**ECGRIDOS API Getting Started - Extended Capabilities**

*Take control of EDI and B2B Document exchange with our library of APIs.*

**Contents**

* [Introduction to ECGridOS](#Bookmark1) 1
* [Quick Reference Notes](#Bookmark2)  4
* [Getting Connected to ECGridOS](#Bookmark3)  4
	+ Postman
	+ Visual Studio
* [Getting Started with ECGridOS](#Bookmark4)  6
	+ First Steps
	+ Commonly Used Calls – Extended Capabilities
		- Upload Document
		- Download Document
		- Look up Transactions
		- Document Error Check
		- Administrative
		- Find Trading Partner on ECGrid Network
* [ECGridOS Nomenclature](#Bookmark5)  11
* [Technical Nomenclature](#Bookmark6)  13
* [Commonly Asked Questions](#Bookmark7)  14

**Introduction to ECGridOS™**

The ECGridOS platform was developed by Loren Data Corp., an established provider of supply chain messaging infrastructure and services since 1996. ECGridOS™ is an Operating System as a Service (OSaaS), that manages the ECGrid® Supply Chain Integration Network, transporting transactions, messages, and reports between businesses around the world. ECGridOS hosts a robust set of discrete platform processes such as Web Service APIs, enabling developers to integrate supply chain communications capabilities into mission-critical applications, systems, and workflows.

ECGridOS is specifically designed for quick, secure, and accurate electronic business documents exchange, such as purchase orders, invoices, and shipping notices, between hundreds of thousands of trading partners around the globe. ECGrid supports all file formats including legacy Electronic Data Interchange (EDI) standards such as X12 and EDIFACT as well as all modern formats including XML, JSON and all user-defined data formats.

Cloud applications frequently publish sets of web service APIs to extend access into system data, allowing users and developers basic functions to integrate externally. ECGridOS takes web services to the next level as a 100% API-based web service-based operating system. ECGridOS delivers all the capabilities required to internally operate and externally integrate a global supply chain communications network.

ECGridOS publishes an extensive library of APIs which delivers full command, control, and communication functionality over operations, integrations, and implementations. Like a traditional operating system, ECGridOS is a foundational application with over 200 fine grain API’s, covering all aspects of EDI network configuration and communications. ECGridOS delivers all core ECGrid internal processes and provides the platform for all external communication channels and user portal access. External developers will find the complete set of tools to support the full-lifecycle of an organization’s implementation – Onboarding, Management and Offboarding. ECGridOS provides a set of essential services and functions that enable the effective and efficient use of the ECGrid network directly from enterprise applications, such as an ERP or WMS system, or integration into cloud platforms that support supply chain data exchange.

Strategically, ECGridOS is about increased control and flexibility using “fine grain” (SOAP) APIs for a more interactive programming model than found in “coarse grain” (REST) API integration projects. **The ECGrid approach puts partners, customers, and SaaS platform teams in full control of their implementation.**

***Benefits of a Fine-Grain API Approach***

A fine-grain API approach, where the API exposes a larger number of methods or functions that perform smaller, more specific tasks, can offer several benefits depending on the context and requirements of a particular software development project. Some advantages of using a fine-grain API include:

1. **Granular Control:** Fine-Tuned Operations: Developers have more control over individual components or operations. This level of granularity allows them to fine-tune interactions with the system, performing specific actions as needed.

2. **Flexibility and Customization**: Tailored Solutions: Fine-grain APIs enable developers and ISVs to create more customized and tailored solutions for their customers using ECGridOS. They can choose specific methods relevant to their use case and build solutions with precisely the desired functionality.

3. **Efficiency in Resource Utilization**: Developers can optimize resource usage by invoking only the necessary fine-grain operations. This can lead to more efficient utilization of system resources, such as network bandwidth, memory, and processing power.

4. **Scalability:** Developers can scale different components independently, making it easier to adapt to changing requirements and handle increased loads in specific areas of the application.

5. **Modularity**: Fine-grain APIs promote a modular development approach. Each fine-grain method can represent a module or a specific feature, making it easier to develop, test, and maintain modular code.

