its worksheets.

Here's what we will cover in this comprehensive guide:

- [Potential Problems and Risks](#)
- [Introducing SIAM](#)
- [The Scenario](#)
- [Step 1: Identify the Business Pain Points](#)
- [Step 2: Uncover the Business Requirements](#)
- [Step 3: Involve the Right People](#)
- [Step 4: Design and Engineer System Requirements](#)
- [Step 5: Know the Right Solution for your Customer](#)
- [Step 6: Implement Iteratively](#)
- [Step 7: Deploy to Production](#)
- [Step 8: Operation and Support](#)

However, embarking on a cross-company integration project is not always a walk in the park. There are quite a few potential risks and problems that we need to take into account.

Potential Risks and Problems

So, what are the potential problems and risks you might encounter when starting a cross-company integration project?

1. If data mapping is not consistent and comprehensive, it can lead to potential bottlenecks. Cross-references of this data must be accurate between systems. Otherwise, it leads to incorrect or delayed information exchange (For instance: If the same task is called an issue in one system and an incident/ ticket in another, then it must be acknowledged at the onset that both mean the same).
2. Dependencies on APIs of 2 different companies often introduce a lag in the implementation cycle. (For example: If there is a problem with an API of one system, then the integration cannot proceed until it's resolved). This creates a dependency between the companies involved in the integration process and delays it.
3. Comprehensive testing needs to take place. If this testing is not adequate, it can cause issues in post-production. End-to-end scenario testing must be defined and executed. This will simplify the maintenance in the future whenever a change needs to be validated. The lack of such scenarios will lead to failing integrations and will reduce its usability.
4. People often resist digital transformation. This can be a potential risk if different teams in a project need to collaborate and they are not ready to change their mindset. Following are 2 instances of this point.
 - a. People lack the skills/ know-how to use and benefit from new technologies. In traditional companies, people do not understand the “big picture” and how new technologies might improve businesses and processes.
 - b. Integrating one system with another could lead to risks of unauthorized disclosure or malicious tampering of data being exchanged. So, questions on integrity and confidentiality are of the utmost concern.

So how do we move forward taking into account these challenges?

Here, we consider SIAM and ITIL 4 best practices to help mitigate the above problems and generate a well-structured cross-company integration project driven by IT professionals.

Introducing SIAM

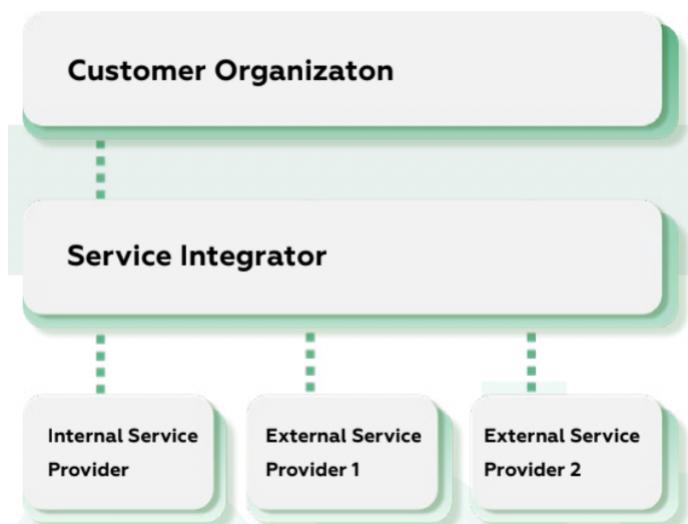
Let's understand why we chose SIAM and ITIL 4 best practices for our cross-company integration effort. Why are we using SIAM as a scaffold for building across-company integration?

A multi-sourced environment has a lot of challenges when it comes to managing interactions and collaborations between service providers. Here SIAM (Service Integration and Management) introduces the concept of a “service integrator” working with providers alongside customers to achieve the common goal of service improvement. Such a service integrator is a single, logical entity held accountable for the end-to-end delivery of services and business value that the customer receives.

The SIAM ecosystem has different layers: the customer, the service integrator, and the service providers.

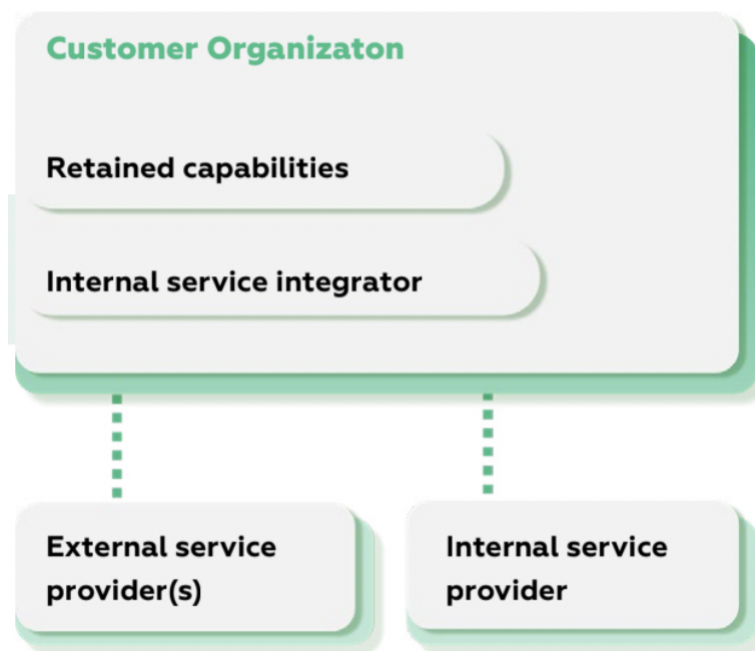
These 3 roles include:

1. **The customer(s):** The customer(s) to whom the services are being provided.
2. **The service providers:** The suppliers/ vendors/ partners that provide the service.
3. **The service integrator:** A strong competent voice to make sure the customer and the providers are on the same page.



