Using HPE-Provided Resources to Integrate HPE Support into Internal Incident Management

1st John Gann
NERSC
Lawrence Berkeley National Laboratory
Berkeley, CA
jgann@lbl.gov

2nd Daniel Gens
NERSC
Lawrence Berkeley National Laboratory
Berkeley, CA
dygens@lbl.gov

3rd Elizabeth Bautista
NERSC
Lawrence Berkeley National Laboratory
Berkeley, CA
ejbautista@lbl.gov

***Abstract*—High Performance Computing (HPC) has a demand for streamlining incident management workflows while keeping information synchronized between internal tickets and vendor support cases. Before HPE acquired Cray, NERSC developed an integration process between their ServiceNow incident management platform and the Crayport platform. This is now obsolete once HPE purchased Cray and NERSC staff had no choice but to manually input information each time a new incident was opened or required updating. Further, this manual entry needed to be done in both ServiceNow and HPE's platforms.**

This paper presents a novel integration between ServiceNow, a leading IT service management tool, and Hewlett Packard Enterprise's (HPE) support portal. This approach ensures a synchronized and efficient workflow between the two platforms, enhances visibility and tracking of issues, leads to quicker resolutions and improved service levels, and creates an automated method to update incidents. We will detail the development journey, the technical challenges we overcame (including some ServiceNow-specific security concerns), and the operational benefits of this integration. Readers of this paper will gain insights into the strategic planning and technical considerations required for such integrations, making them well-equipped to handle similar transitions and integrations in their respective organizations.

***Index Terms*—HPE Support Incident Management ServiceNow**

I. Introduction

The National Energy Research Scientific Computing Center (NERSC) is the premier scientific non-classified computing facility funded by the U.S. Department of Energy (DOE) Office of Science. As a source of high-performance computing and data analysis capabilities, NERSC serves over 10,000 scientists worldwide and has more than 900 research projects spanning various scientific disciplines. NERSC's current system is Perlmutter, an HPE Cray Shasta system comprised of 3,072 CPU-only and 1,792 GPU-accelerated nodes.

On such a large-scale system with many complex components, many technical issues regularly need diagnosing and remediation, particularly immediately after installation or significant upgrades. Managing, documenting, and responding to these issues quickly and efficiently is key to making this system highly available to NERSC users. Resolving these issues promptly requires close collaboration between NERSC staff and HPE support personnel. In our current environment, NERSC engineers collaborate with onsite HPE support staff, submitting an average of 75 hardware support requests each month. During outlier events, such as when new software has been installed or system maintenances, this number can dramatically increase, making coordination especially challenging and requiring service management tools that reduce the manual overhead of communicating between the two organizations.

Several tools exist for tracking issues and incident management in production IT environments, such Atlassian's JIRA [1], Zendesk [2], Salesforce Service Cloud [3], and ServiceNow [4]. These tools play a crucial role in documenting system and data center issues. They allow staff to consolidate relevant monitoring information, narrative statements about remediation steps, and inter-staff communication about an incident. This last aspect is vital when managing HPC systems where even relatively simple issues may require experts in different domains to collaborate to solve an issue. However, at NERSC, the effective use of a single platform for this communication is stymied because NERSC and HPE use two non-interoperable systems.

NERSC has used ServiceNow for more than a decade. HPE uses an internally developed platform called the HPE Digital Customer Experience, which is Salesforce-based. For this reason, when an issue arises on NERSC HPC systems that requires support from HPE personnel, it is necessary to input the information into both systems manually. This can potentially cause errors where both platforms do not have the same information.

A. Integration with CrayPort

Before HPE acquired Cray, NERSC internally developed a software integration between the ServiceNow incident management platform and Crayport [5]. This integration allowed us to open Cray cases directly from ServiceNow and, more importantly, synchronized service notes between the two platforms in near real-time, so there was timely, robust, and error-free communication between NERSC and Cray stakeholders.

Shortly after HPE's acquisition of Cray Inc. Crayport was retired, and NERSC's existing integration was no longer functional. Following that, engineers had to open cases in HPE's Digital Customer Experience (DCE) and manually copy and paste any pertinent information between DCE and NERSC's ServiceNow instance. While the administrative overhead of having to do a formerly automated task by hand is significant,

it is notable that because of differences between CrayPort and HPE DCE, managing HPE cases manually had a higher administrative burden than opening Cray cases manually before an integration was even implemented.

This paper addresses the design of a similar integration that works with HPE's case management system. For various technical, security, and administrative reasons, designing such an integration involved overcoming several challenges that were not encountered when designing and deploying the previous integration with CrayPort. This paper documents these challenges and the solutions used and explores potential alternative solutions where they may be edifying.

II. Summary of The Integration

This integration allows users to open, update, and close HPE cases directly from ServiceNow, in a part of the interface where information relevant to the case is readily available and programmatically accessible. The dialog box to open a case is accessible from the ServiceNow form, which accesses an individual incident record. When a case is opened, it is automatically paired with this record, and relevant information from the incident record is used to populate case data, reducing both administrative overhead and the number of errors introduced by copying the information over manually. Once a case has been opened, updates to the corresponding Incident record are reflected in the case, and vice versa, ensuring that all parties from NERSC and HPE are working with the same information.

A. User Interface Basics

From the Incident record view in ServiceNow, there is a button at the top of the page to open an HPE case. When this button is clicked a dialog box (Figure 1) appears, with fields that allow you to specify which HPE-provided system the case refers to, the priority of the case, the title of a the case, and the description of why the case was opened. The field to specify the HPE system is restricted to HPE systems for which NERSC's ServiceNow instance has an existing record. The field for the priority of the case is a dropdown that allows the user to select "Normal," "Critical Degraded," or "Critical. Down". Both the title and description fields are free text fields. The fields for the system and title are pre-populated from the corresponding data in the incident record. There is also a field that specifies which ServiceNow Incident record the case will be paired with, which is also pre-populated from the Incident record, but unlike the system and title fields, cannot be changed by the user.

