Roadmap 2020/2021

Status: Approved by TSC vote

Document Purpose

The purpose of this document is to collect various ideas for inclusion in the Fledge roadmap; these ideas can then be presented to the TSC for discussion and prioritisation into a roadmap for Fledge for 2020/2021. Other input is requested to extend the set of items for possible inclusion.

Once this document has been reviewed and discussed at the TSC meeting each potential item will be given an ID and the voting members of the TSC will be asked to give a priority on the item for inclusion. This will be used to determine what items become part of the roadmap for the coming year and which are excluded. It does not guarantee inclusion within Fledge within that year as this is dependent on resource availability. It is also recognised that specific requests from users may interrupt the roadmap progress and introduce new additions or modifications to Fledge that are not part of the roadmap.

Scope

A reminder of the scope of Fledge is included here to help bound and direct the ideas proposed for inclusion in the roadmap.

- The major purpose of Fledge is an Industrial IoT platform, enhancements should be targeted to use within an industrial setting.
- Fledge is not intended to replace control systems. The real time, safety critical control is outside of the scope of Fledge.
- Fledge should be a hands free appliance that may be run in unattended environments.
- The intended location of operations is within edge devices, it is not precluded that Fledge instances may be run in more powerful hardware, however enhancements should not require hardware more powerful than an average specification edge gateway.

Development Goals

The roadmap items are broken down into a number of groups; core Fledge enhancements, the various plugins, the user interface, documentation and deployment environments.

Core Enhancements

Device Updates via South Service

This is the ability to update settings within a device via a south service and the associated plugin. It is not real time device control, this is outside of the scope of Fledge. Rather this is the ability to make changes within a device to alter the operating characteristics of that device.

This may be triggered either by an external input via the north connection or the Fledge API, or by an internal event within the Fledge instance. Such as a notification rule triggering causes a change to the device.

Watchdog Functionality within notifications

Currently rules can only be triggered by data that has come from a south device. It is not possible to have a rule that is triggered due to the absence of data. This means the notification mechanism can not be used to detect when a device stops sending information to the Fledge edge device.

Notification of internal events

The notification system is based on the data coming from events, however there are cases when an internal event may be required to trigger action on the edge device. Such events might be

- Failure to connect to north systems
- Storage thresholds being crossed
- Fledge shutdown requests
- Service shutdown requests
- Service failures
- Administration changes

High Availability Improvements

The addition of hooks to allow better control of highly available Fledge pairs to support both primary/secondary pairings of Fledge instances and shared load instances. Requests for more complete and integrate high availability functionality than is currently present with Fledge have been growing.

Simply having high availability by implementing primary/secondary pairs is not the total solution to availability within an application like Fledge, there are situations where you can not move functionality from one edge device to another, e.g. where sensors are directly connected to the edge device. Fledge already has internal monitoring capabilities that are used to restart individual micro services if they should fail, it is crucial that this is in place to provide the best possible changes of maintaining functionality when a failover to another edge device is not practical because of the sensor connectivity.

The following set of options defines a high availability solution with descending levels of service and descending complexities. Any of these may be acceptable to an end user in differing circumstances.

Fledge Failover without Data Loss

This offers the highest level of protection but is also the most difficult to implement. In order to do this with any degree of certainty it is highly probable a clustered or highly available data storage backend will be required. Currently we can provide this by means of a PostgreSQL highly available pair. It is limited to having a single active Fledge in the highly available pair, with the secondary Fledge starting from cold and attaching to the data store on failover.

Adding the ability to have the secondary Fledge in a warm standby mode is the next level of refinement in the roadmap for development. This would allow for faster recovery when the failure is detected.

The next level of refinement is to allow both Fledge to be active, i.e. an active/active pair rather than a primary /secondary pair. However this will need greater integration of the HA failover service into Fledge itself to allow the Fledge to understand which services are operating on which cluster node.

Fledge Failover with Minimal Data Loss

In this form of a Fledge cluster data is sent from one Fledge to another in order to allow unsent data to be preserved during a failure. There is a small window of data that may be lost due to the latency between the Fledge pair.