6. **Enhanced Debugging and Testing**: Fine-grain APIs allow for more granular testing of individual components, making it easier to identify and address issues. Debugging can be more focused on specific functionalities, simplifying the troubleshooting process.

7. **Diverse Use Cases**: ECGrid’s fine-grain APIs support diverse use cases. Developers can leverage the flexibility of the API to address a wide range of requirements without being constrained by a predefined set of higher-level coarse-grain APIs.

With ECGridOS you can start with a single use case and extend to more advanced capabilities over time. Engineers and ISVs are not limited in their options while automating their B2B data exchange.

Key Capabilities supported by the ECGridOS library to facilitate your B2B data exchange requirements include:

1. **Account provisioning:** ECGridOS allows for full control of account provisioning and management via APIs right from your organization’s infrastructure and systems consistent with your organizational controls and workflows.
	1. **User/Customer Onboarding**:
		1. **Account and Mailbox Creation**: When a new employee joins an organization, account provisioning involves creating user accounts for them in relevant systems. This includes assigning a unique username, password, and any necessary identification details with role-based access controls.
		2. **Role-Based Access**: Account provisioning often aligns with role-based access principles to ensure users have access only to the resources necessary for their job roles and implementation requirements.
	2. **Access Permissions:** Based on the user's role and responsibilities, account provisioning establishes the appropriate access permissions to IT resources. This may include access to email, file repositories, applications, and other services.
	3. **Updates to User Attributes**: Changes to user attributes, such as contact information or job title, may also be part of the account provisioning process to keep user records accurate.
	4. **User/Customer Offboarding**: When an employee leaves the organization or changes roles, account provisioning includes deactivating or disabling their account to prevent unauthorized access. This is a critical step in maintaining security and compliance.
2. **Connectivity:** Use the APIs to create and manage communications to and from your network and external environments (trading partner, service provider, VAN, etc.). ECGridOS allows you to connect to your partners, utilizing all communication protocols and networks available through the ECGrid network.
3. **Transaction Management and Data exchange**: The platform allows users to send and receive data directly from their trading partners and to track the status of EDI transactions/Files, providing visibility into the status of messages as they move through the communication process.
4. **Message Routing**: ECGridOS helps in routing EDI messages between trading partners, ensuring that the right information reaches the intended recipient. The platform also provides rich content support and content routing by supporting various EDI standards and protocols, ensuring compatibility with different trading partners and industry requirements. ECGridOS also allows the configuration of managed file transfer (MFT) for non-standards-based data.
5. **Trading Partner Management:** ECGridOS facilitates the management of trading partner relationships, including the onboarding of new partners and the configuration of EDI communication settings.
6. **Alerts and Notifications:** Users can set up alerts and notifications to be informed of important events, such as the receipt of new messages or the status changes of transactions.
7. **Error Handling/Management:** The OS monitors the system for errors and faults, providing error handling mechanisms to prevent catastrophic failures. It may also log errors and generate reports for diagnostics and troubleshooting.
8. **Security**: Secure data exchange is essential in EDI, and ECGridOS likely includes mechanisms for secure data transmission, authentication, and authorization to protect sensitive business information.

**Quick Reference Notes**

* **ECGridOS Versioning**: The latest version of the ECGridOS API is 4.1.
	+ Version 4.1 API URL: <https://os.ecgrid.io/v4.1/prod/ECGridOS.asmx>
* **Full ECGridOS library of API calls, including full descriptions**: <https://github.com/LorenData/ECGrid-API/wiki/API-Calls>
* **ECGridOS Web Services**: Given the confidential nature of electronic commerce, ECGridOS can only be accessed securely over HTTPS.
	+ ECGridOS is officially a SOAP interface, but can be accessed through HTTP, POST endpoints and allows for *application/x-www-form-urlencoded* content type to be sent.
* Data returned from the APIs can be single values, datasets, objects or collections of objects (returned as arrays).
* **ECGridOS Support**: ECGrid Support can be reached at support@ecgrid.com.
* **API documentation**: Please check out our [Wiki Page](https://github.com/LorenData/ECGridOS_API/wiki) for more information.
* **Full** [**ECGridOS User's Guide**](https://github.com/LorenData/ECGridOS_API/wiki/ECGridOS-User%27s-Guide)