Once this information is filled in to the user's satisfaction, it can be submitted to HPE with the submit button at the bottom of the dialog box.

Once a case has been opened, updates to the case made by HPE are reflected in the ServiceNow case record as narrative entries in the "Additional Comments" field, and updates to the case can be made by making a narrative entry in ServiceNow.

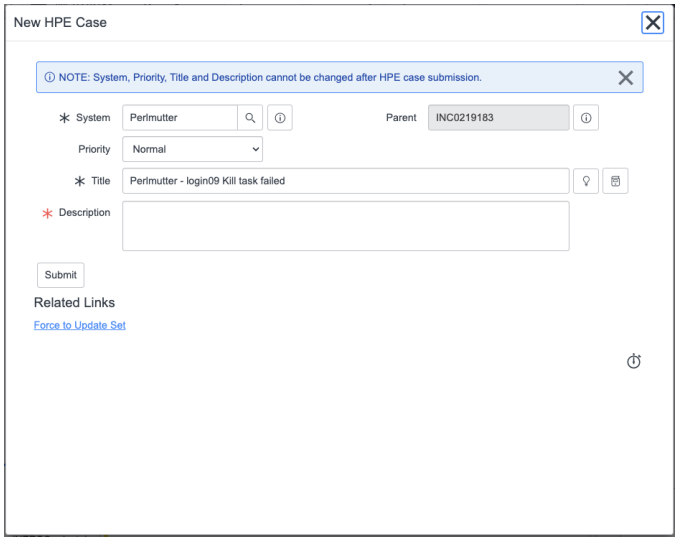

Fig. 1. Dialog Box within ServiceNow for Opening an HPE Case

B. Timeliness of Synchronization

Updates made by HPE are reflected in the corresponding ServiceNow incident record in near real-time, while opening cases and updating information from ServiceNow might take a bit longer. The nature of this delay is discussed in the section of this paper dealing with the integration's limitations. In practice, these delays are not substantial enough to significantly impede workflow, although they can impact the smoothness of interacting with the user interface.

III. Internal Structure of the Application

A. The Three Component Parts

There are three component parts to the integration. Naturally, ServiceNow is one of these parts since it is used at NERSC for internal incident management. To interface with HPE case management, we use the HPE-provided Global Service Event Management(GSEM) platform. Additionally, since there are some instances where it was necessary to mediate the interaction between these two platforms, an intermediate API runs in NERSC's data center that helps facilitate outbound requests to GSEM.

1) ServiceNow: ServiceNow offers several facilities that enable the integration to function smoothly. First, within the interface to update a record (or, in our case, a "ticket"), there exist scripts that are triggered by changes in the form for the record. These scripts can run on both the front end (triggering changes to other fields in the form) or on the back end, which can perform a variety of functions, including making changes to the database underlying the records, making a request to an external API, or most things that could be expected of server-side programming. Additionally, ServiceNow provides the ability to script RESTful APIs, where an HTTP endpoint is exposed to the world. When receiving a JSON payload with proper authentication, the contents of the payload can be used in a back-end script whose running is triggered by the

payload's receipt. Each of these features —front-end scripts, back-end scripts, and scripted REST APIs — is used by our integration to provide an easy-to-use user interface and facilitate communication between the integration's components.

a) Front-end scripts: Front-end scripts in ServiceNow enable client-side form actions, such as enabling a field when another has been filled out with a specific value. These scripts are used in the integration to prevent the user from performing nonsensical or undesirable actions, such as updating a closed case or closing a case without closure notes. They also provide helpful messages and notes that guide users as they fill out the case creation dialogue.

b) Back-end scripts: The ServiceNow server-side scripts used by the integration fall into three main categories.

First, there are business rules that run when records—such as cases or incidents— are displayed, inserted, or deleted or when a table of records is queried. These are used by the integration to handle the "bookkeeping" of ServiceNow records, such as generating appropriate serial numbers for external reference.

c) Scripted REST APIs: The integration uses scripted REST APIS to accommodate inbound information about changes to case information. GSEM can be configured to send such information using a user-supplied endpoint, and our integration uses this facility to receive information about case creation, updates, and closure.

2) Global Service Event Management: One uses the HPE DCE to open cases manually, but an alternate facility exists for creating and manipulating HPE cases programmatically. The HPE documentation introduces the Global Service Event Management (GSEM) system, which is designed to automate the transfer of calls between different Call Handling Systems (CHS) without human interaction. GSEM is designed for both internal (HPE to HPE) and external (Customers to HPE, HPE to Partners) communications. This system uses the Service Incident Exchange Standard (SIS), which provides a comprehensive framework for the GSEM process. SIS covers the data flow, call states, and the structure and content of the data exchanged in these communications. The act of passing a call within the SIS framework is referred to as a "Service Request," while the interactions between a Requester (the entity initiating the call) and a Provider (the entity receiving the call) are termed "Transactions." These transactions are designed to communicate information or change the state of the Service Requests, promoting a dynamic and efficient response system. SIS and GSEM aim to streamline communication between service requesters and providers, enhancing efficiency in high-volume service environments and reducing manual errors in service incident management. The documentation emphasizes practical application and benefits. It offers guidance for integrating GSEM and SIS into existing frameworks, improving service incident management. It is beneficial for sites and users looking to understand or integrate the GSEM's operational framework.

3) Intermediate API: While it is obvious that any integration between local incident management and HPE case management must necessarily include both of those systems as components, there is also a third component to the integration. NERSC's data center hosts a server running a simple API that performs minimal translation of outbound communications from ServiceNow and relays them to GSEM. This intermediate software was necessary for various reasons, including security requirements, the need for a more robust programming environment than ServiceNow provides, and the enhanced observability that comes with a locally running service.