Currently this form of high availability is difficult to configure, requiring knowledge of which data is ingested by which Fledge device, however it does allow for active/active pairs to be configured. The aim of this roadmap item is to make the configuration of such a Fledge pair considerably simpler, automating what is currently a complex and not well documented procedure for configuration of the Fledge pair.

Fledge Failover without Data Preservation

This is a slightly lower level of protection and allows data to be lost when a failure occurs. Unsent data that is buffered in the failed Fledge node will be lost during failover. This has the advantage that a clustered or HA data store is not required and the lighter weight, higher performance SQLite data store can be used. Currently it is limited to having a single active Fledge in the highly available pair, with the secondary Fledge starting from cold when the failure is detected.

Adding the ability to have the secondary Fledge in a warm standby mode is the next level of refinement in the roadmap for development. This would allow for faster recovery when the failure is detected.

The next level of refinement is to allow both Fledge to be active, i.e. an active/active pair rather than a primary /secondary pair. However this will need greater integration of the HA failover service into Fledge itself to allow the Fledge to understand which services are operating on which cluster node.

Always on North Service

Currently traffic north can only be scheduled via tasks, there is no option for a continually running service that sends data immediately. This may be simulated by running a frequent sending process schedule, however it relies on polling the storage service for data availability. A permanent north service that is passed the data as it comes into Fledge would be more efficient and reduce latency in situations where an always on connection to the north destination was available.

Flexible Policy Based Data Retention

Currently we implement a blanket data retention policy for a single Fledge instance. This does not allow us to flag some data as being more important or having a longer lifetime than other data. We should consider providing a retention policy per asset that can override the default retention policy. This would allow

- Data that changes infrequently to be retained for longer to allow reference to it.
- High Bandwidth data to be purged more frequently in order to prevent data building up and overwhelming the available storage in the gateway.
- Data that has the most value to be retained for longer to allow for failures in communications over longer periods of time without running out of storage space.

User Interface

Extensibility Features

The Fledge core services are designed to support extensibility of Fledge through the addition of plugins for various features; north, south, storage, rules and notification deliveries. However the user interface has no such ability. It would be logical to also allow the user interface to be extended to support new services and have specific support for plugins.

Service Support

Allow new GUI modules that support specific Fledge services that may be added in addition to those in the current set; core, south, north and notification.

Visualization Support

Allow for plugable, external visualization to be added for all or particular assets, giving flexibility to show specific data using more applicable visualization techniques.

Plugin Support

Allow for specific user interface to be introduced to configure and manage the features of specific plugins. All plugins should still be configurable or manageable by the standard user interface, this feature merely allows for a richer user experience for specific cases.

Asset Tracking Visualization

Fledge maintains data regarding the path taken by each asset code within the system, however there is currently no mechanism for a user to visualize this data within the user interface. This should be added.

Storage Plugins

There are currently no plans for any development of new storage plugins, work has recently been undertaken to improve the performance of the SQLite plugin, similar work cups be undertaken for the PostgresQL plugin if the SQLIte work shows it would be beneficial.

South Plugins

The addition of new south plugins tends to occur because of particular need for a PoC or user installation of Fledge. Some have and will continue to be implemented for more strategic reasons or to reinforce the industrial nature of Fledge. Those more opportunistic south plugins can not easily be added to a long term roadmap, however there are a few directions it is clear that we will be investing resources into in the future and those are outlined here as general areas in which plugins will be written rather than specific protocols or devices to be supported.

Electrical Generation

Following conversations both within the LFEnergy group and elsewhere it is clear that a set of International Electrotechnical Commission **(**IEC) protocols and IEEE 2030.5 are important in this group of potential users. We currently have one ongoing contribution in development for IEC-104 and it is very likely that other IEC and IEEE protocols related to the energy sector will be implemented within the next year.

Programmable Logic Controllers

PLC’s are an important source of data, and an area where we will be wanting to set data back into devices if we follow the route outlined above for writing data from the south plugin. This will require work on PLC’s plugins that exist and it is likely that new PLC’s will be added to the supported list.

North Plugins

North plugins, similarly to the south, can be implemented as a result of user requirements, however there are some more strategic examples of north plugins that we have already discussed to help engage more users and enlarge the Fledge community.