Also, SIAM supports the idea of an external and internal service integrator. An "external service integrator" entails a third-party organization/team playing the role of an integrator towards meeting end-to-end service levels, whereas in an "internal service integrator" framework the customer organization plays the integrator role providing integration capabilities. Care must be taken to manage the service integrator roles and customer roles separately to avoid confusion over responsibilities.

A typical internal service integrator ecosystem in SIAM looks like this.



Why do we use ITIL 4 in cross-company integration?

Many organizations are already following ITIL v3, and have realized the benefits it brings to service delivery and management. ITIL 4 introduces the concept of "Service Value Systems" where it focuses on co-creating value for customers in a business ecosystem and proposes that IT service management and business requirements are always aligned.

Let us also understand how ITIL and SIAM relate to each other here.

ITIL 4 in a SIAM Ecosystem:

SIAM does not replace ITIL. It builds on top of it and extends it across the multi-sourced ecosystem. SIAM adapts to ITIL 4 concepts such as Service Value Systems, practices, and techniques to effectively work in a multi-service provider environment.

This helps establish that SIAM and ITIL can go hand in hand.

So what's your role in all this?

Every organization has its own drivers while choosing to play the part of a service integrator. And as IT Professionals, you can proactively showcase the benefits of such an opportunity to generate cross-company integration capability within your organization using SIAM and ITIL best practices. It can help structure and streamline your efforts to integrate disparate systems by establishing the concept of an internal service integrator (essentially your organization) with potential or existing customers.

You can learn more about SIAM in [this blog post](#). Also interested in ITIL 4 and service management? Then check out [this article](#).

Note: Throughout this guide, adaptation to **ITIL 4** and **SIAM** practices have been distinctly mentioned.

The Scenario

This case is used in the article to illustrate the concepts that are presented. A company, by the name of Infinity Solutions, is confronted with challenges in managing multiple projects and collaborating with multiple teams, all using different work management systems.

Infinity Solutions specializes in the sales and distribution of clothing and has decided to digitalize its business. Now the users can order via their website, make payments for their purchases, and browse the collection on the online kiosk.

John, the Solutions Architect working with Infinity oversees the technical and architectural requirements of software solutions to align them with business objectives. He uses Jira for handling his tasks.





Elaine, the Project Manager with Infinity just got assigned its e-commerce project. She uses ServiceNow for managing her tasks.

Now, John and Elaine need to work together on this new project for which they need both their teams to collaborate and work with each other.

Also, Infinity decided to outsource its customer service to **Mercuri Inc**. It uses Zendesk to handle all incoming requests.

Infinity did not have the required resources/ expertise for the development and outsourced it to **Global Pvt Ltd**.

The team at Global has David working as the DevOps engineer to bridge the gap between development and operations. Marco is a Sys Admin here to manage and support the IT Infrastructure. David uses Azure DevOps.

Name Of The Company	Job Role	Responsibility	Management Systems
INFINITY SOLUTIONS Specializes in the sales and distribution of clothing and has decided to digitalize its business. Now the users can order via their website, make payments for their purchases, and browse the collection on the online kiosk.	 John Solutions Architect	Oversees the technical and architectural requirements of software solutions.	Jira
	 Elaine Project Manager	For keeping track of project deadlines, managing risk and ensuring day-to-day operations run smoothly.	ServiceNow
GLOBAL PVT LTD Infinity has outsourced the software development to Global.	 David DevOps Engineer	Bridging the gap between development and operations.	Azure DevOps
	 Marco Sys Admin	Manages and supporting the IT infrastructure.	
MERCURI INC Infinity has outsourced customer service to Mercuri.		Handling system.	Zendesk

Infinity faces the following challenges for collaborating across its multi-sourced ecosystem:

- Since both Elaine and John use different applications, the same issues handled by both teams cannot always remain consistent. This is because the issue-related information has to be passed back and forth manually between their teams. This results in delays and inconsistencies in resolving issues and has increased friction between them.
- They had to go through the pain of reaching out to Mercuri via email or phone calls if they wanted to access their tickets' status in Zendesk. They could not keep any sort of archives either.
- Whenever a development issue, network issue, or a change request comes in, it needs to be handed over to Global for resolution, but since Global already uses AzureDevOps, Infinity cannot keep track of the issue in their Jira/ ServiceNow while Global is working on it. Also, David needs to assign network/ connectivity-related issues to Marco through emails. Marco, after working on them manually, sends the status back to David, who then updates it in his Azure DevOps.

Evidently, every team in the above scenario is working in siloes and several problems have started to crop up when they try to communicate with each other. This has severely impacted Infinity's incident resolution times and the customers are disgruntled even more. To resolve this, there arose a need for a cross-company integration solution that would help Infinity collaborate seamlessly within and outside its borders.

As a result, ITIL 4 and SIAM methodology were deliberately put together which enabled us to come up with a well-thought-out cross-company integration guide. Here are the steps Infinity carried out that co-created business value with its customers at the end of this journey.

Step 1: Identify the Business Pain Points

As IT Professionals, you represent highly qualified leadership roles. You are not only experts in analyzing business requirements, designing and implementing them, but also supplying IT solutions end-to-end. So, what do you do when your internal or external teams are confronted with challenges introduced by a multi-sourced environment? You acknowledge them, step up and take a lead on

pressing issues like SLAs being missed, lacking accurate information, etc. because you have experienced them first hand.

A cross-company integration process can be a strategic decision here. By involving senior management, it will be possible to take a step back from the day-to-day business or operational views and think about the future such integration can bring to the organization.