Security considerations were the main driver of needing this intermediate API. HPE requires host-specific certificates to interact with GSEM and does not permit certificates with a wildcard host. Because ServiceNow is a distributed Software-as-a-Service platform, no guarantees can be made as to what specific ServiceNow server will be making the call to GSEM. Therefore, even though ServiceNow allows for the installation of certificates for the purposes of interacting with external APIs, a certificate that GSEM will consistently accept cannot be embedded. Other incident management platforms that can make guarantees as to what host is making outbound API calls may not have this restriction. Nevertheless, to provide a consistent host for interacting with GSEM, all API calls that would originate from ServiceNow to GSEM are instead generated by the server hosting the intermediate API.

Secondary to the security considerations, a more robust programming environment was needed than ServiceNow could provide. ServiceNow is a full-featured incident management platform, and its programming environment exists to support that. However, due to the platform's distributed cloud nature and the fact that at the time of the integration's development, ServiceNow only supported ECMAScript 5 [11], attempting to write software on the platform that extends beyond moderate complexity incident management tasks was often challenging due to the lack of such facilities as advanced flow control, easy ability to import third-party libraries, and synchronization primitives. While a simpler version of the integration could have been written without the use of the intermediate server, it would have lacked such essential features as retry logic, robust error handling, and sensible default behavior.

Finally, mediating the outbound data flows with an intermediate server allowed us greater visibility during key points of the development process, including during debugging, unit and integration testing, and security reviews. Due to the many levels of abstraction present in the ServiceNow platform, debugging things as straightforward as API calls and responses can be unnecessarily complex as any log data must be interpreted keeping the distributed, parallel nature of the platform in mind. Having outbound requests mediated by a simple service consisting of a few hundred lines of Python made debugging much more transparent.

The intermediate API is secured by two means. First, it is only accessible within NERSC's internal network. ServiceNow can access it via privileged communication through a ServiceNow Management, Instrumentation, and Discovery(MID) server that resides on the internal network. Secondly, it is protected with a username and password pair over HTTPS.

B. Data Flows

These three parts work together to make the integration function. There are two main categories of data flows that happen in different circumstances. When a data update is initiated in ServiceNow, this is referred to as an outbound data flow. Conversely, when a data update is initiated by HPE and the data needs to be integrated into ServiceNow, this is referred to as an inbound data flow. It is generally intuitive when each is applicable, with one notable exception: When a case is created in ServiceNow.

1) Case Creation Data Flows: When a case is opened from ServiceNow, there is an outbound data flow informing HPE via GSEM. When this happens, scripting on ServiceNow also creates a placeholder record for the HPE case. Since the case creation process takes some time, there is not an immediate HTTP response. Instead, after some time, there is an inbound data flow that contains the case ID and other information, which is then integrated into the temporary case record.

2) Outbound Data Flows: Data that is flowing from the ServiceNow platform to HPE is mediated by the intermediate API. Examples of an outbound data flow are the initial step of case creation, ServiceNow-initiated case updates, and case closure. Each of these three actions has a corresponding endpoint on the intermediate API. These endpoints receive limited data from ServiceNow and transform it into the payload needed by GSEM. As an example, for case creation, the following pieces of information are sent:

- *Incident Record Identifier*: The corresponding incident record identifier used by ServiceNow. For example: INC0098453
- *Case Severity*: The severity of the case, as selected from the dropdown menu in the dialogue box
- *Originator's First Name*: The first name of the ServiceNow user opening the case.
- *Originator's Last Name*: The last name of the ServiceNow user opening the case.
- *System Serial Number*: The HPE-assigned serial number for the hardware asset the case concerns.
- *Case Tracking Number*: The corresponding incident record identifier prepended to a one-up serial number of the ServiceNow case record.
- *Case Title*: The title of the case, as supplied by the user in the case creation dialogue box.
- *Case Description*: The long-form case description supplied by the user in the case creation dialogue box.

The intermediate API supplements this information with static values, values generated at the moment of transmission (such as the submission time), or values found via lookup tables using the information provided by ServiceNow as keys (such as the contract number corresponding to a specific hardware asset). An example of the entire JSON object constructed from these values can be found in Appendix A. This JSON blob represents the minimum amount of information that can be used to open a case with GSEM, and this was not a fact that could be derived from existing documentation. While many of the fields in the JSON blob are self-explanatory, the keys in the "AGREEMENT" block are worth examining. "Contract ID" references a contract number associated with a specific HPE system. The number to use may not be obvious, as it does not appear in contexts outside of GSEM, and this information needs to be gleaned from a site's GSEM point of contact. The "AgreementType" field is a combination of the same system serial number used in HPE DCE (SN) combined with whatever identifier is being used for the case locally(LN).

3) Inbound Data Flows: Data about HPE cases flowing from HPE to NERSC's ServiceNow instance are not mediated by the intermediate API. This is because security requirements do not necessitate it, and the logic for handling these inbound payloads is reasonably straightforward. It is easily handled by the logic that can be embedded in a Scripted REST API endpoint. Alternative designs confer some benefits, which will be discussed, but ultimately, the simplicity of exposing a Scripted REST API endpoint to HPE won out on the basis of maintainability.

a) Alternate Design Using the GSEM Queue: During development, an alternate design that did not make such extensive use of ServiceNow's Scripted REST APIs was explored. GSEM also offers a queue-based API where outbound messages can be actively fetched and acknowledged when appropriately processed. This alternate design has the server running the intermediate API also actively fetching things from the GSEM queue, processing them, and passing them to ServiceNow either through a single, universal Scripted Rest API, ServiceNow's built-in APIs, or through ServiceNow's Management, Instrumentation, and Discovery platform. This design did offer the advantages of consolidating more of our logic in one place and greater observability of the processing of inbound data flows. However, the cumbersomeness of the multi-step acknowledgment process for queue messages made the integration's code more complex than was desirable. The Scripted REST API implementation performs this same acknowledgment much more simply by responding to inbound payloads with an appropriate HTTP response code after the payload's data has been integrated, but for sites implementing integrations on platforms that do not have such a robust facility for handling inbound HTTP data, using the queue-based systems may be a viable alternative.