Cloud Services

Using an open source platform such as Fledge to capture data and send it to cloud services is attractive to end users as it prevents them being locked into a specific cloud service by giving flexibility on the data collection side.

Amazon Web Services

Some discussion has already taken place regarding adding a plugin to send data to Amazon Web Services and this is clearly a strategic direction that we should consider pursuing.

Microsoft Azure

Although less advanced than the AWS interface mentioned above this is one that has been mentioned in multiple conversations with potential Fledge users and should also be taken into consideration for inclusion within the roadmap.

IBM Watson Cloud

Addition of the IBM Watson IoT core in Ibm’s cloud service as a north plugin to Fledge is also considered a strategic objective of the Fledge project.

Filters

There are no particular filters that are in the roadmap currently, there are thoughts of general direction of the filters that will be added;

- Signal processing filters that can be used to remove noise, or extract features from raw sensor data.
- Machine learning filters.

Notification Service

The current notification service is limited to a single delivery channel per notification event. This leads to multiple notifications benign configured with the same rule conditions to allow delivery to multiple destinations. This is wasteful and can be eliminated by allowing more than one notification delivery channel to be defined per notification.

Rules

Currently we have no roadmap items to add new rules. Rules will probably be added as and when required to satisfy requests from users.

Notification Delivery

Currently we have no roadmap items to add new notification delivery plugins. New delivery plugins will probably be added as and when required to satisfy requests from users.

Documentation

Although efforts have previously been put into improving the documentation of the Fledge projects there are still areas where it is not as full as it should be and requires more work to be put into bringing it up to date and complete.

Developer Guides

We have developer guides for some plugins types but by all means not all. We also do not cover all aspects of developing plugins. In particular the following areas should be improved;

- Asynchronous South Plugins - although polled plugins are well covered asynchronous plugins are not as well explained and documented.
- North Plugins - Both C++ and Python plugins need to be more fully documented. The current documentation is incomplete in a number of areas.
- Storage Plugins - We currently have no documentation for anyone wishing to write a new storage plugin.
- Filter Plugins - Considerably more detail is required for filter plugins.
- Notification Rule Plugins
- Notification Delivery Plugins
- Persisting data from Plugins - a mechanism exists to persist data from within a plugin alongside the usual Fledge configuration data. This needs to be added to the development documentation.

API Guides

Fledge currently has a REST API guide that covers 80% of the API, it should be included to cover as close as possible to 100% of the API and enhanced to include more examples and details of the API entry points.

User Guides

The current user guide is limited to the basic information to get started with Fledge itself and outlines of each of the Fledge plugins. It does not adequately cover more advanced topics or the philosophy of Fledge that needs to be understood to fully utilise all the features of Fledge.

Documentation will be produced to cover the following areas

- Security of data and data flows
- Configuration of complex processing pipelines
- Notifications and interactions between services
- Configuring high availability Fledge clusters
- Multiple north services

LFEdge Project Integrations

Integration with others projects within the LFEdge group of projects where appropriate.

EdgeX

Work is already underway to provide a mechanism for sending data from Fledge into the EdgeX core. Likewise there is a project within EdgeX that will enable data to be sent to Fledge by EdgeX.

OpenHorizon

Discussions are underway with members of the Open Horizon project to enable greater integration. In consultation with the Open Horizon TSC a phased approached to Fledge/OpenHorizon integration is proposed. The first phase of this being to implement the deployment of a Fledge instance using Open Horizon.

SDO

Discussions are at an early stage to determine how the SDO functionality can be used within a Fledge deployment. These discussions are set to continue and roadmap items will emerge as these discussions mature.

Akraino

Two Akraino blueprints have already been developed that feature Fledge in particular scenarios, others are in discussion, although no concrete examples have yet been defined. It is highly likely the collaboration with the Akraino project will continue as and when these are identified.

Eve

Integration with Project Eve has already been demonstrated and there are currently no roadmap items related to this work. However it is likely as both projects move forwards there will be requirements in the future.

Other LFEdge Projects

Currently there are no integration plans with other LFEdge projects.