Adopting such a solution involves critical decision-making. The presence of a good strategy driven by the strategy team provides a clear vision of the future, confirms the purpose and values for your customers, defines objectives, clarifies threats and opportunities, determines methods to leverage strengths, and mitigates weaknesses (at a minimum).

Following is a comprehensive list of pain points that you can appreciate:



Productivity Pain Points

- Limited visibility on critical data and important points
- Long user complaint handling times
- Scattered and disconnected information across different tools
- Copy and paste when navigating between systems
- No real-monitoring and tracking of tickets and tasks



Customer Pain Points

- Limited information and capabilities available on the customer portals
- Slow repair time / turnaround time
- Incident reported not showing up
- Fault handling is purely reactive i.e. through customer complaint calls or user complaints



Financial Pain Points

- Limited visibility to know if good financial decisions are made
- Overpaying equipment and tools, but don't know where to cut costs



Process Pain Points

- Inconsistencies in each employee's workflow, which leads to disorganization and varying performance
- Customer churn is high because the service department is inundated and can't keep up.

Gathering the above pain points within your organization is easy, but how can you accomplish it with your common customers, suppliers, or vendors?

This can be achieved through a face-to-face interview, customer feedback, project feedback, performance reporting, business level feedback, input to new products, incident reviews, and complaints.

Working on getting such feedback helps tackle tougher questions that can be asked henceforth. Continual improvement can be proposed and even service or system integration proposals can be explored.

The primary purpose of this step is to create a "to-be" vision. This will help with uncovering the business pain points gathered here and prioritize improvement suggestions in the next step.

The Scenario

The "To-Be" Vision: A frictionless and flexible collaboration across internal teams and outside company borders.

Infinity Solutions envisioned an environment where the systems are integrated in such a way that whenever a customer complains, it can be resolved by escalating a ticket directly from Mercuri towards it while keeping track of this complaint in both ServiceNow and Jira used by the IT Professionals, John, and Elaine.

Additionally, it also envisioned an environment where development or a network/ connectivity issue is directly forwarded to Global to be handled by the team under David (and Marco, if needed) using AzureDevops, keeping Infinity's Jira/ ServiceNow in the loop.

The increased efficiency has a direct impact on resolution times and customer satisfaction. If this works, the approach can then be applied to all their suppliers.

Note: *Plan and Engage from ITIL v4 - Discovery and Strategy in SIAM*

Step 2: Uncover the Business Requirements

In this step, business requirements are uncovered and described. Ideally, every requirement is formulated in a way that the integration can be validated. Typical questions that can help uncover these requirements are:

- What are the business cases to cover?
- What are the current types of information being exchanged?
- Should the data that is exchanged be visible to all the necessary stakeholders and whether it's useful to them?
- What is the critical data, its accuracy, and who should have access to it?
- What are the current workflows, how long does it take to process a typical event, and what are the current potential bottlenecks?
- To check if any of the problems identified are affecting SLAs in resolving tasks.

The Scenario

For Infinity identifying the pain points will mean the following:

Currently, when a ticket is raised in Mercuri (using Zendesk), the customer service representatives manually send an email or place a phone call to Infinity. They could even go so far as to copy and paste the relevant ticket information in a template and send it across.

The project manager at Infinity, Elaine, receives the ticket and gathers the relevant/ additional information about it before sending it for initial investigation. After this, the ticket is passed on to the Solutions Architect, John for checking if the problem is architecture/ design-related or not.

If it is, then it needs to be worked on under John and Elaine, who use ServiceNow and Jira. And if the issue is development-related, it needs to be forwarded to Global (using Azure DevOps). All this is done **manually** or through a pre-decided medium.

Global upon receiving the ticket, manually updates its status in Azure DevOps. If the issue is development-related it is assigned to the appropriate team member or is passed on to Marco, the SysAdmin, if it's a connection or a network issue.