IV. Limitations of the Integration

A. Not a Near Real Time Sync

Due to constraints introduced by the nature of the GSEM API, this integration has many limitations. These limitations are expected to be present in any similar integration that leverages GSEM. One such limitation is that the initial opening of cases is not near real-time and takes approximately five minutes, with an additional five minutes required for the case to populate in HPE DCE. This delay is introduced by the GSEM back-end as the process to verify entitlement and look up assets in the remote database is time-consuming. In practice, this delay does not significantly impede case management activities, although the lag in back-end responsiveness

does have some implications for the smoothness of the user experience.

B. Unable to Manage Cases in HPE DCE

Another limitation is that cases are not synchronized between GSEM and HPE DCE. Case updates made via the integration (using GSEM) will be visible to HPE engineers and support staff but will not be present in their corresponding cases in DCE. Therefore, if an organization wishes to use an integration to keep HPE service support information in its local incident management platform, it should expect to forgo using HPE DCE for case management. This is an inconvenience, and HPE support has been consulted to inquire whether support for synchronization between the two platforms would be added. However, it appears there are no plans to do so in the immediate future.

C. HPE User Metadata is not Available

When a case is updated with a comment, metadata containing user information is not included in the incoming GSEM payload; this means that case comments are not attributable to individual HPE staff in ServiceNow. Moreover, this missing user information makes it impossible to determine the assignee for a case without logging in to the HPE DCE portal. This can complicate workflows when trying to identify the person of contact or the team working on a specific case and makes it harder to discern between information provided by local versus remote HPE staff.

V. Navigating the HPE DCE API

HPE provides access to comprehensive documentation, including a general summary and an in-depth overview of the GSEM system, focusing on automated call transfers between Call Handling Systems (CHS). In the case of our integration, these are ServiceNow and HPE DCE. The documentation explains how GSEM facilitates both internal (HPE to HPE) and external (Customers to HPE, HPE to Partners) communications. It introduces the Service Incident Exchange Standard (SIS) that establishes the framework for GSEM calls, including the model for the GSEM process, data flow, the various states of a call, and the structure and content of the data involved. This standard streamlines the transmission of service requests, provides and enforces process standardization, reduces errors, and improves efficiency. While the API documentation is available for review as soon as integration development is initiated, sites must complete the API Authorization process before accessing the API itself, including through the HPE-provided API simulator called the Dev Console.

A. API Authorization

In order to use the GSEM API, sites are required to complete the API authorization process designed to guarantee secure and controlled access. This process is mandatory for both development and production environments. While it is permissible to maintain the same authorization configuration across these environments, each environment requires its authorization to be submitted independently. Only after completing this authorization process can access to the API be granted, whether one is using the API simulator or connecting directly to the API. This ensures that all interactions with the API adhere to established security protocols. A significant part of this is registering the SSL certificates that will be used.

a) SSL Certificate Registration: The only authentication method accepted by HPE for the GSEM API is SSL (Secure Sockets Layer) certificate authentication.

To register the certificate, sites must submit an approved SSL certificate in PKCS#7 (.p7b) format for registration and access [7]. Upon request, HPE will provide a list of trusted CAs used in the certificate review process. A mapping request can be submitted to the HPE Security Gateway by the GSEM Project Manager if the certificate is issued by certificate authority not on the trusted CA list [8]. This team has regular change freezes during which they're unable to register new certificates from CAs that aren't on the trusted list, so this process might take several weeks [9].

b) Administrative Setup and Tool Review: Once the certificate registration is complete, the GSEM Project Manager will provide configuration details for the environment that is being set up, such as an authorized agent key. The Project Manager will also provide training on using the API simulator, and a review of the GSEM documentation upon request [8].

Additionally, developers will have to obtain the list of contract numbers and match them to the associated serial numbers for that organization's HPE systems, most likely through the general HPE point-of-contact or the HPE representative for the site.

B. Working with the Test and Development Environment

a) Dev Console API Simulator: GSEM integration development is facilitated by the HPE-provided API simulator called the Dev Console which mimics the behavior of the real API. It is a web tool that allows one to make API requests to the development environment, and the GSEM engine will provide a response. Dev Console also expects the user to interact with the request queue, acknowledging the messages or clearing the queue as needed. The assigned GSEM Project Manager will provide training and a walkthrough of the Dev Console functionality.

Dev Console allows for a direct connection to the GSEM engine without involving the site's back- or front-end services, and so can be used to triage any pipeline issues. Additionally, users can simulate the Provider functionality, which would enable them to test and explore without involving the GSEM system. This can be especially useful during the development and testing phases for several reasons:

- **Development Efficiency.** Developers can continue working on an application even if the actual front end or back end isn't ready or available. This is particularly useful in large teams where different components are developed in parallel.
- **Testing.** The Dev Console helps developers create consistent, controlled, and predictable responses, enabling

thorough testing of the application. They can simulate various scenarios, including failures, slow responses, and edge cases, which might be difficult or impractical to test through the full integration pipeline.

