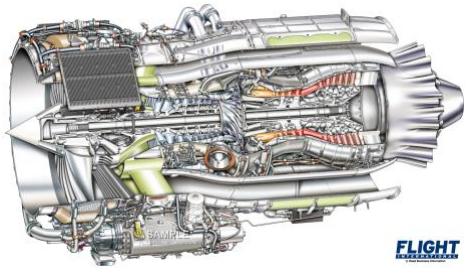
CS486C – Senior Capstone Design in Computer Science

Project Description

Project Title: Connected Engine Data System Administrative Portal	
Sponsor Information: Honeywell	**Harlan Mitchell,** Systems Technical Manager HTF7K Controls Systems Integration https://aerospace.honeywell.com/en/products/engines/htf7000-turbofan-engine https://en.wikipedia.org/wiki/Honeywell_HTF7000 https://youtu.be/E8RlNXnyIgE Honeywell Harlan.mitchell@honeywell.com 602-206-7223

Project Overview:

Honeywell is known for small to mid-sized engines including auxiliary power units (APUs), propulsion engines, and turbo-chargers (note: vehicle turbo-chargers are similar in some ways to gas turbine engines). Honeywell is the largest producer of gas turbine APUs found on many leading aircraft with more than 100,000 APUs produced and more than 36,000 in service today. Within the propulsion engines group Honeywell has applications on helicopters, business jets, turbo props, military jets, and even the US Army Abrams Tank.



Within the Honeywell engines enterprise, our group, the Controls Systems Integration (CSI) group, is responsible for all aspects of engine control including the ECU (Engine Control Unit). In addition to the ECU itself the control system is comprised of temperature/pressure sensors, valves, the fuel pump, and other hardware components. Our most recent project is the re-application of the HTF7000 engine on the new Cessna Citation Longitude business jet which was certified on August 25th 2017. The Cessna Citation Longitude site is: http://cessna.txtav.com/citation/longitude

The Problem

The ECU saves trending and maintenance data in non-volatile memory (NVM) during normal operation. This data is then downloaded by maintenance personnel either on a routine or as needed basis. The current method for doing the engine download is to connect a laptop to the aircraft engine maintenance port (usually in the cabin) using a 4 port RS-422 USB device and a cable. The user then uses software called EEI (Electronic Engine Interface) to do the download and review the data. There are two obvious drawbacks of this system: a) A hard-wired connection is needed for maintenance personnel to physically access the cabin, plug in a laptop, and initiate the data download and b) Data is not collected very often leading to missed maintenance opportunities and an inability to use big data to trend different aspects of the engine.

Honeywell is currently developing a connected engine product called CEDAS (Connected Engine Data Access System) to allow engine downloads to be completed autonomously with the data uploaded wirelessly (Wi-Fi) to the cloud where it will then be accessible remotely. The product will be hosted on a small embedded computer located in the aircraft along with the Wi-Fi antenna. When fully deployed, CEDAS will be installed on approximately 1,200 aircraft flying all over the world automatically downloading data and sending it to Honeywell via Wi-Fi. The problem is that if we simply don't get

data “on schedule” from a particular aircraft, it’s difficult to know what’s wrong; it’s possible that the aircraft is grounded, currently unused, flying to locations that are not configured…or something is broken. If we invent and put in place additional information infrastructure we can mitigate the impact of this problem. This is where this Capstone project comes in.

The Envisioned Solution

In order to completely address this information gap we know when and where each aircraft *has landed*. If we could somehow integrate this external data on aircraft landing disposition, then it could be known whether an aircraft is active (not grounded), where it is flying to, and when it should arrive (and come into Wi-Fi range to report). Knowing the destination will also allow us to recognize when an aircraft has landed at an airport not yet configured for Wi-Fi access…which would explain a failure to report data. More generally, knowing what the status of data reporting for various aircraft is could help gracefully initiate special maintenance action, e.g., if an aircraft has arrived at a destination but not reported then personnel can be notified to connect to the product with Bluetooth and check the device status.

