Feb 7, 2019

Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, BC, V5A 1S6

RE: ENSC 405W/440 Requirements Specifications for Distributed Computing Network

Dear Dr. Rawicz,

Optigo has approached Oakion Systems to help solve a inevitable computational and bandwidth bottleneck on their Visual BACnet SaaS product. At a high level, Optigo strategically placed packet sniffers (low performance Linux based processors) in customer local networks, where the packets are sourced and aggregated to a central cluster in the cloud. This results in their data rate being roughly 8GB/day and growing. Optigo is concerned that without changing the way they collect and process their data, they will potentially face extreme overhead.

Oakion Systems is proposing a Microservice solution to be a middleware in-between the capturing of the packets through previously deployed embedded system capture tools on the client side, and Optigo Networks visual BACnet service on the cloud. The middleware will pre-filter the data and complete analysis and aggregation in an optimized manner that will be robust and scalable, reducing computational and bandwidth congestion for Optigo Networks. The attached document outlines the proposal for a software design and implementation for Optigo.

Oakion Systems will divide out the necessary modules and tasks to 5 talented engineering students: Justin Singh, Tony Tan, Shawn Wang, Swimm Chan, and Aaron Nguyen. These individuals come from a strong background in software development, quality assurance and systems design. Oakion is confident that the task at hand will be delivered with success and confidence to solve Optigo's request.

Thank you for reviewing our proposal. If you have any inquiries regarding the proposal, please contact Justin Singh.

Sincerely,

Justin Singh.
CEO
Oakion Systems

Enclosed: Requirements Specifications for **Distributed Computing Network**

Requirements Specification

Distributed Computing Network

flexible scalable robust

Project Members:	Swimm Chan
	Aaron Nguyen
	Justin Singh
	Tony Tan
	Shawn Wang
Contact Person:	Justin Singh
	jksingh@sfu.ca
	250-961-3527
Submitted to:	Craig Scratchley (ENSC 405W)
	Dr. Andrew Rawicz (ENSC 440)
	School of Engineering Science
	Simon Fraser University

Issue Date:	Feb 7, 2019

Abstract

This document outlines the requirement specifications for the Distributed Computing Network project provided by Oakion Systems to Optigo Networks in accordance with ENSC 405W/440. The project comprises a complete software systems solution which will optimize Optigo's computational and bandwidth bottleneck issue that are likely to arise.

Oakion Systems' Distributed Computing Network leverages the existing embedded systems infrastructure deployed with Optigo Networks, to form a Cluster Computing Distributed System on LAN that provides a Microservice solution. Computing nodes (Capture Tools) produce packet data, perform pre-filtering, and strategically split the packet stream, before sending to the Data Stream Management System (DSMS) hosted through a cloud service provider. DSMS then publishes tasks to available subscriber nodes to complete analysis and aggregation. The end result is then sent to Optigo's server for visualization. This process solves the bottleneck issues that currently face the centralized client-server model.

Table of Contents

List of Tables

List of Figures

Glossary

ADT Abstract Data Type, theoretical data type that is largely defined by the operations and work on it and the limitations that apply where a data type is defined by its behavior from the point of the user of the data.[13]

ARM Advanced RISC Machines Architecture makes 32-bit and 64-bit RISC multi-core Processors, designed to perform a smaller number of types of computer instructions so that they can operate at a higher speed, performing more millions of instructions per second.[13]

APM Application Performance Management monitors and manages performance and availability of software applications that strives to detect and diagnose complex application performance to maintain an expected level of service.[13]

API Application Programming Interface, is a set of protocols, routines, functions and/or commands that programmers use to develop software or facilitate interaction between distinct systems.[13]

BACnet Communications protocol for Building Automation and Control networks that leverage the ISO 16484-5 protocol.[12]

CPU Central Processing Unit, To control instructions and data flow to and from other parts of the computer, the CPU relies heavily on a chipset, which is a group of microchips located on the motherboard.[13]

Cloud Storage is a cloud computing model in which data is stored on remote servers accessed from the internet, or "cloud." It is maintained, operated and managed by a cloud storage service provider on a storage servers that are built on virtualization techniques.[13]

DSMS Data Stream Management System, a software that takes in data and converts various kinds of data into a single storage container, or aggregates diverse data into a consistent resource, such as a database. [14]

IEEE Institute of Electrical and Electronics Engineers, an organization that provides professionals with a technical community that keeps integrity globally. [15]

ISO International Organization for Standardization provides world-class specifications for products, services, and systems, to ensure quality, safety and efficiency. [16]