**Getting Connected to ECGridOS**

ECGridOS is a Web Services API built on the Microsoft .NET Framework. It can be accessed through any system capable of making Web Services calls. Any programming language can be used that can make a HTTP or SOAP request. Postman and Visual Studios are just 2 examples of how you can connect to our Web Services API:

*Postman*

Postman is a free API platform for building and using APIs. You can sign up at <https://www.postman.com.>

1. In the upper left-hand corner, click on the Workspaces header, then Create Workspace:



1. In the upper left-hand corner, click on Collections, then on Collections again in the pop-up box:



1. Manually create the new collection by clicking New (in the red box). In the screenshot below, you can see how we manually created the LorenData collection, highlighted in green.
2. Create the actual API calls, (in the gold boxes, below). In the first box, you can see that we have post and then the URL for the ParcelUploadExA. Here you can select the different commands like POST, GET, DELETE, etc. The second box above are the important parameters and headers for the call itself. These must be set up properly to send the call.

In the upper right-hand corner, you will see the Save button. When you have your parameters correct and you get a result set, you will want to save the call to be reused later. For example, when you want to use the same call, you don’t have to remember or look up your API key as it is already saved in the call parameters. Finally, once all the parameters are entered, click the “Send” button. The results will be posted under the parameters.



*Visual Studio*

1. In your Solutions Explorer, right click on references and select "Add Service Reference".



1. On the Add Service Reference window, click the "Advanced" button.



1. On the Service Reference Setting window, click the "Add Web Reference" button.



1. Enter the ECGridOS API URL: https://os.ecgrid.io/v4.1/prod/ecgridos\_admin.asmx



1. Once ECGridOS API is Loaded – “1 Service Found” shows on the right top box. Click “Add Reference” at the bottom.

This will build all the Objects and Create the SOAP Connector for you and you can reference everything like a normal class library.



**Getting Started with ECGridOS**

**First Steps**

To begin using the ECGridOS API, you must first be an ECGrid customer. If you are not currently an ECGrid customer, please reach out to our Sales team at sales@ecgrid.com.

To get started, you will need an API/Session ID. More information on what API Keys/Session IDs are and how to use them in ECGridOS can be found [here](https://github.com/LorenData/ECGrid-API/wiki/API-Keys-%26-Sessions).

Each individual user can find their API key under Access Information in their profile:



Network and Mailbox Admins also have access to all API keys for each user in their applicable network or mailbox. These can be found under Management/Users on the left side navigation panel, then choosing the correct User profile and looking under the Access Information section towards the bottom of the page. By clicking on the eye next to the API key, you can see the key in plain text.



Once you have your API/Session ID, you have access to execute all calls in the full ECGridOS API library.

***Commonly Used Calls – Extended Capabilities***

These calls are standardly used to start the VAN business process flow for most companies. For a downloadable or printable version, please click [here](https://support.ecgrid.com/support/solutions/articles/6000269806-ecgridos-api-business-process-examples).

Extended capabilities are available for customers with a private network node needing to support larger and more complex implementations. The capabilities listed in this document as well as additional APIs can help manage your network, data flow, mailboxes and users.

Many of our API calls have 2 distinct options. API calls with “-Ex” at the end are considered advanced. The basic version without the –Ex is specific to the Network/Mailbox of the API Key being used. However, for Network Admin users that want access and visibility into all the mailboxes, IDs, or other Objects from a single API Key, using the –Ex version is valuable.



**Upload Document - Outbound Calls from your network**

* ParcelUploadExA - Post an EDI document to ECGrid. It will return the ParcelID of the document successfully uploaded. This parcel ID can be used to retrieve additional information.
* InterchangeInfo – Return the metadata about a specific interchange – including the document status.
* InterchangeManifest – returns the audit information which shows the step/process it went through on ECGrid.