- **Isolation for Debugging.** Having a direct connection to the GSEM engine is invaluable for troubleshooting the integration. Dev Console makes it possible to isolate integration-pipeline versus API-use issues by allowing the transmission of API requests directly from the browser, thus making it easier to pinpoint the source of problems.

b) Dev Console Authentication: In order to authenticate with the Dev Console interface, the HPE Secure Gateway expects a PFX (.p12) certificate to be installed in your Google Chrome browser. This certificate bundle has to match the approved SSL certificate provided to the HPE GSEM team during the SSL Certificate Registration process. [12]

c) Dev Console Interface Overview: The Dev Console interface introduces a way for developers to easily act as the Requester, the Provider, or both, effectively taking over different components of the data flow. Acting as Requester or Provider is called a Requester or a Provider loop, and acting as both Requester and Provider is known as ”loopback”.

Using the **Requester loop**, developers are able to submit, receive, and respond to requests. The loop can be started with the ”Start” button. Submitting a Service Request requires provide mandatory field information, as illustrated in Fig. 2. Note that the Remote Agent Name selection will depend on whether requests are intended to go into the GSEM queue and to SFDC, or into the Provider queue in the Dev Console. Once all mandatory fields are populated, the ”Generate JSON” button will provide the payload that can submitted to the queue using the ”Submit” button. If working with SFDC remote agent, submitting will send the Service Request payload to the GSEM backend in the development environment where it will be processed in the same way as other requests that did not originate from the Dev Console interface. The GSEM engine or the Provider loop will send a response back to the requester queue with the ”Acknowledge message” button.

Running the **Provider loop** is mostly automated and does not require manually sending requests, and once it’s started it will automatically respond to any pending requests in the queue that can be sent from Dev Console or sent from the integration, as illustrated in Fig. 4. ”Mocking” the provider functionality is particularly useful since site developers would not able to access SFDC in the development or production environment, and so would not be able to send case updates on behalf of HPE.

Running the loopback where developers are acting as both requester and provides full control over data flow and allows them to explore the API and the GSEM documentation. A loopback working session is illustrated in Fig 4.

C. The Service Incident Exchange (SIS) framework

The SIS framework is a key component of the GSEM system. It standardizes the automated exchange of service

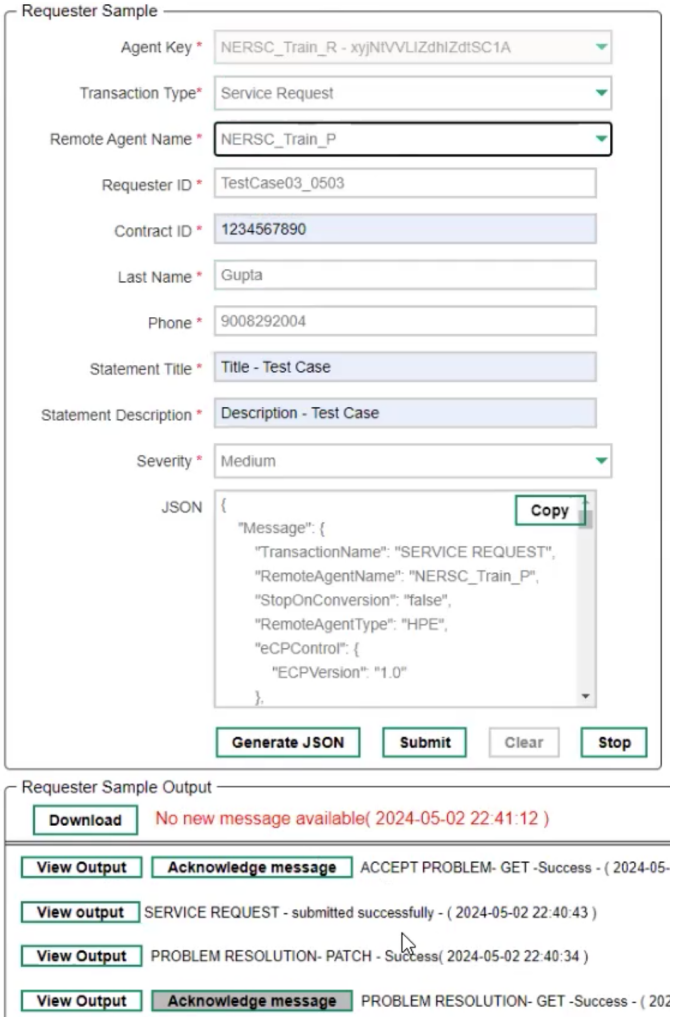

Fig. 2. Running the Requester loop in Dev Console

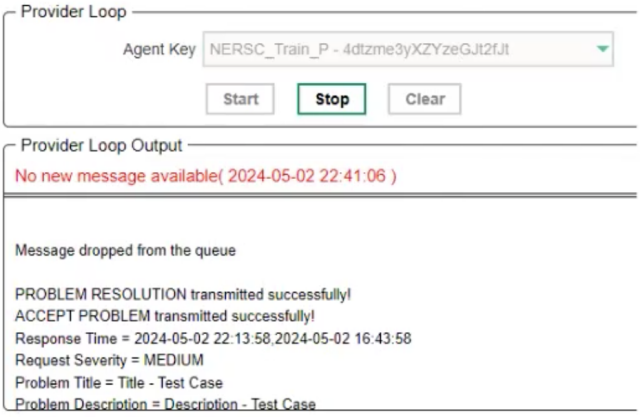

Fig. 3. Running the Provider loop in Dev Console

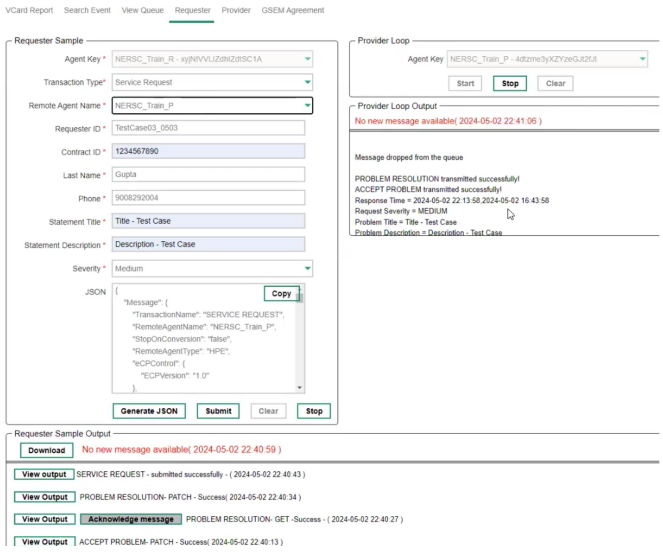

Fig. 4. Loopback session in Dev Console

incidents between different Call Handling Systems. By providing detailed models for process flow, data exchange, call states, and data structures, SIS ensures consistent and efficient communication between the integration and the GSEM engine.