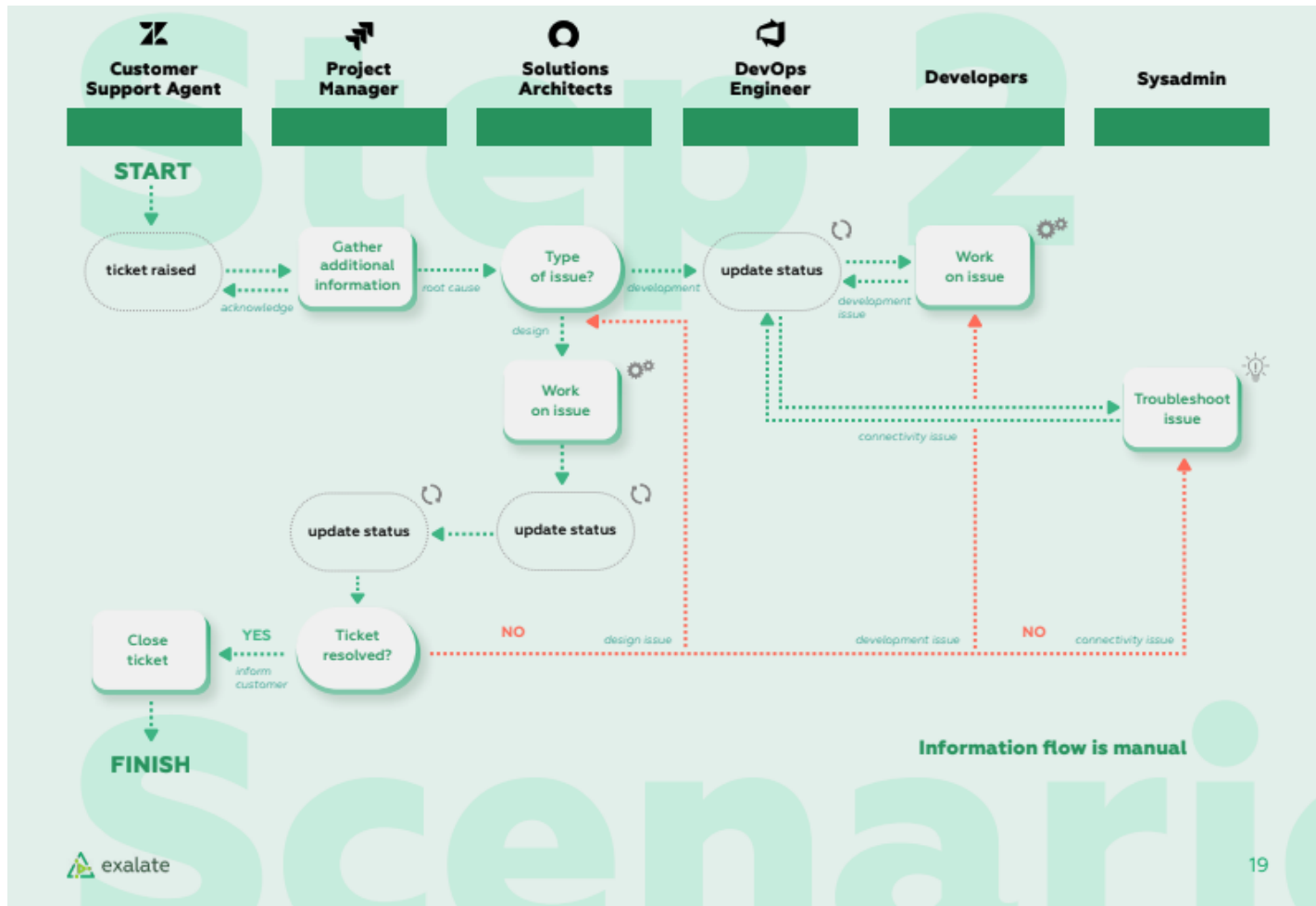
All this time, John and Elaine (in Infinity), using Jira and ServiceNow, manually update the ticket status on their applications by gathering feedback from Global through phone calls or emails. Such manual data entry mistakes are creating friction between their teams.

Meanwhile, Mercuri is in the dark about the current location and status of the ticket and is unaware of what needs to be communicated to their customers. Once the issue has been resolved and approved, it needs to follow the reverse process and reach Mercuri for the final resolution from the customer. This has brought down Infinity's SLA and has impacted its business hugely.

So based on the pain points gathered, we uncover business requirements by prioritizing them as follows:

- Infinity wants to have a holistic view of information (e.g tickets, issues, change requests) and must be able to track it end to end. This information entails data exchanged intra-company (within Infinity) or cross-company (outside its borders).
- It wants to ensure autonomy while collaborating with all its suppliers/ stakeholders or between different departments. It means that whenever it decides to integrate disparate systems to exchange information, it must be done in such a way that neither collaborating parties lose control of what information is sent out and how incoming information is processed. Keeping the systems loosely coupled is essential to ensure scalability and maintainability.
- It also wants to ensure consistent information across all the different work management systems so that each company and department remains on the same page. Hence, it needs to be synchronized in real-time, and automation (in terms of triggers) must be in place for it to be transferred.
- To enable all collaborating parties to securely exchange information - encryption methods and access control mechanisms need to be present to ensure that no user from Infinity can have access to critical/ insider data from Mercuri/ Global or within different departments.

Here, there is an obvious need for exchanging and synchronizing information between all the different work management systems used by these companies so that information can be tracked and monitored throughout its lifecycle.



Based on this analysis - key performance indicators and critical success factors can be defined:

- How to measure improvements?
- How to track and perform trend analysis?
- Monthly service reviews to see how different services are holding up.
- Customer feedback to see if there are any deviations on the service levels.
- What is the current baseline, against which system improvements can be defined?

The above points play a significant role in deciding a continual improvement framework for the collaborating companies opting to integrate their applications.

After the content and specifications of the system are initialized, it is the ideal moment to define how the different businesses will cooperate. The integrated solution will need maintenance as businesses evolve. Having the right governance organization in place to monitor the efficacy of the provided solution (along with the defined KPIs) and deciding on the prioritization of improvements is an essential part of ensuring a proper return on investment.

Once the scope for continual improvement is acknowledged and agreed upon, it's time to involve the right people in the integration process.

Note: *Continual improvement from ITIL 4 and Improve from SIAM*

Step 3: Involve the Right People

Most integration projects fail before they even start because they do not involve the right people in their team. For instance, they might not consider a data mapping expert which leads to inaccurate or wrong mappings affecting the data synchronization. So the aim is to have a cross-functional, cross-process, cross-department, and cross-provider integration team designed to co-create value in a multi-

sourced environment.

Specific roles and responsibilities across the entire framework are identified and assigned. They can include the following subject matter experts:

1 SERVICE INTEGRATOR
Held by IT professionals for ensuring the end-to-end delivery of integration services. The role can be divided amongst them based on expertise and resource availability.

2 SOLUTIONS ARCHITECTS/CONSULTANTS
Consultation on setting up the SLAs. Deciding on the response times, bandwidth, speed, latency (of cloud vs. on-premise), capacity expansion, and other architectural requirements, volumetrics like the number of issues created / how many hours saved.

3 PROJECT MANAGER
Keeping track of the progress of the integration project, to make sure deadlines are not missed and the day-to-day operations run smoothly.

4 DEVOPS ENGINEERS
Bridging the gap between the development and operations team.

5 SYSTEM OR APPLICATION ADMINISTRATORS
Installation and maintenance purposes. Also for network and connectivity issues. Good knowledge of the whole integrated system is very important.