IP address Internet Protocol address, is a logical numeric address that is assigned to every single computer, printer, switch, router or any other device that is part of a TCP/IP-based network.[13]

IoT Internet of Things, a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data through network without requiring human-to-human or human-to-computer interaction.[8]

MSTP Master Slave Token Passing, a token passing protocol that favors BACnet is mainly used for connecting devices to controller, routers, or control applications.[13]

Microservice Architecture An Architectural style that structures an application as a collection of services that are highly maintainable and testable, loosely coupled, independently deployable, organized around business capabilities.[13]

Monolithic Architecture A software design concept where the system is designed to be self-contained; components of the system are interconnected and interdependent.[13]

Pcap Data Packet Capture consists of an application programming interface for capturing network traffic.

P2P Peer-to-Peer architecture, a computer networking architecture which each workstation, or node, has the same capabilities and responsibilities.[13]

REST Representational State Transfer, a software architectural style that defines a set of constraints to be used for creating web services. RESTful web services, provide interoperability between computer systems on the internet.[13]

SSL Secure Sockets Layer, a cryptographic protocol that enables secure communications over the internet.[13]

TLS Transport Layer Security, a cryptographic protocol that provides end-to-end communications security over networks and is widely used for internet communications and online transactions.[13]

x86 Architecture An instruction set architecture series for computer processors that primarily handles programmatic functions and provides services, such as memory addressing, software and hardware interrupt handling, data type, registers and input/output (I/O) management.[13]

1 Introduction

Background

The prevalence of the IOT has presented an opportunity for Oakion Systems to manage the inundation of data that companies like Optigo face. The amount of data that flows to Optigo exponentially increases as their client base grows. This growth will cause a computational and bandwidth bottleneck in the near future (~8GB/day currently). Without changing or improving their current architecture, Optigo will face issues with continuous analytics and performance.

To date, optimization for these purposes are typically done through hardware acceleration at the centralized server [1]. Oakion Systems' Distributed Network solution differs by focusing on a primarily software solution to this issue. This solution allows a less expensive and scalable alternative to clients, and empowers them to maximize their software capabilities before justifying changes at the hardware level.

Currently there are few generic systems dealing with this issue on the market. Larger companies have integrated their own software solutions to fit their own internal needs, but the value of Oakion Systems' product comes from its ability to not only be lightweight and cross platform (working on both x86 and ARM architectures), but also the security measures in place that guarantee all data will be encrypted in transit and no data will be stored on an edge node in our system.

The model of Microservice solution that will be provided by Oakion Systems is an event based one [2]. Our system will not require a response to, or acknowledgement of, every communication that occurs. This allows the system to consist of modules that can be fully asynchronous allowing higher performance and faster computation. Our solution will be made up of 5 modules: a Producer, Splitter, Consumer, Reducer and a DSMS centralized queue that will each provide functionality toward the common goal of optimization. A high level diagram of this functionality is outlined below in Figure 2.1.

Scope

This document outlines the functional requirements of the distributed network solution provided to Optigo networks by Oakion Systems. It will also highlight sustainability and safety specifications of the solution and its compliance with Engineering standards. The functional requirements and specifications will give a system overview of each component and their application. For a more broad overview of the solution, please refer to the high level design that illustrates the architecture of the system in a production system.

Requirements Structure

Oakion Systems have outlined expected timeline for individual requirements as follows:

1. Proof of Concept Prototype - Requirements will be executed by April 2019
2. Engineering Prototype - Requirements will be executed by August 2019
3. Final Prototype - Requirements will be executed past August 2019

To specify a requirement, the document will use the following scheme:

REQ [Section].[Module Sub Section].[Requirement Number]-[Code Scheme]

Code Scheme	Description
PoC	Proof of Concept Prototype
EP	Engineering Prototype
FP	Final Prototype

NOTE: All requirements dates may be subject to change.

Table 1.1: Code Scheme

For example, the fourth requirement in section 2.6 that will be included in the Engineering Prototype would correspond to

REQ 2.6.4-EP Insert Requirement Description Here

2 System Analysis

The Distributed Computing Network consists of five main modules:

Module 1: **Producer** - Collects original packets from smart devices
Module 2: **Data Stream Management System (DSMS)** - Stores all data for centralized access
Module 3: **Splitter** - Strategically breaks up incoming packets through packet mapping
Module 4: **Consumer** - Processes related packet chunks
Module 5: **Reducer** - Combines the processed packet chunks through packet reducing