**Download Document - Inbound to your network**

Batch Method

* Get list of parcels in network
	+ ParcelInBoxEx – use SessionID (API Key of network Admin), NetworkID, and -1 (ALL) for rest to capture all parcels for all mailboxes.
* Loop through ParcelIDInfo to get each document and get ParcelID value.
* Get document in base64String: Call ParcelDownloadA using SessionID and the ParcelID.
* Call ParcelDownloadConfirm to remove the file from the pending download list.

Callback – Webhook

* CallBackAddEx – Once you create this callback request, it will send a response to the URL provided with the parcelID of the document that arrived in the inbox.
* Get document in base64String: Call ParcelDownloadA using SessionID and the ParcelID.
* Call ParcelDownloadConfirm to remove the file from the pending download list.

**Look up Transactions**

* InterchangeInBoxEx – able to search by date range and interchange control number. The date needs to be in YYYY-MM-DDTHH:mm:ss in UTC format. The times can take an offset as well: Ex – 2023-02-01T00:00:00-05:00 where the UTC offset is -05:00. The InterchangeControlID can be an empty string and ECGridIDTo, ECGridIDFrom and Mailbox can have a -1 to get “ALL”.
* InterchangeOutBoxEx – able to search by date range and interchange control number. The date needs to be in YYYY-MM-DDTHH:mm:ss in UTC format. The times can take an offset as well: Ex – 2023-02-01T00:00:00-05:00 where the UTC offset is -05:00. The InterchangeControlID can be an empty string and ECGridIDTo, ECGridIDFrom and Mailbox can have a -1 to get “ALL”.
* To view the actual document, you can call the ParcelDownload function.

**Document Error Checks**

* CallBackAddEx
	+ Set SystemObject (network, mailbox, parcel)
	+ Set ObjectStatus – Invoke Callback for each parcel status that would need investigation. So, if parcel reaches error status, Callback will hit specified URL to act on a specific parcel. For outbound errors, a Callback would need to be set for the outbox. For inbound errors, a Callback would need to be set for the inbox.
* Call InterchangeManifest for specific parcel to get audit information for the errors. This will show each step/process through the ECGrid network.

*Snippet from return set:*

 <ManifestInfo>

 <ManifestDate>2022-11-09T15:03:33.09</ManifestDate>

 <NetworkID>47</NetworkID>

 <NetworkName>Loren Data Corp. - ECGrid Testing</NetworkName>

 <Type>ECGridOS</Type>

 <ParcelID>1416918926</ParcelID>

 <InterchangeID>0</InterchangeID>

 <StatusCode>M2000</StatusCode>

 <StatusMessage>NETWORK-IN: Parcel Ready for Grid Processing</StatusMessage>

 <StatusColor>blue</StatusColor>

 </ManifestInfo>

**Administrative**

* TPAddEx - Add new IDs to a mailbox. RoutingGroup is mandatory and usually ProductionA
* MailboxAddEx – Add new mailbox to network. User needs to exist before creating mailbox
* To setup an interconnect/Trading partnership
	+ Call TPSearch – this will search the ECGrid network for the trading partner IDs to see if they are currently set up on our network. Make sure -1 is set for NetworkID and MailboxID to search the entire ECGrid Network, not just a specific network node or mailbox. This will need to be called for each set of qualifiers and IDs you want to set up the partnership.
	+ Capture the ECGridIDs (internal UUIDs for the qualifier and IDs)
	+ Call InterconnectAdd with the ECGridIDs captured with the TPSearch function
* Resend Interchange – You need the ECGrid internal Interchange ID to resend an interchange.
	+ InterchangeInboxEx or InterchangeOutboxEx– Search for the interchange you want using one of these calls and the ISA control number.
	+ Find the InterchangeID element value.
	+ Call InterchangeResend with the InterchangeID value.