Within the SIS framework, the act of transferring a call is referred to as a ”Service Request”, while the interactions between a Requester (the entity initiating the call) and a Provider (the entity receiving the call) are termed ”Transactions”. These transactions are designed to change the state of the Service Requests according to a defined process flow from initiation to resolution.

A key feature of SIS within GSEM is its data structure and flow management. By standardizing the way calls are passed, updated, and closed, it allows for a more systematic and error-free operation. This is particularly crucial in high-volume service environments where the speed and accuracy of service request handling can significantly impact customer satisfaction and operational efficiency.

In practical terms, the integration of SIS into GSEM means that when a service incident occurs, the Requester, which could be an internal service desk or an external client, uses the system to send out a call. This call is formatted and transmitted based on SIS guidelines, ensuring that it is received and understood correctly by the Provider, which could be another department within HPE or an external partner. The Provider then uses the same standards to update the call status, communicate resolutions, or request further information. This symmetry in communication, facilitated by SIS, ensures that all parties have a clear understanding of the service request and its status at all times.

The advantages of using GSEM powered by SIS include improved resolution times due to efficient communication channels, reduced waiting times for service calls, minimal errors due to standardized data structures, and overall improved service quality.

D. Limitations of Existing Documentation

The existing GSEM documentation is detailed and comprehensive, however it does not cover some broader sections that could inform system design decisions.

a) GSEM to Salesforce Field Mapping: While the GSEM documentation includes a complete mapping for the Service Request object passed between Requester and Provider, it does not go over the mapping from GSEM Service Request object to a Salesforce (SFDC) record that’s referred to as the HPE case. The site’s onsite and offsite HPE contacts, aside from teams specifically related to GSEM engine functionality, will not have access to the GSEM engine records; instead, HPE uses an SFDC instance for case work and tracking. Knowing the mapping from GSEM Data Field names to SFDC field names would help developers ensure that the SFDC record includes all necessary information.

Certain fields that are not mandatory for the GSEM API will be required in order for an SME to work on the case or even for the SFDC case record to appear complete. Once the integration project begins, sites will have an opportunity to work with the GSEM Project Manager to determine and document field mappings from the site’s CHS to GSEM to SFDC. However, when submitting test cases, case records in SFDC can only be evaluated and verified for completeness by your site’s general HPE contact, not the GSEM team. Submitting a case through GSEM that SFDC deems incomplete could result in entitlement issues for this case, delaying case resolution, so it is recommended to confirm case visibility with onsite teams.

b) Trigger Events: GSEM has webhook-like functionality where certain predefined events trigger a case update message. For example, an SFDC case status change event would trigger a case update message that contains the new case status. These events, as well as the expected payload fields and content of the message, could be incorporated into the integration design process. This information was not available for review in the GSEM documentation, and was shared with us by the HPE GSEM team during the development process. It is recommended to request this information early in the process.

VI. Joint HPE/NERSC Development and Project Management

Organizations that are considering developing an integration with the GSEM system should begin by contacting their site’s HPE point of contact. HPE will assign a GSEM Case Exchange Project Management (PM) and Subject Matter Expert (SME) to provide assistance throughout the integration project. Developers should expect to work closely with the HPE GSEM team on all HPE-facing aspects of the integration as the information and experience working with other sites will help inform integration design. The team will also provide guidance through all the steps of the administrative procedure and towards obtaining production release approval.

A. Working with the HPE GSEM Team

As outlined above, the assigned HPE GSEM team will extend support with the following:

- **Dev Console training.** After completing the API authorization process and installing the SSL certificate per HPE requirements, developers can gain access to Dev Console API Simulator with help from the GSEM team. The team will assist with any connectivity issues, and will provide an overview of the Dev Console API simulator, including a demo walkthrough of the tool's functionality upon request.
- **Support and troubleshooting.** The GSEM team can work closely with the site's development team throughout the integration lifecycle and can help validate GSEM connectivity or any other GSEM functionality, troubleshoot any issues during the development, and can give additional overviews of Dev Console and GSEM documentation.
- **Testing.** The GSEM SME will handle any real-time tests of the integration during development and prior to production release. Developers will not have access to the HPE's SFDC instance, so it is not possible to simulate any SFDC case workflow for testing without the help of the HPE GSEM team. Any incremental end-to-end testing will need to be scheduled with the team in advance, depending on their availability.
- **IT Certification.** Before moving to the production phase, any integration with the GSEM system needs to pass the HPE IT certification. This ensures that the integration complies with HPE's standards and is ready for the production GSEM API environment.
- **Future work.** If developers are looking to add new functionality after the initial production release, they should be able to receive support from the HPE GSEM team that was assigned previously. Depending on the extent of these changes, the updated integration might have to be reviewed for IT certification, especially if call handling changes are introduced.

This process helps develop a supportive and productive partnership with HPE GSEM team throughout design, implementation, testing, and production stages, while facilitating a secure and compliant integration.