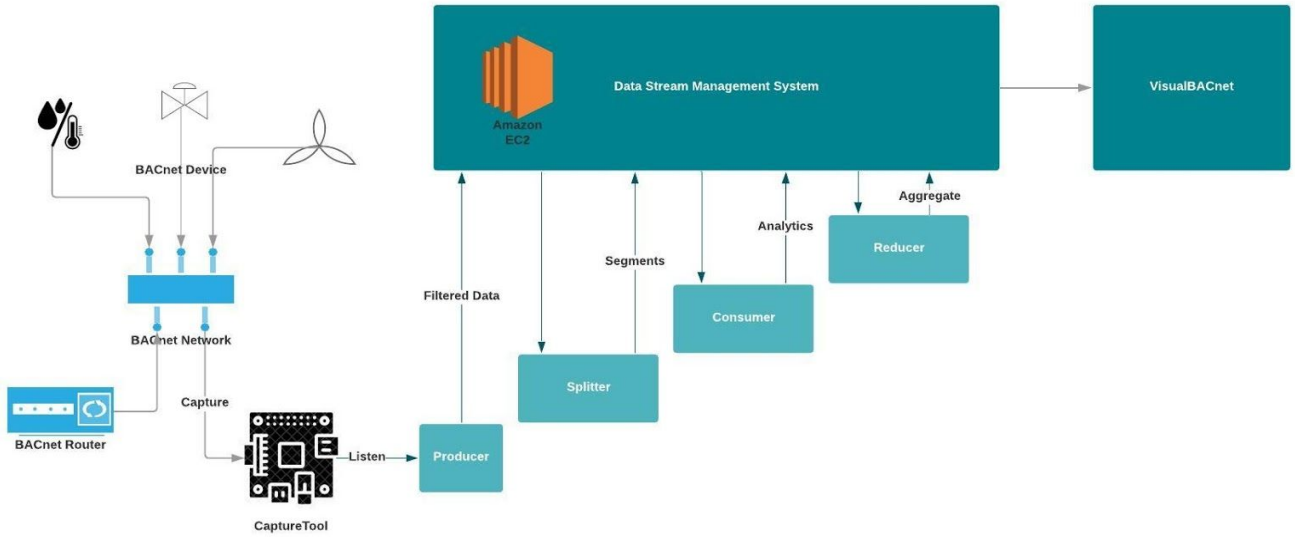

Figure 2.1: System Overview

All packets will initially go to the Producer on the capture tool from the building smart devices. This packet data will be prefiltered separately on *the Producer* and compressed to reduce the captured data's size before it's sent to the *Data Stream Management System (DSMS)*. The DSMS will be a centralized queue that will send and receive packet data from other modules. *The Splitter* module, upon receiving packet data will then systematically slice this data down to appropriate sized packet chunks (PCs) to reduce the amount of processing required for other modules in a particular instance. *The Consumer* gathers the necessary PCs where they are processed in parallel to result in more throughput with minimal synchronization. Once all the packet chunks have been analyzed, *the Reducer* module will gather processed packet chunks which are then recombined through packet reducing. The Reducer is the last job completed before being sent back to the Optigo server for analysis and displayed on the front-end.

This system described in the above paragraph and represented in figure 2.1 is known as a Microservice architecture. This system design was chosen for our solution as it represents a more scalable and robust design over the current centralized Monolithic architecture in place at Optigo Networks. Scalability and robustness of design are two core principles of Oakion Systems as we provide secure solutions that can be scaled to the requirements of any customer's network. Monolithic architectures, although simpler to develop and test, have issues with not only reliability, but scalability when different modules draw dissimilar resource requirements [5]. Oakion Systems takes pride in following the current trends of large platform industry leaders that have internally been evolving their applications toward Microservice architectures since 2015 [6]. The advantage of this solution over other architectures, like P2P solutions, is that it guarantees data integrity and redundancy on services. If one module were to go down, it can always be replaced with another, providing a robustness that cannot be matched with a P2P architecture given the problem at hand [7].

3 High-Level Requirements

3.1 General Requirements

REQ 3.1.1-FP	The system must not disrupt the workflow of original system of Optigo until it is deemed as a replacement.
REQ 3.1.2-PoC	The system shall process all incoming BACnet data.
REQ 3.1.3-PoC	The system shall not have any data contamination from original pcap.
REQ 3.1.4-EP	The system must be scalable to support future pcap checks.
REQ 3.1.5-EP	All pcaps shall be transferred using encrypted transfer protocol.
REQ 3.1.6-PoC	The system performance and optimization must be measurable.
REQ 3.1.7-EP	At least two existing analytic checks will be ported over.
REQ 3.1.8-PoC	The system must be able to display all performing checks on the frontend.
REQ 3.1.9-EP	The system must be able to handle processing a single pcap that is sized up to 1GB.
REQ 3.1.10-EP	The system must be able to be integrated into the current Optigo system.
REQ 3.1.13-EP	The system must improve the current performance in at least one aspect