**Find Trading Partner IDs on ECGrid Network**

* TPFindEX – Search the ECGrid Network by description. -1 on NetworkID and MailboxID will allow a search across all our networks and mailboxes. % at the beginning/end of the description will wildcard the entry.
	+ When using GET, you need to have %25 in front of the ID so it is accepted in the HTTP request. Ex: %25DEMO%
	+ If using POST with the parameters in the body, no need for the %25 in front. Ex: %DEMO%
* TPSearchEx – Search the ECGrid Network by qualifier and ID. % at the beginning/end of the qualifier and/or ID will wildcard the entry.

**ECGridOS Nomenclature**

ECGrid is a system to move electronic business documents from one organization to another in a reliable, efficient and trackable manner. So, when everything else is put aside: It's all about the data.

Using the physical world as a model, all these business documents are wrapped in envelopes. The lowest level that ECGrid penetrates is the Interchange. In X12 this is represented by the ISA envelope and in EDIFACT it is the UNB envelope.

It is our philosophy and policy that nothing within this envelope is within our domain. This area is left to individual businesses, their VANs and ECSPs. This clearly delineates ECGrid from the networks that connect to it. It also allows us to clearly concentrate on our mission.

The terms below are important for you to understand as you work through the API library. More detailed information for each term can be found by clicking on the term.

*ECGridOS Specific Terms:*

[**Parcel**](https://github.com/LorenData/ECGrid-API/wiki/Parcels%2C-Interchanges-%26-Mailbags): Regardless of communication protocol, all files uploaded to and downloaded from ECGrid are considered Parcels. Within EDI itself, data is considered a stream and is not tied to the concept of a file. For this reason, ECGrid uses the term Parcel to describe EDI data turned into a file-based format.

On the ECGrid network, all interchanges will have a unique inbound parcel ID and a unique outbound parcel ID, both of which reference the single interchange.

[**Interchange**](https://github.com/LorenData/ECGrid-API/wiki/Parcels%2C-Interchanges-%26-Mailbags)**:** Also known as an envelope, this is the level where Sender and Receivers are defined. All data is routed on ECGrid and through all Interconnects at the Interchange/Envelope level.

ECGrid natively routes many EDI standard envelopes and can route any data types within an X12 ISA envelope.

[**Network**](https://github.com/LorenData/ECGrid-API/wiki/Networks-%26-Mailboxes)**:** On ECGrid, each Network is identified by an Integer value called the NetworkID. You can use NetworkList to find the Network by name and NetworkInfo to find the name by NetworkID.

A traditional Network, such as a VAN, provides its own routing and mailboxing services to its end users. Many Electronic Commerce Service Providers (ECSPs) also provide their own internal mailbox services. Prior to ECGridOS, the ECGrid Infrastructure only provided Network connections without Mailboxes.

ECGridOS has a complete set of APIs that allows the ECGrid attached Network to create and maintain individual mailboxes within their Network connection.



[**Mailbox**](https://github.com/LorenData/ECGrid-API/wiki/Networks-%26-Mailboxes): Each Mailbox on ECGrid is assigned a unique MailboxID. The default Mailbox for each Network is 0 (zero) called the Root. Traffic can be sent to the individual mailboxes or to the Root mailbox, data flow can be configured individually for each mailbox on a network.

It is also possible that one Network's traffic may be redirected to another network. ECGridOS allows all of this rerouting to be mapped internally. In the event of a redirection, the ECGridIDInfo class shows the intended network in the NetworkID property, and the redirection in the OwnerInfo class.

**Technical Nomenclature**

## **Data Types:** All parameters are required for each web service call. In some cases, default values may be used as indicated when a value itself is optional. All data types are listed in Visual Basic format for simplicity.

**Strings:** Wherever necessary Strings are listed with their minimum and maximum lengths using the following format:

*String(min,max)*

If this String is fixed length, it is listed as follows:

*String(len)*

**Integers**

The following table describes the integer types used throughout ECGridOS.



**Boolean**

Boolean true/false values are used within the system. Your system might interpret these as integer zero (false) and non-zero (true) values or some other way.