B. HPE Release Approval

Once the integration is ready for production, and the initial functionality testing is complete, the GSEM Project Manager will discuss and arrange the scheduling for the **HPE IT certification**, which is a review process with the HPE IT certification team that usually lasts approximately 1.5 hours. This meeting involves the HPE GSEM team, the IT certification team, and the developers responsible for the integration.

The HPE IT certification process focuses on preserving the performance and functionality of the GSEM IT infrastructure. Although this certification step is not documented in GSEM documentation or other related materials, it is mandatory before moving to a production environment. This process guarantees that the new integration will not disrupt the existing functions or compromise the system integrity of the GSEM platform and that it is compatible and reliable within the larger HPE ecosystem.

The certification process is designed to cover all functionality included in the integration. Participants are required to complete a series of tests and evaluations, each aimed at different aspects of the integration, including but not limited to system compatibility, data integrity, and operational resilience. These evaluations are conducted in a controlled environment to simulate various operational scenarios and potential system disruptions.

The certification components are as follows:

1) **System compatibility and general functionality testing.** During this stage, participants are expected to demonstrate all aspects of the integration and ensure all functionality is working as expected. This testing includes handling extremely long strings and strings with special characters.
2) **Operational resilience.** In the event of an outage, participants must demonstrate the operational resilience of the integration by confirming that the system can resume operations without or with minimal intervention.
3) **Data integrity testing.** In the event of an outage, the integration must preserve data integrity and message order, and continue to process or send messages from the queue in the correct order after the outage is resolved.

While the first two components are relatively straightforward, the data integrity testing process has some notable details. The most significant crucial component of this process is the establishment and verification of a message queuing and retrigger mechanism. By requiring these mechanisms to be in place, HPE aims to safeguard against data loss, preserve data integrity and message order, and ensure continuous, uninterrupted service.

There are two main scenarios used to test the message queueing and retrigger mechanism:

- *GSEM is offline, or the network is down.* The IT certification team will bring the test GSEM endpoint offline and examine the behavior during and after this outage. GSEM system design provides a message acknowledgement mechanism, and the integration should retry the most recent message until the endpoint is back online and able to acknowledge sent messages. The integration should then continue to send messages from the integration queue and process responses in the order in which they were received.
- *Integration server or integration endpoint is offline.* Integration developers will bring the test endpoint offline and examine the behavior. GSEM will continue sending messages to the queue, and once the integration is back online, these messages should be processed in the order they were received.

The IT certification process allows for multiple attempts. This approach provides developers with the flexibility to troubleshoot and adjust their architecture based on the feedback and results from each certification attempt. It encourages a cycle of continuous improvement and learning, ensuring that by the time it is signed off, the integration is fully optimized and meets all of HPE's standards.

Ultimately, the HPE IT certification process for GSEM integration is not just a procedural formality but a critical step in ensuring that new integrations enhance the overall system's functionality without compromising its integrity. By adhering to this standard, HPE ensures that all integrated systems are reliable, efficient, and capable of withstanding the challenges of a dynamic IT environment.

C. Legal Considerations

One of the prerequisites for using GSEM in production is signing a GSEM EULA (End-User License Agreement). Developers should be aware that the GSEM EULA is comprehensive and may contain unexpected provisions or a process that conflicts with the existing contract. Developers working on the integration should discuss this process with their assigned HPE lead early in the development process to ensure that using GSEM is compatible with their contract, and to be mindful of potential delays or roadblocks caused by the EULA approval process.

In the case of NERSC, we had to include GSEM integration as part of an "additional" contract renewal. A challenge was around the language of the usage of the software. It resulted in a lengthy negotiation process between HPE and NERSC that took several months for both sides to agree and finalize the contract.

VII. Post-deployment Experience

After deploying the integration to production, our internal staff users, as well as the onsite HPE team, began reporting issues with submitting or interacting with new cases. These issues were triaged and resolved by HPE after an initial delay caused by not being able to find the right point-of-contact for remediation.

A. User and Organization Management

In the HPE DCE system, organizations include members who are authorized to submit cases. Each organization is linked to specific contract numbers derived from HPC system serial numbers. When a case is submitted, the associated contract number is verified against the organization's record to confirm system ownership. Incorrect contract numbers lead to case rejection during entitlement verification or submission under a "test" organization. Additionally, incorrect user-organization membership can lead to case rejection or case visibility issues.

B. Ensure End-User Visibility for Support Engineers

The HPE GSEM team's permissions and visibility settings differ from those of the onsite HPE engineers who actually manage the cases, leading to some false assurances that permissions are configured correctly during testing with the HPE GSEM team. To effectively address issues, HPE support engineers need to receive case numbers and have visibility of the support cases in the system. Visibility issues in SFDC often cause delays in resolving these issues. In order to prevent such delays, it is recommended that developers establish a review process before production deployment to ensure all support engineers have end-user visibility, which can be verified by submitting various test cases.

VIII. Future work

There are many opportunities to improve upon the integration presented in this paper, including some refinements to the application that would remediate the issues discovered post-deployment and some new features that would enhance functionality and ease of use.

A. Remediation for ongoing issues

There are some ongoing issues with the integration that, while not impacting the functionality, do sometimes result in delays or confusion.

- **Incomplete case information.** Cases come in with wrong or incomplete information, while the payload is sent with correct data. There have only been a handful of these instances, and we are working with the HPE point-of-contact to troubleshoot this issue.
- **Case visibility issues.** We came across some problems regarding inconsistent case visibility, wherein certain NERSC staff could access a case in HPE DCE while others could not. This issue is probably linked to varying permission levels within HPE DCE, and efforts are underway to address it.
- **Case data differs depending on the submission method.** In some situations where very similar submissions were made through GSEM as through HPE DCE, GSEM cases would be stripped of user contact info, would sometimes have additional populated fields that were not requested and would end up with wrong data. This issue has been difficult to isolate and might be resolved as there has not been any new occurrences for several months.