	of computational or bandwidth, at the same cost of hardware, measured through comparison studies.
REQ 3.1.14-EP	The system shall not disclose any customer or commercial information beyond the scope of Optigo.
REQ 3.1.15-PoC	The deployment of the system must be done with a deployment application management tool.
REQ 3.1.16-EP	Connector/API for performance statistics
REQ 3.1.17-EP	Overall system health check and status report
REQ-3.1.18-EP	Logging and auditing of system modules

Table 3.1: High-Level General Requirements

4 Software Requirements

4.1 General Requirements

This section describes the various intricates of the system's internal workings.

REQ 4.1.2-EP	The system components communication must be secured with TLS/SSL.
REQ 4.1.3-PoC	The system components shall be written in Javascript, Go, Python or C++.
REQ 4.1.4-EP	The system must contain an abstract data structure for communication between modules.
REQ 4.1.5-PoC	The communication data structure shall either be pcap, binary or json.
REQ 4.1.6-EP	The system shall have REST endpoints/calls to Visual BACnet system for data exchange.
REQ 4.1.7-PoC	The software must be able to run both on x86 and ARM.
REQ 4.1.8-EP	The system components shall be able to join/leave the network on the fly.
REQ 4.1.9-EP	The system must have redundancy, if one worker component fails, another should take over.

REQ 4.1.10-FP	The system must not have data loss when a worker module fails.
REQ 4.1.11-EP	The system software shall be updated with Optigo's update cycle
REQ 4.1.12-EP	The system must have an automatic timeout for processing on the server side for each module
REQ 4.1.13-EP	Health Check must be integrated on the client side for each module.
REQ 4.1.14-EP	Performance shall be monitored through APM per module.
REQ 4.1.15-EP	The system must have compression on the Queue message data to reduce transfer overhead and storage cost.
REQ 4.1.16-PoC	All modules in the system must consume minimal CPU/memory when modules are not processing.
REQ 4.1.18-EP	All modules in the system must also work in Linux containers.
REQ 4.1.19-EP	All modules shall not require root permission to operate.

Table 4.1: Software General Requirements

4.2 Producer Module Requirements

The Producer module is the first to receive the raw incoming packet data stream from the smart devices as it resides on the capture tool. It will systematically filter this data to help ease the amount of bandwidth necessary to then securely communicate to the DSMS and send the filtered incoming packet stream data to this queuing mechanism.

REQ 4.4.1-EP	The Producer shall be able to receive packets sent from the Capture Tool.
REQ 4.4.2-EP	The Producer shall upload the pcap to cloud storage.
REQ 4.4.3-FP	The Producer shall delete the pcap from cloud storage.
REQ 4.4.4-EP	The Producer shall place in a reference ID of the original raw pcap into DSMS.
REQ 4.4.5-EP	The Producer shall requeue the pcap in the event the DSMS is unavailable due to connection issues.
REQ 4.4.6-EP	The Producer shall requeue the pcap in the event the DSMS is unavailable due to the DSMS failures.

REQ 4.4.7-EP	The Producer must stash unsent pcaps locally until the connection is re-established between the Producer and the DSMS.
REQ 4.4.8-EP	The Producer must have a throttling mechanism if the queue is too busy/full
REQ 4.4.9-EP	The Producer can be operated under 2GB of RAM

Table 4.2: Producer Module Requirements

4.3 Pre-filtering Requirements

Pre-filtering is the first step that the system performs on raw, incoming pcap data stream. An intelligent, elaborate pre-filtering process computationally, may significantly reduce the workload of the subsequent processes and is thus crucial to system performance. In terms of bandwidth, it is likely that this process may help ease congestion in this aspect as well.

REQ 4.2.1-EP	The pre-filtering shall remove all network data except BACnet data.
REQ 4.2.2-EP	The pre-filtering must support MS/TP, BACnet IP.
REQ 4.2.3-EP	The pre-filtering must not affect diagnosed results against data that has not been filtered.
REQ 4.2.4-EP	The pre-filtering as part of producer module to minimize network load

Table 4.3: Pre-Filtering Requirements

4.4 Data Stream Management System (DSMS) Requirements

DSMS is the dispatcher of the distributed server that orchestrates the microservices. By choosing cloud service providers, while this server is not located on site of the client, the performance is nevertheless favourable due to the access, location, replication, and failure transparencies that it provides.

