currently being done by the technician is sent to the device.

Another potential cause is a join in a sync rule that was believed to be more restrictive than it is e.g. a link to the parts for a task that attempts to include tasks only at request level but then also selects all parts on all tasks related to that request. These will show as much larger individual transactions. A good step to confirm would be to revert to the FSM standard rule if it is different to see if this improves the performance and message size. After which the sync rule script would need to be redone to only select the data needed and no more than that. Extra filters usually correct this.

The next steps should only be performed by an experienced DataBase Analyst with good knowledge of the FSM data tables.

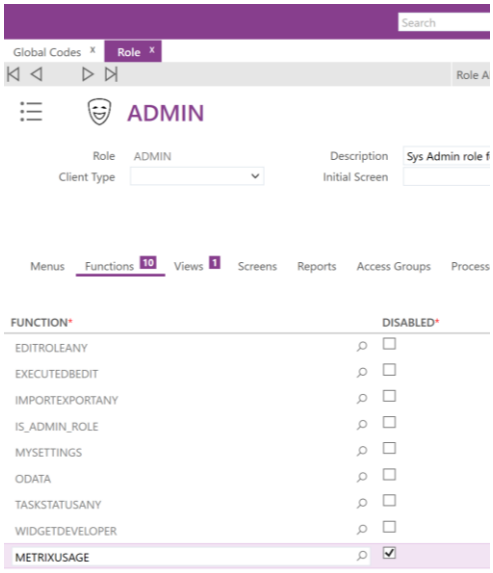
In each of these cases, the problem records will likely need to be removed from the message queue to allow the proper syncing to resume. This should be done during down time on the database to avoid errors. This should only be done after the relevant fixes are in place to prevent an instant reoccurrence of the issue.
Only remove the queued records from the table shown to be the issue. Once these are removed, the device should be allowed to sync the remainder of their queues. This will ensure no responses on work in progress are lost from the mobile to the database.
When a user's mobile device shows a clear queue for sync on both the device and the database then the user can re-initialise to ensure all relevant data is on the device.
These initialisations should be staggered to avoid excessive load and only affected users need do this.

Audits and Extracts

It is worth checking if any audits and/or extracts have been setup and if these are active. If there are then confirm if there are constraints in place or they are looking at tables with low activity. An unconstrained audit/extract on a high Activity Feed can cause an impact similar to the activity feed where all work is doubled with the system trying to record all the detail of every action whilst also actioning it.
There are also alternatives such as using notification messages or reporting just as with Activity Feed to avoid the area all together.

MetrixUsage Function

Check the roles in the system for the METRIXUSAGE function



If this is found on any user role remove it. This function is designed for background tasks and adding it to any user role will steadily increase the database size until it impacts performance. Some of the system data tables may need to be cleared following this so a DataBase Analyst should be engaged to review large tables and archive. This should only be done by an experienced DBA.

Global Codes

Global codes are loaded into memory so, if they grow too large, they can cause a slow down of the system. Usually global codes would have below 100 entries so any in the multiple hundreds and up should be checked. Special attention should be paid to those codes that have hierarchies attached, especially if the hierarchy is to another large table. If a code is found with this type of hierarchy then start by disconnecting the hierarchy and see if this improves performance. After that review the codes to

check if all are required or if they need to be global code.
The other code tables are not loaded into memory, only global codes, so they should not need to be checked.

Configuration Changes

Performance usually drops after some type of change. If all configuration changes are being monitored and well recorded this should just be an issue of looking up what changes were made at the time and reverting to confirm improvement. In situations where no such record is being kept, more investigation is required.

Lobbies

Lobbies can be very useful for reporting but depending on how they have been setup they can be very resource intensive. Identify any lobbies that are non-standard. The quickest way to confirm them as a source of the issue is to temporarily disable them and see if performance improves. If they are the issue, then their setup will need to be re-investigated to determine if a less frequent update or a more restricted dataset can achieve a suitable result.

Business Rules

Business Rules can sometimes cause looping when not setup ideally which will impact performance. This should be characterised by the performance dropping when loading specific screens rather than starting as soon as the application is started, at specific times or seemingly random occurrences.
The screen that triggers the performance drop will help direct to the business rule at fault but even without that there are some ways of tracking the Business Rule down.
The first would be to run a Business Rule trace as described in the Server Log section of this guide.
The other would be to manually deactivate all rules and then, if performance improves, reactivate one by one until the drop is seen.

When the Business Rule is located, check that the output of the rule causes the Business Rule trigger to no longer be valid e.g. if the rule triggers off of a field value ensure that the rule sets this to a different value on completion.
If the rule output is to run XML script then this will need to be reviewed, ideally change this to be a standard output rule and create multiple linked rules if required rather than running the script if the issue with the script cannot be identified.

Database Indexing and Structure

Sometimes the issue can come down to database fragmentation over time or database setup and configuration not being updated to match changes in the data use.

Fragmentation

This next step must be performed by an experienced DataBase Analyst

There should be a regular reindexing of the FSM database scheduled to run during downtime. Check that one is setup and active. If one is not, then agree and arrange one to run on some regular schedule.
The IFS guidance is that any table with over 75% fragmented should be reindexed.

New Indexes

In some cases, it may be beneficial to introduce new indexes to account for changes in use of some of the fields in FSM. The main example of this would be user_def fields used in join or query constraints as user_def fields are not indexed as standard but if heavily used, creating an index could offset some of the resource use. This needs to be a case by case decision made by an experienced DataBase Analyst though and not a decision to be made lightly.

Large Tables

Check the database tables to ensure none are growing beyond an acceptable size.