B. Planned features

1) Support for Attachments: Both HPE cases and ServiceNow Incident records support attachments, but the integration does not currently sync attachment data between the two. A cursory examination of the relevant APIs suggests that implementing this should be possible. Adding this functionality would allow the easy sharing of log information, photographs of visible hardware faults, and other data that does not easily fit into short narrative fields.

2) Onsite Task Tracking: Currently, one of the most common case updates made from the HPE side are updates about tasks that onsite HPE staff are performing. These updates are made automatically and, like other updates, show up as comments in the ServiceNow Incident record corresponding to the case. However, since these updates are made programmatically, they are not particularly human-friendly and are challenging to read. It would be possible to process this information into its own record type, which could be subordinated to the relevant ServiceNow Incident record, making the information easier to work with and keeping the narrative fields more focused on actionable information.

3) Part Order Tracking: In addition to onsite task tracking, updates concerning part order tracking are also common and similarly pollute what would otherwise be a human-friendly narrative text field. Introducing a new record type that can exist in a parent-child relationship to the relevant Incident record would be helpful to enhance readability, consolidate information about parts shipments in one place, and enhance the ability to track and interact with that information programmatically after it has been ingested into ServiceNow.

ACKNOWLEDGMENTS

This manuscript has been authored by an author at Lawrence Berkeley National Laboratory under Contract No. DE-AC02-05CH11231 with the U.S. Department of Energy. The U.S. Government retains, and the publisher, by accepting the article for publication, acknowledges, that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under Contract No. DE-AC02-05CH11231.

REFERENCES

[1] https://jira.atlassian.com/. Accessed 12 Apr. 2024.
[2] https://www.zendesk.com/. Accessed 12 Apr. 2024.
[3] https://www.salesforce.com/products/service-cloud/overview/. Accessed 12 Apr. 2024.
[4] https://www.servicenow.com/. Accessed 12 Apr. 2024.
[5] D. Gens, O. James, E. Bautista, and M. Abdelbaky, "The Art of Conversation with CrayPort (Bidirectional Record Management)," in Cray Users Group 2019 Proceedings, 2019
[6] https://docs.oracle.com/cd/E13211_01/wle/security/publicke.htm. Accessed 27 Apr. 2024
[7] https://api-uns-sgw.ext.hpe.com/gw/hpit/gsd/gsemhlp/apiauth.html. Accessed 27 Apr. 2024
[8] D. Carreiro, private communication, Jan. 2022
[9] N. Gupta, private communication, Apr. 2024
[10] https://www.w3.org/TR/WCAG20/#a. Accessed 2 May. 2024
[11] https://www.servicenow.com/community/developer-advocate-blog/learn-ecmascript-2021-with-earl-duque/ba-p/2804911. Accessed 3 May, 2024
[12] D. Carreiro, private communication, Feb. 2022

APPENDIX A
SAMPLE JSON BLOB FOR GSEM


```
{
'TransactionName': 'SERVICE REQUEST',
'RemoteAgentName': 'SFDC_Train_REST_P042',
'RemoteAgentType': 'CUSTOMER',
'eCPControl': {
  'ECPVersion': '1.0'
            },
'SERVICE_INCIDENT': {
  'RequesterID': 'INC0097539',
  'RequesterSeverity': 'MAJOR',
  'ResponseTime': '2024-04-29 04:52:40',
  'SERVICE_REQUESTER': {
    'ORGANIZATION': {
      'OrganizationName': 'NERSC',
      'ADDRESS': {
        'GeoAddress1': '1 Cyclotron Rd',
        'City': 'Berkeley',
        'Region': 'CA',
        'PostalCode': '94720',
        'Country': 'US'
                }
                  },
      'PERSON': {
        'FirstName': 'John',
        'LastName': 'Gann',
        'ADDRESS': {
          'LOCATION': [
            {
            'Type': 'PRIMARY_VOICE',
            'ID': '5104866821'
            },
            {
            'Type': 'EMAIL',
            'ID': 'jgann@lbl.gov'
            }
                      ]
                    }
                }
                  },
      'AGREEMENT': {
        'ContractID': 'CN=2155240525',
        'AgreementType': 'SN=5UF044F7CY;LN=INC0097539_3'
                  },
      'PROBLEM': {
        'EXPRESSION': {
          'Relation': 'AND',
          'STATEMENT': [
            {
            'StatementRole': 'DESCRIPTION',
            'StatementText':
            'The are issues on the high speed
            network that we suspect are caused
            by faulty hardware.'
            },
            {
            'StatementRole': 'TITLE',
            'StatementText':
            'HSN issues caused by possible
            hardware fault.'
            }
                        ]
                      }
                },
      'ACTIVITY': [
```

```
        {
        'ParameterList': 'Problem:Add',
        'ActionLog': 'Alternate Contact',
        'LocalDate': '2024-04-29 04:52:40',
        'NameValuePairList': {
          'eCPExtMessage_eCPExtControl_Version': '1.0',
          'eCPExtMessage_EXT_SERVICE_INCIDENT_SERVICE
          _REQUESTERALTERNATE_PERSON_FirstName':
          'NERSC',
          'eCPExtMessage_EXT_SERVICE_INCIDENT_SERVICE
          _REQUESTERALTERNATE_PERSON_LastName':
          'Operations',
          'eCPExtMessage_EXT_SERVICE_INCIDENT
          _SERVICEREQUESTERALTERNATE_PERSON_ADDRESS
          _LOCATION_PRIMARY_VOICE':
          '5104866821',
          'eCPExtMessage_EXT_SERVICE_INCIDENT
          _SERVICEREQUESTERALTERNATE_PERSON_ADDRESS
          _LOCATION_EMAIL':
          'operator@nersc.gov'
                              }
        }
                ]
    }
}
```