REQ 4.3.1-EP	The DSMS shall be able to receive and handle requests from all other modules defined.
REQ 4.3.2-EP	All modules shall only communicate through the queue on the DSMS.
REQ 4.3.3-PoC	The module workers shall connect to the DSMS with efficient client

	library.
REQ 4.3.4-PoC	The DSMS must run on a cloud server.
REQ 4.3.5-EP	The DSMS must store data for at least 14 days.
REQ 4.3.6-EP	The DSMS must garbage collect obsolete data.
REQ 4.3.7-EP	The DSMS must be entirely transparent to the client.
REQ 4.3.8-EP	A firewall must be set up on the DSMS.
REQ 4.3.9-EP	Under heavy traffic cloud response shall be delayed/throttled but should never fail.
REQ 4.3.10-EP	The DSMS must be highly available with redundancy.
REQ 4.3.11-EP	The DSMS must be able to process data concurrently to achieve higher performance.
REQ 4.3.12-EP	The DSMS shall have periodic backups on cloud level.
REQ 4.3.13-EP	The DSMS shall have periodic backups on application level.
REQ 4.3.14-PoC	There shall be methods to manage the DSMS
REQ 4.3.15-EP	There shall be authentication on the DSMS

Table 4.4: Data Stream Management System Requirements

4.5 Splitter Module Requirements

The Splitter module receives filtered packets stream from the DSMS before systematically splitting the packet stream into packet chunks (PCs) through packet mapping and sending these chunks back to the DSMS for delegation.

REQ 4.5.1-EP	The Splitter shall be able to only receive raw pcap.
REQ 4.5.2-EP	The Splitter shall be able to strategically split the pcap file into PCs
REQ 4.5.3-EP	The Splitter shall be able to successfully send all PCs back into the queue without data contamination.
REQ 4.5.4-EP	The Splitter shall pull raw pcap from the cloud storage
REQ 4.5.5-PoC	The Splitter shall perform data type conversions on PCs to fit the DSMS

	message format if needed
REQ 4.5.6-EP	The Splitter can be operated under 2GB of RAM.

Table 4.5: Splitter Module Requirements

4.6 Consumer Module Requirements

The Consumer module processes the now packet chunks through computational analysis. The module is a key part of the system as it analyzes the data stream for the critical information useful for analytics and visualization. The consumer module will need to perform certain data checks and is scalable for increased analysis in the final product.

REQ 4.6.1-PoC	The Consumer shall be able to receive only required PCs from the DSMS.
REQ 4.6.2-PoC	The Consumer shall be able to process all PCs gathered from the DSMS.
REQ 4.6.3-PoC	The consumer shall decode PCs from the raw pcaps the DSMS.
REQ 4.6.4-EP	The Consumer can be operated under 1GB of RAM.
REQ 4.6.5-PoC	The Consumer shall contain a unified data structure on output data for the Reducer.
REQ 4.6.6-EP	The Consumer must be modular enough to add in more analytics/checks in the future
REQ 4.6.7-EP	Each job must finish under 10 minutes

Table 4.6: Consumer Module Requirements

4.7 Reducer Module Requirements

The Reducer module's job is to aggregate the analytical data that has been computed by the Consumer module back into packets to be sent to Optigo's server for visualization on the frontend.

REQ 4.7.1-PoC	The Reducer shall be able to receive from the DSMS.
REQ 4.7.2-EP	The Reducer shall be able to aggregate analytical data of every consumer without missing data.
REQ 4.7.3-EP	The Reducer shall be able to distinguish PCs that depend on each other.

REQ 4.7.4-EP	The Reducer must be modular enough to add in more analytics/checks in the future
REQ 4.7.5-PoC	The Reducer shall contain a unified data structure on output data for the Consumer.
REQ 4.7.6-EP	The Reducer can be operated under 4GB of RAM.

Table 4.7: Reducer Module Requirements

5 Hardware Requirements

The hardware requirements for this project pertains mostly toward the existing infrastructure currently deployed by Optigo Networks in the Capture tools used to send and receive packet data from smart devices to the Distributed Computing Network. These Capture Tools are embedded devices enclosed in cases and deployed to customer sites to act as intermediaries between the smart devices and Optigo Networks. For our prototype, we will be using a Capture Tool to provide the same functionality that it does in the field toward our network. The capture tool will match the same specifications as the current ones deployed by Optigo, for ease of integration as well as accuracy in the measurement of optimization our network provides in comparison to the current architecture at Optigo Networks.