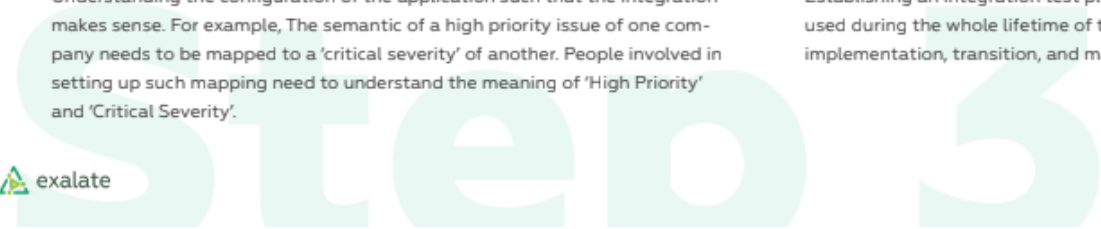
6 INFORMATION SECURITY MANAGERS
Ensuring security features like defining roles for access rights, authentication mechanisms, and the like.

7 SERVICE DELIVERY MANAGERS
Ensuring services are delivered on time and proper support is provided for them.

8 DEVELOPMENT/CONFIGURATION TEAM MEMBERS
Special configuration requirements (scripting language usage, OS restrictions, compatibility considerations), Configuration of volumetrics, version control.

9 DATA EXPERTS/ ANALYSTS
Understanding the configuration of the application such that the integration makes sense. For example, The semantic of a high priority issue of one company needs to be mapped to a 'critical severity' of another. People involved in setting up such mapping need to understand the meaning of 'High Priority' and 'Critical Severity'.

10 TEST ENGINEERS
Establishing an integration test plan which can be used during the whole lifetime of the project (i.e implementation, transition, and maintenance).



The Scenario

For the above roles, consider the following possible scenario:

1. IT Professionals working as service integrators. They work with Infinity/ Global as Solutions Architects, Project Managers, DevOps Engineers, Service Delivery Managers, and the like.
2. The SME (subject matter experts) mentioned in the table must be divided across the 3 companies based on the availability of expertise or service delivery capabilities.

The basic aim is to have people involved from all the 3 companies and departments together to reduce friction between them and improve decision-making capabilities.

To help map these roles and responsibilities in an organized manner, RACI charts can be used.

RACI chart (also known as responsibility assignment matrix, or RACI matrix) is used to map the tasks and deliverables to the roles assigned to team members. The obvious benefits are better decision-making capabilities and a proper understanding of the roles and responsibilities of each member of the team.

RACI stands for:

- Responsible - Person(s) responsible for doing work
- Accountable - Person accountable for signing off the work (max 1)
- Consulted - Person consulted before and during the task
- Informed - Person informed of work progress/ completion.

You can plan this RACI model well in advance so that when it comes to completing a task everyone already knows who is responsible. Following is a simple example of a RACI chart with clearly defined roles and responsibilities.

Project Deliverable	Project Leadership				Project Team Member						External
	Susan Project Sponsor	Sharon Project Manager	Mark Manager	Anil Manager	Pad Lead	Jen Dev	Sam Analyst	Frank Admin	Prince SME	Brian SME	Martha Application Consultant
Develop Baseline Project Management Plan	A	R	I	I	C	C	C	C	C	C	C
Define Business Requirements	I	I	I	I	-	-	R	-	C	C	A
Design Technical Solution	I	I	I	I	A	R	C	C	C	C	C
Develop Requirements Traceability Matrix	I	I	I	I	-	-	A	-	-	-	R
Integration Testing	I	I	I	I	A	R	C	I	I	I	C
Deploy Solution to unit test in stance	I	I	I	I	A	C	-	R	-	-	C

RACI helps in streamlining communication between all team members, avoiding work overloads and silos, and setting clear expectations.

In the RACI matrix for Infinity's integration project, the solutions architect, John, is the accountable person if anything goes wrong in the design of the architecture. So the people involved in the integration process must acknowledge and be informed about this at the onset to avoid situations like "Who to report to?", "Whom to go to?" etc. They know John will be the one handling problems related to designing. This will help set and identify clear escalation paths.

Monthly steering meetings can take place between these companies to keep track of the integration effort and help identify continual improvement imperatives. Weekly/ daily stand-up meetings can also be arranged to effectively handle and manage day-to-day operations.

So, the right blend of a team involving all stakeholders is necessary to move forward for designing and engineering customer and technical requirements.

Note: *Identifying and assigning roles and responsibilities in the SIAM environment*

Step 4: Design and Engineer System Requirements

Once the right kind of team is set up, it can be responsible for coming up with the detailed design of the integration.

In this step technical requirements and constraints need to have full attention. If the requirements are not designed end-to-end, it can lead to incorrect data mappings, wrong configurations, and the like. In turn, it will lead to customer dissatisfaction, and the benefits of such an integration solution will remain unrealized.

To help you with your conversation, here is a sample set of questions you can ask. They are divided into different categories:

Infrastructure

- What network layout is applicable between the different systems?
- What to do in case of upgrades?
- What are the APIs provided by these toolsets?

Information-flow

- What is the data that will be exchanged?
Specify a Data Dictionary outlining the mapped fields - with samples and clearly defined semantics.
For instance, "short-description" in ServiceNow is typically the "summary" in Jira.
- How is the data mapped to each other?
 - For instance, priorities in one instance are 'Highest', 'High', 'Normal', 'Low' while in the other instance 'Severity' is being used instead.

- What are the rules that affect the way that information can be exchanged?
 - For instance, in one system it is not possible to add new information to closed tickets.
- What are the workflows, and how should these orchestrate?
- What information is exchanged at what events?