**Objects**

ECGridOS makes extensive use of Objects to return complex information, and to guarantee a consistent response format. Please familiarize yourself with the class structures used throughout ECGridOS.

## *Uploading & Downloading Files*

When uploading and downloading files Byte Arrays or Base64 strings are used. This allows for easy programmatic data sending and receiving without having to use the file system if you so choose.

**Commonly Asked Questions**

* Does the ECGridOS API use SOAP or REST? ECGridOS allows you to connect using either SOAP Protocols or "REST Like" HTTP POST form encoding protocols. For a more thorough explanation of our philosophy, please click [here](#Bookmark1).
* *How do I upload a document?* To upload a document using ECGridOS, you will use the following API calls:
	+ ParcelUploadExA - Post an EDI document to ECGrid. It will return the parcelID of the document successfully uploaded. This parcel ID can be used to retrieve additional information.
	+ InterchangeInfo – Return the metadata about a specific interchange – including the document status.
	+ InterchangeManifest – returns the audit information which shows the step/process it went through on ECGrid.
* *How do I download a document?* There are 2 ways to download documents. You can use a batch method or Callback/Webhook event
	+ Batch Method
		- Get list of parcels in network
			* ParcelInBoxEx – use SessionID (API Key of network Admin), NetworkID, and -1 (ALL) for rest to capture all parcels for all mailboxes.
		- Loop through ParcelIDInfo to get each document and get ParcelID value.
		- Get document in base64String: Call ParcelDownloadA using SessionID and the ParcelID.
		- Call ParcelDownloadConfirm to remove the file from the pending download list.
	+ Callback/Webhook
		- CallBackAddEx – Once you create this callback request, it will send a response to the URL provided with the parcelID of the document that arrived in the inbox.
		- Get document in base64String: Call ParcelDownloadA using SessionID and the ParcelID.
		- Call ParcelDownloadConfirm to remove the file from the pending download list.
* *How do I search for trading partner IDs?* There are multiple ways to search for a Trading Partner ID through ECGridOS. You can search on the Company Name/Description or you can search by Qualifier and ID.
	+ Search by Company Name/Description
		- TPFindEX – Search the ECGrid Network by description. -1 on NetworkID and MailboxID will allow a search across all our networks and mailboxes. % at the beginning/end of the description will wildcard the entry.
			* When using GET, you need to have %25 in front of the ID so it is accepted in the HTTP request. Ex: %25DEMO%
			* If using POST with the parameters in the body, no need for the %25 in front. Ex: %DEMO%
	+ Search by Qualifier and ID
		- TPSearchEx – Search the ECGrid Network by qualifier and ID. % at the beginning/end of the qualifier and/or ID will wildcard the entry.
* *How do I find an ICN, (Interchange Control Number)?* You can easily search both inbound and outbound transactions on your network. The format and call order are the same, but the initial calls are slightly different:
	+ Search for inbound data to your network/mailbox
		- InterchangeInBoxEx – able to search by date range and interchange control number. The date needs to be in YYYY-MM-DDTHH:mm:ss in UTC format. The times can take an offset as well: Ex – 2023-02-01T00:00:00-05:00 where the UTC offset is -05:00. The InterchangeControlID can be an empty string and ECGridIDTo, ECGridIDFrom and Mailbox can have a -1 to get “ALL”.
	+ Search for outbound data from your network/mailbox to your trading partner.
		- InterchangeOutBoxEx – able to search by date range and interchange control number. The date needs to be in YYYY-MM-DDTHH:mm:ss in UTC format. The times can take an offset as well: Ex – 2023-02-01T00:00:00-05:00 where the UTC offset is -05:00. The InterchangeControlID can be an empty string and ECGridIDTo, ECGridIDFrom and Mailbox can have a -1 to get “ALL”.
	+ To view the actual document, you can call the ParcelDownload function.

**Basic Capabilities for Mailbox Users**

Basic capabilities to access a single mailbox on ECGrid are also available. For more information on ECGridOS’s basic capabilities, can be found here.