5.1 General Requirements

REQ 5.1.1-PoC	The Capture Tool shall run on Linux OS.
REQ 5.1.2-PoC	The Capture Tool shall be on a 32 or 64-bit ARM platform.
REQ 5.1.3-PoC	The Capture Tool must have a minimum of 1GB memory
REQ 5.1.4-EP	Oakion Systems softwares does not affect stability of Capture Tool
REQ 5.1.5-EP	Current Optigo functionalities on the Capture Tool remains untouched after Oakion Systems softwares is deployed
REQ 5.1.6-PoC	The Capture Tool must have internet connection via wired ethernet.
REQ 5.1.7-PoC	The Capture Tool internet connection must have a minimum of 5 mbit upload/download to maintain Oakion Systems softwares' usability

REQ 5.1.8-EP	The Capture Tool OS maintains a boot time within 5 minutes.
REQ 5.1.9-EP	The Capture Tool that runs Oakion Systems softwares must be managed in a resource pool.
REQ 5.1.10-EP	The Capture Tool must support current Optigo software update cycles.
REQ 5.1.11-EP	The Capture Tool communication to the DSMS must be encrypted.

Table 5.1: Hardware General Requirements

6 Engineering Standards

Complying to and seeking out official standards and policies is integral toward the legitimacy of a marketable product. Agencies like ISO and IEEE provide regulatory guidelines adopted by many companies for quality management [3]. Considering the majority of software aspects regarding the architecture of the solution proposed by Oakion Systems, the following section will cover a majority of software requirements that the system architecture will meet and comply to. The section will also touch on the communications aspect of the project that is core to the current system in place as well as the solution proposed.