Security

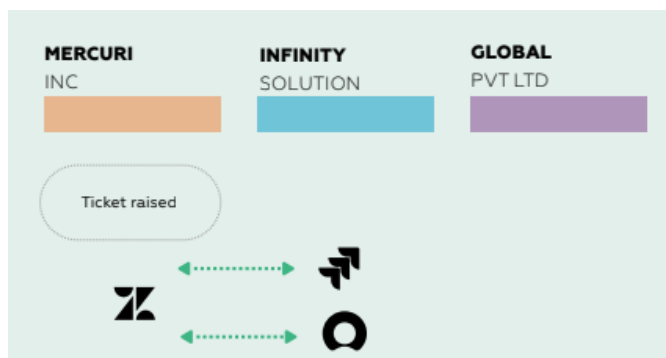
- What is the retry mechanism for resolving errors?
- Is there any way to recover from failures and resume synchronization from the point of interruption?

Deployment related

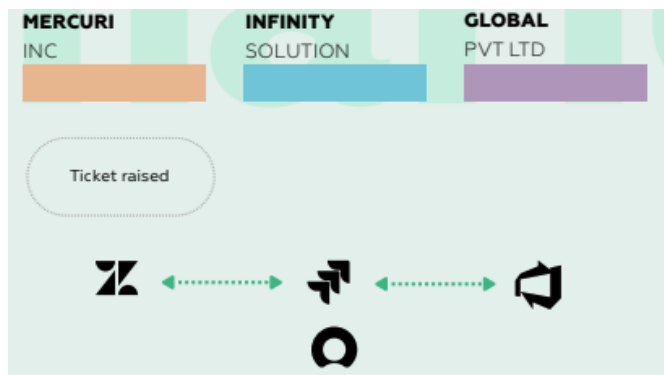
- What configuration management system will be used to ensure system traceability?

The Scenario

For effective workflow automation for Infinity, the following steps can be proposed:



1. Suppose Mercuri's Zendesk has received a ticket from the end-user. This will automatically create an issue under the appropriate project in Infinity's Jira used by the Project Manager Elaine.
2. Elaine then gathers the necessary information (what, where, and how) of the ticket in case it's not provided. She might also need additional inputs from the customer. All this back and forth can be accomplished by automatic bi-directional syncs between Jira and Zendesk in the form of comments/ attachments/ any other custom fields.
3. After Elaine finishes her task, it moves to John, the Solutions Architect for the root cause analysis of the ticket.
4. This means identifying whether it's a design/architecture issue or a change/ feature request affecting the design of the system, in which case John (using ServiceNow) handles it in collaboration with Elaine (using Jira). This collaboration between ServiceNow and Jira is achieved by bi-directional syncs between them. Once the design is worked on and finalized, the issue is then automatically updated under the correct status and assigned to Global's development team using Azure DevOps.
5. If the issue is a small bug or an incident, it needs to be handed over to the appropriate team member or if it's a connection/network issue it needs to be handed over to Sys Admin Marco. It's important to note here that all issue-related information is already synced and sent over to Global's Azure DevOps.



Here the UI fields must already be mapped from one system to the other i.e any issue or incident filed in Zendesk would translate as a project in ServiceNow, a task in Jira, and a work item in Azure DevOps

6. Here the synchronization process starts through automatic triggers: feature requests/ tickets.

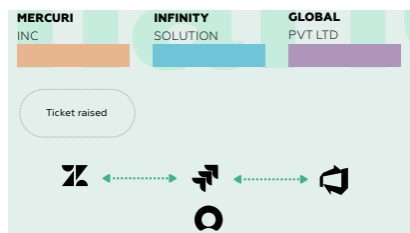
7. A consistent and correct entry about the current status of the issue is made in ServiceNow (Infinity), Jira (Infinity), and Azure DevOps (Global) throughout. Any additional information is passed either as comments or attachments to help all the teams resolve the issue at the earliest.

8. The issue status is also updated on Mercuri's Zendesk so that the support agent on its side can view its current status and inform the customer if required.

Infinity, Global along with Mercuri are able to monitor and track it throughout.

9. For the development of the feature request, once implemented and testing successfully established, it needs to migrate towards production. Appropriate status for this is also updated in the respective issue trackers.

10. After it's successfully deployed on production and the proper status has been updated, feedback about the particular issue is provided to the end-user, and post a go-ahead, it stands resolved.



For such a seamless workflow, the governance organization (involving people from all the communicating parties) recommended in step 2 should be able to monitor the incident across its entire lifecycle.

The whole conversation needs to lead to a firm specification of the to-be system in the following categories.

User Stories

Identify the user stories posed by the IT Professionals or stakeholders and have them around for clarifying if necessary. They form an important part of the design process to accumulate ever-changing requirements. For example: As a support representative of a company, I want to have visibility on the status of the issue ticket. So that I can proactively provide feedback to my customer.

Use Cases

After an initial level of specific user stories, move ahead to conceptually design them in the form of use cases. This will help you identify how the users react to your systems. Example: You can diagrammatically show how bi-directional synchronization between 2 systems takes place.

Data Management

- Entity mapping, issue type mapping, automated or manual synchronization, bi-directional or uni-directional synchronization are considered here.
- All the possible and valid data values that your systems will pass to each other are identified.
- Data structure between the 2 systems needs to be accounted for. Because if 1 system has fields longer than another, data can be truncated.
- You must know what information needs to be exchanged.

- Address questions like "Do you need a fully integrated synchronization scenario or integration only at specific points"?

Data Security

How will integrity, confidentiality, and accessibility of the data be ensured? For this:

- A role-based access control mechanism can be implemented. Here, read/ write/ full control in terms of an Identity Access Matrix document based on particular roles can be maintained which can be referred by the entire team in case there is any unclarity.
- **Secure file transfer between companies:** The data to be transferred is encrypted (HTTPS) with properly defined certificates. Furthermore, an unencrypted transfer between two computers is possible if the remote connection between two locations is encrypted as a Virtual Private Network (VPN).