Note on privacy: We will not violate any laws and do not intend to track or follow aircraft. Our only intention is to understand when and where connectivity should and does take place. We do not want to violate our customer’s trust/privacy by exposing their location data.

What we envision is a server application with a secure web-based GUI interface that we can call XYZ (Capstone team to name). Although inter-related, the functions of the envisioned XYZ can be divided into several modules, each centered on a different task.

Part 1: Landing status monitor

Aircraft of interest will be registered with XYZ using the tail numbers (unique id for aircraft); the module’s dashboard shows the status of all landed aircraft, and notifies us of when an aircraft completes a flight but does not upload data within 24 hours. Designing this module must begin with a feasibility study to determine if public flight databases (e.g. FlightAware) can be used/accessed to update aircraft flight information (initial analysis indicates it is feasible but some aircraft may be blocked).

Part 2: WiFi auto configuration manager

An “operator” is a company (NetJets, some private company, etc.) that is operating an aircraft with Honeywell engines. Different operators visit various airports and, in order for Wi-Fi downloads to occur, their CEDAS Wi-Fi connection must be configured with the unique Wi-Fi access codes for that destination…otherwise they can’t connect. In many cases, we may have a situation where Operator A has configured a Wi-Fi connection at a fixed-base operator (FBO) and another Operator B visits this same FBO but a download is not completed due to Operator B not having the correct Wi-Fi configuration for that FBO. What we would like is, knowing Operator B visited a location with known Wi-Fi, to auto-configure operator B’s Wi-Fi modules with the FBO credentials so that CEFAS data can be collected at that FBO. To do this, XYZ needs to populate a database with the exact location of where downloads are taking place. This needs to be by both the exact LAT/LON and the airport code (translate LAT/LON to airport code). This will allow us to auto-configure the Wi-Fi connection details for any aircraft having visited that location without any manual configuration by customers needed.

Part 3: Customer connection information module

Using the data from part 2, we can make a world map that, when you click on an airport, it will tell customers where to physically stage their aircraft *at that particular airport/FBO*, so they can get a Wi-Fi signal that is adequate for connection and CEDAS data download. In essence, this amounts to “crowd-sourcing” connection information to build an evolving Wi-Fi coverage map for every airport: each aircraft records the precise LAT/LON where the engine shut down and started, indicating that signal is good from there; the evolving shape of these data points shows the coverage area. Aircraft may also be able to report the signal strength of their connection, allowing stronger/weaker areas to be mapped (may be tricky due to environmental factors). Ideally, this module’s GUI would use a Google

maps satellite view of the airport, overlaying green/orange areas to denote where Wi-Fi signal strength is adequate/marginal.

Sample LAT/LON data will be provided but the remaining data will need to be generated using a test harnesses that provides/simulate the outside data infrastructure that exists at airports, etc.

Knowledge, skills, and expertise required for this project:

As with all projects, the team will be expected to learn the knowledge and skills required for this project early on. Beyond standard senior-level capabilities in programming and software design, helpful skills will include:

- Acquire basic familiarity with CEDAS and the nature of ECU monitoring.
- Familiarity with modern Web2.0 development, including applicable languages and frameworks.
- Configuration of back-end server with database and other server-side services.
- Skills in interface design, testing and refinement.

Equipment Requirements:

No special equipment should be required beyond a standard development platform (your laptop), as well as freely available environments and software tools.

Software and other Deliverables:

1. System and software requirements document. This requirements document will require extra effort beyond the typical Capstone Project class deliverable. The Capstone Team will need to learn how to develop and capture good requirements. Honeywell will provide training material.
2. Software implementing the functionalities outlined above.
3. Complete professionally-documented codebase, delivered both as a repository in GitHub, BitBucket, or some other version control repository; and as a physical archive on a USB drive.
4. A strong as-built document that details the design and implementation of the software. This must be robust enough to allow a future development team to easily pick up where you left off.