6.1 ISO/IEEE Requirements

6.1.1 ISO 12207 - Software Life Cycle Processes

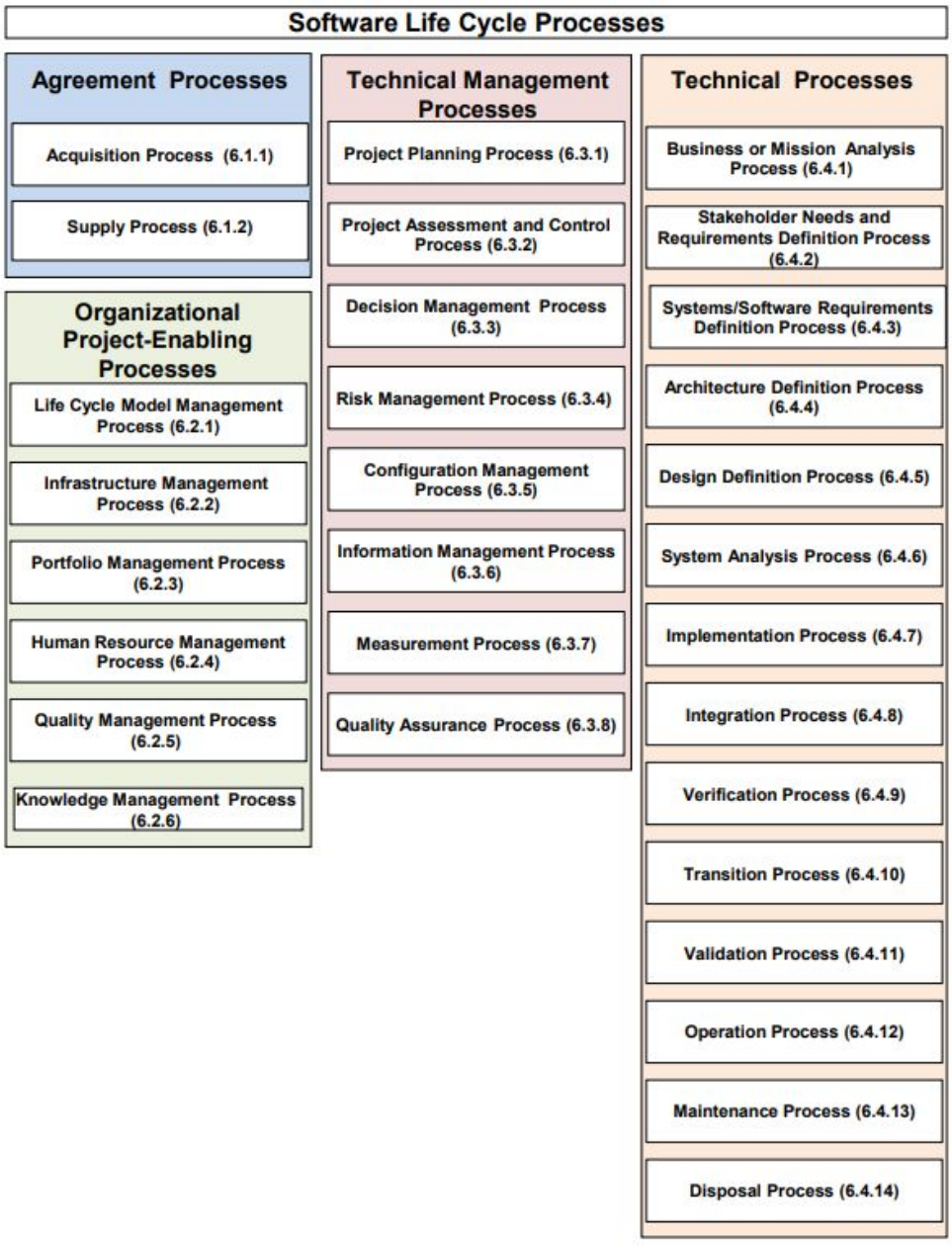

Figure 6.1 - Software life cycle processes [reference]

ISO 12207 is an international standard for software lifecycle processes that highlights required tasks for software development and maintenance [ref]. Oakion Systems will comply the following processes:

REQ 6.1.1-PoC	The system will comply with all of ISO 12207 Agreement Processes under section 6.1.
REQ 6.1.2-PoC	The system will comply with ISO 12207 Organizational Project-Enabling Processes under section 6.2.
REQ 6.1.3-PoC	The system will comply with ISO 12207 Technical Management Processes under section 6.3.
REQ 6.1.4-PoC	The system will comply with ISO 12207 Technical Processes under section 6.4.

Table 6.1: ISO 12207 Requirements

6.1.2 ISO 15026 - Systems and Software Engineering -Part 1: Concepts and Vocabulary

ISO 15026 specifies minimum requirements for the structure and contents of a assurance case to improve the consistency and comparability of assurance cases and to facilitate stakeholders communications and engineerings decisions [10]. Oakion Systems will comply the following sections:

REQ 6.1.5-PoC	The Oakion Systems will comply with all of ISO 15026 Terms and Definitions under section 3.
REQ 6.1.6-PoC	The Oakion Systems will comply with all of ISO 15026 Basic Concepts under section 5.
REQ 6.1.7-PoC	Oakion Systems will comply with all of ISO 15026 Integrity levels under section 8.

Table 6.2: ISO 15026 Requirements

6.1.3 ISO 15289 - Content of life-cycle information items

ISO 15289 specifies the content of all identified systems and software life-cycle and service management information items [11]. Oakion Systems will comply the following sections:

REQ 6.1.8-PoC	The Oakion Systems will comply with all of ISO 15289 Life-cycle data and information items under section 6.
REQ 6.1.9-PoC	The Oakion Systems will comply with all of ISO 15289 Mapping of information items to the life cycle and service management processes under section 8.
REQ 6.1.10-PoC	The Oakion Systems will comply with all of ISO 15289 Specific information item under section 10.

Table 6.3: ISO 15289 Requirements

6.1.4 ISO 16484-5 - Building automation and control systems (BACS) -Part 5: Data communication protocol

ISO 16484 provides a standard for BACnet where this protocol provides a comprehensive set of messages for conveying encoded binary, analog, and alphanumeric data between devices including, but not limited to [12]:

a. hardware binary input and output values
b. hardware analog input and output values
c. software binary and analog values
d. text string values
e. schedule information
f. alarm and event information
g. files
h. control logic.

Oakion Systems will comply to the following sections during processing:

REQ 6.1.11-PoC	The Oakion Systems will comply with all of ISO 16484-4 Networking under section 6.
REQ 6.1.12-PoC	The Oakion Systems will comply with all of ISO 16484-4 Data Link/Physical Layers: Master-Slave/Token Passing under section 9.
REQ 6.1.13-PoC	The Oakion Systems will comply with all of ISO 16484-4 File Access

	Services under section 14.
REQ 6.1.14-PoC	The Oakion Systems will comply with all of ISO 16484-4 Object Access Services under section 15.
REQ 6.1.15-PoC	The Oakion Systems will comply with all of ISO 16484-4 BACnet Procedures under section 19.

Table 6.4: ISO 16484 Requirements

6.2 Coding Standards

REQ 6.2.1-PoC	Javascript shall follow Airbnb standard.
REQ 6.2.2-PoC	GoLang shall follow Effective Go coding standard.
REQ 6.2.3-PoC	Python shall follow PEP8 coding standard.
REQ 6.2.4-PoC	C/C++ shall follow Google coding standard.
REQ 6.2.5-PoC	The code must reside in Optigo version control environment.
REQ 6.2.6-EP	Documentation must be provided for any external API.