Managing Different Versions of Systems

Sometimes you might need additional hardware or software to integrate a source application with its target. Sometimes older versions of 2 applications have been integrated, but newer versions haven't. These different permutations of versions must be known and handled beforehand to ensure the solution is up-to-date with its correct versions.

Customizing the Core Setup

Often different departments or customer or partner systems have their own applications running in their local environments. Each application on both sides of the communication ecosystem has its own features and functions (e.g: custom fields) making it difficult for the integration solutions provider to give a definite solution. So often integration solution providers come up with generic "out-of-the-box" templates which allow the customers and service providers to set up easy-to-use, interactive connections between them.

Workflow Design/ Automation/ Orchestration

This refers to the arrangement and coordination of automated tasks resulting in a consolidated process or workflow. All this is to ensure that your data moves and lands in the right direction/ system.

QA/ QC Process

A cross-company integration QC is different from traditional QC processes. System and integration testing play an important role in an integration scenario and hence exhaustive test cases must be designed around them. The integrator must have knowledge about different service provider systems and customer systems under different scenarios. For instance: A proper test record of whether the data generated from the user end is synchronized correctly at the providers' end must be prepared. This saves a lot of time and effort for everyone.

Rollback/ Deployment Plan

Deploying in live environments can happen as needed. Deployment between environments of testing to staging/ pre-acceptance for roll-backs needs to be designed.

End-to-End Service Levels

Managing end-to-end service levels is necessary to deliver value. The presence of an SLA is to align the customer's expectations with the service providers. As a service integrator, ensure that these end-to-end service levels are met.

Examples of expected service levels can be:

- How reliable is the solution?

- Has fast is the performance of the solution?
- What are the backup mechanisms in case of downtime/ failure?
- Are the systems always connected and running?

Many times, these service levels end up being incidents and can cause errors in the system. Effectively handling these incidents with a "fix-first, argue-later" approach is very important in a multi-sourced environment.

Typical end-to-end service agreement in the above-stated examples can be:

- If data is lost in transition, then it needs to roll back to its previous state, thus ensuring reliability.
- Effective back-ups need to be present for every critical transaction. Downtimes are to be planned in advance and minimal system disruption ensured. System failures need to be handled with faster resolution times without causing a major disturbance for the customers.
- Systems can maintain agreed-upon downtime if they are always connected to ensure predictable service disruptions.

Knowledge Base

For effective operations and support, there is a need to create a knowledge base outlining specific scenarios that can become potential risks/ raise query requests. It can serve to become an "operation hand-book" enlisting the steps to be followed for their workarounds and solutions proposed. This can definitely help in managing service requests in an organized and speedy manner ensuring customer satisfaction.

Training

For the purpose of the upcoming integration setup, training requirements and materials are to be identified. They must exist for all the business process levels – the customers (your organization in the SIAM ecosystem), the service providers, and the service integrators.

Now that you have the right people for your project, thoroughly designed and engineered requirements, specifications, and other technical considerations, you're ready to help your customers pick the best solution.

Note: *Design and transition in ITIL 4. Entails change management, release management, project management, architecture management, knowledge management, information security management.*

Plan and Build in SIAM Roadmap

Step 5: Know the Right Solution for Your Customer

While deciding whether to buy a COTS (Commercial-off-the-shelf) or build an in-house solution to address the integration needs of your organization, the essential output is to deliver value to them through service providers. This is pivotal to ensuring the setup is fit for purpose (utility) and fit for use (warranty).

Before a company can accurately assess the advantages or disadvantages of a make or buy decision, they must clearly spell out the requirements for integration. We have successfully helped build a case from steps 1 to 5 outlining and designing these requirements for our integration setup.

Now, let's explore why developing an in-house integration solution might not always work in your favor.

Handling customization requirements: An integration solution often demands additional customization requests as business processes are fairly dynamic. Each request will get the solution to work a little better every time. If you have an in-house development team, customizations often need to be handled by them. Additionally, these teams work on many projects at the same time and customization of data integration can end up being one of them. This will unnecessarily burden the in-house team and will slow down the entire integration process.

Maintaining the integration solution: It's already an established fact that the majority of software projects incur higher costs during maintenance. These costs run even higher for cross-company integrations. When underlying systems are upgraded, or completely replaced, integration solution needs to adapt to this too. Moreover, documentation maintained on the original application must be rewritten and updated. Knowledge transfer needs to happen. And all this will be burning a hole in an already overworked pocket. Here, costs of maintenance can be significantly reduced by using the right commercial integration solution. This will allow the customers to use expertise in integrating applications seamlessly without the stress of maintenance costs rising rapidly.

Ever-changing configurations: Businesses generally require a lot of changes to be accommodated. These constant changes have a pretty dramatic effect on any cross-company integration. For integration applications developed in-house, this adds additional overheads since they need to be applied to either side of the integration. Changes can include new workflows, permissions, data mapping, and the like. Unless a solid change management process is in place, these changes can cause significant delays if developed in-house.

Every organization must consider its own drivers to achieve the necessary clarity for the anticipated business benefits. But with a cross-company integration solution, you can expect these critical benefits:

- Unique synchronization feature retaining full control of your environment without intervention from the other side. This creates an IT security benefit for both customers and the organization. Keep control over what information needs to be retained by the customers and what needs to be passed between systems. This can make the information exchange more secure and can lower the risk of a data breach. It can allow the systems to be loosely coupled and independent of each other.
- The ease of use of the toolset leaves the users with more time to work on things that matter to them.
- An improvement in the visibility of the data throughout its lifecycle results in effective decision-making.
- A robust synchronization tool can make it easier to troubleshoot known system errors and saves a lot of time.
- A unified platform contributes to a more collaborative cultural shift in an integration ecosystem.
- An integrated retry mechanism can help recover from failures and resume synchronization from the point of interruption.

- For cross-functional teams, integration between toolsets saves time and resources and reduces the possibility of errors. It supports workflow automation, reduces the need to re-enter and translate data. There is less chance of information errors leading to friction between teams.

Step 6: Implement Iteratively

How do you prototype and implement the cross-company integration solution?

"Seeing is believing" holds true in this step. This is where you simulate the cross-company integration solution. Maybe even get a free trial, configure it and see how it works. To perform this activity, you can consider the most important use cases (identified from the pain points) relevant to your integration scenario.

The Scenario

For Infinity, it can be to help them synchronize bi-directionally between Jira, ServiceNow, and Azure DeOps for issues of "High Priority". Their team could then configure a version of the intended product and see how efficiently information gets exchanged and synchronized automatically between these work management systems.

This exercise helps with establishing trust and customer confidence. It demonstrates the technical feasibility of the product too. Once confirmed, you can move towards building the solution iteratively managed by the implementation team.

You can perform the actual installation of the newly-developed or acquired system. You can do this one work management system at a time, one service provider at a time, one practice at a time, or one process at a time (essentially following Agile practices). This approach helps to transition using small, effective, easily managed tasks/ iterations. It can significantly reduce the risks associated with the "big bang" approach. The idea is to ensure the delivery of existing services with minimal disruption.

Another part of this step is verification and validation, both of which will help ensure the program's successful completion.

Note: *Obtain/ Build in ITIL 4. Entails Project Management.*

Implement in SIAM Roadmap

Step 7: Deploy to Production

For smooth deployment of a project to production, there must be a clear and comprehensive release and deployment plan. It is important to ensure that any problems with new services like the built release, changes for new features, are installed and tested before deploying the package into production. This will reduce the possibility of facing problems in production.

Efficient roll-back mechanisms are to be considered as well.

The Scenario

There is a synchronization request for change processing triggered from Mercuri (Zendesk) to Infinity (Jira), and the connectivity is lost during this transaction. Here, once the connectivity is up, all changes must be queued and applied automatically in the same order as the original event. Such an integrated retry mechanism will ensure that Mercuri and Infinity didn't even notice the outage and were rest assured that things were handled effectively by the integration solution.

Training requirements are met here. Additionally, training efforts must also satisfy the training requirements of implementation engineers. After all, they are the key drivers to maintaining and operating the cross-company integration as improvements are constantly made.

Note: *Release and Deployment Management in ITIL 4*

Step 8: Operation and Support

How do you organize operations and support in a cross-company setting?

The eighth and final phase involves maintenance and periodical updates. This step is when the system is fine-tuned based on end-user feedback to boost performance, add new capabilities or meet additional user requirements.

Any incident here is analyzed through its "root cause" by the maintenance and support team. The operation handbook proposed in step 4 can help identify if the solution to this incident is available or not. The solution, if available, is applied immediately and the incident is closed. This makes the resolution quicker, positively impacting the SLA and leaving you with happy customers.

However, if the incident is not present in the hand-book, it needs to be categorized as either a:

- Bug/ Issue- e.g: A wrong report with inaccurate data or an end-user not being able to open a PDF report.
- A service request- e.g: End-user not knowing how to view a particular PDF.
- Change request- e.g: Demanding a new feature such as a change in the UI/UX. Such a change needs a business head's approval. The time required to resolve it is greater than a bug or a service request (as their SLAs are different). Once the change has been implemented, it needs to undergo change regression testing to see how it impacts other parts of the existing system.

Here, there can be a category to map the relative importance of the incident based on its type. This can be stated under priority. It is based on the impact and urgency and can be used to identify the required time for actions to be taken.

		IMPACT			
		1	2	3	4
URGENCY	1	critical	critical	high	high
	2	critical	high	high	medium
	3	high	medium	medium	medium
	4	low	low	low	low

The Scenario

For example, the SLA between Infinity and Mercuri may state that the priority 2 incidents must be resolved within 12 hours. So if such an incident occurs then the average time for resolving the incident must be brought down to lower than 12 hours.

The ultimate goal here: There needs to be a positive improvement in the SLA closure time. For instance, if the average time for resolving a priority 1 incident in the SLA is 2 hours, then you must be able to bring it down to 1.5 hours.

Hence, the essence of this practice is to be able to identify, at the earliest, whether a particular incident is a bug, a service request, or a change and to assign the correct priority to it. Then hand it over to the responsible person/ department after routing it through the service integrator. And all this is taken care of in a shorter time.

The confusion over incidents coming under the purview of the integration solution provider or being related to the information exchange mapping between companies needs to be acknowledged beforehand. If not, it creates "no ownership" of issues between the participating companies leaving a huge impact on the customers.

These practices are crucial for an effective operation and value co-creation activities for cross-company integration.

Note: *Deliver and Support in ITIL 4. Entails incident management, change management, and service request management.*

Run and Improve in SIAM Roadmap

Conclusion

Evidently, this guide offers you an approach to a powerful cross-company integration tool that co-creates value in multi-supplier ecosystems. It incorporates ITIL 4 and SIAM methodologies together. It also serves to illustrate clearly defined steps validated by industry leaders and practiced by IT Professionals. However, it is definitely not cast in stone as we realize that every organization is different, and every situation novel. So there is no single "Do-It-Yourself" that applies to each and every situation. You need to formulate your own unique strategy taking into consideration the benefits accrued.

Special thanks to Claire Agutter from [ITSM Zone](#), Alexander Schmidt from [Value Insights](#), Aggelos Paraskevopoulos from [Cententia](#), and Nirupama Dhaygude from [IKS Health](#), who helped us validate each and every step in this guide.