Table 6.5: Coding Standards Requirements

7 Sustainability/Safety

Given the volume of software aspects in the proposed solution, we will only comment on the hardware that we make use of and their compliance with the Cradle-to-Cradle (C2C) Model. As the C2C Certification Criteria defines, "Electronic and industrial waste is acceptable as long as it does not leave the industrial cycle." [9]

The prototype and final product produced by Oakion aims to eliminate waste, and instead provide Technical Nutrients[18]. As the only electronics involved in our project are Capture Tools (custom embedded Linux board), network switches, and various networking cables, we will focus on reusing all of these. The Capture Tools are kindly provided by Optigo, and will be used throughout our prototyping to product release lifecycle. At the end of the cycle it will be returned to Optigo for testing/deployment purposes. Similarly, the network switches and

cables are used throughout the development cycle. At the end of the cycle they will either be given to Optigo, or owned by group members for future use, given their high versatility.

Thus we will be able to qualify these electronic hardware as Technical Nutrients, minimizing the impact of the material on the environment.

The figure below demonstrates Oakion's presence in the C2C technical cycle:

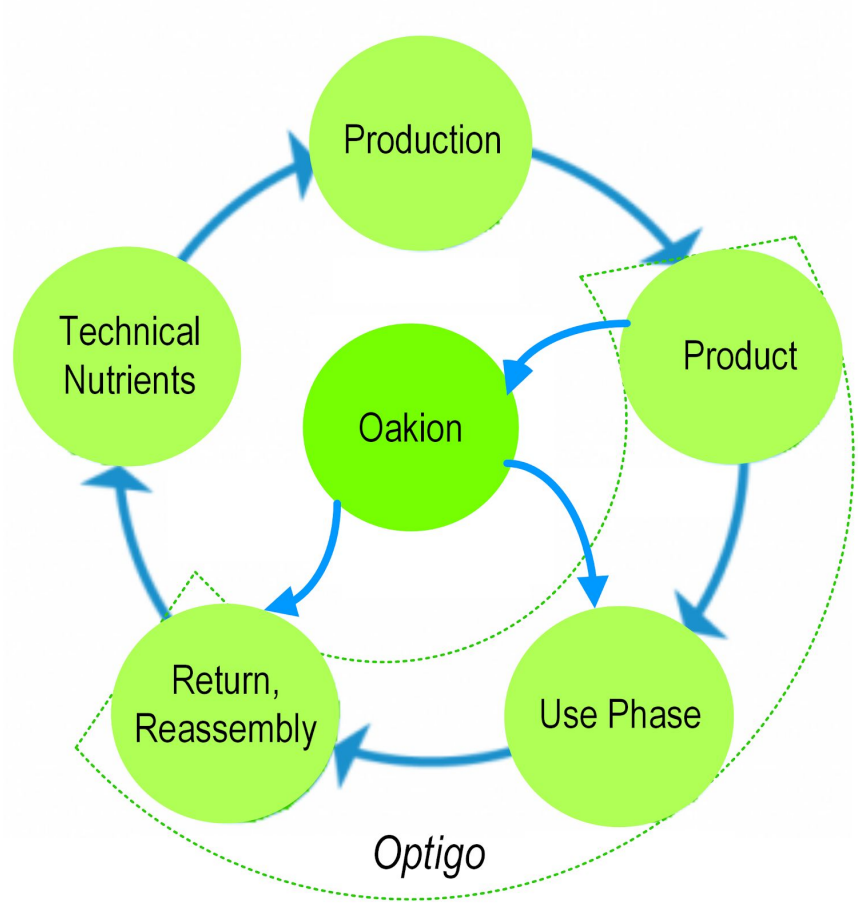

Figure 7.1 : Cradle-to-Cradle Model

As for public safety, while we are deeply concerned, the operation condition of the Capture Tool is beyond the scope and power of our project. However given the robustness in embedded devices' design, we may speculate that a fire hazard is unlikely. Nevertheless, measures will be set up to monitor node failures.

Safety in terms of sensitive data are also outlined below.

REQ 7.1.1-PoC	Corporate ethics and fair labour statements are publicly available and adopted across Optigo.
REQ 7.1.2-PoC	The system shall monitor nodes in order to promptly respond to failures
REQ 7.1.3-PoC	The system shall keep all pcap files confidential to itself.
REQ 7.1.4-PoC	The system must be encrypted to prevent vulnerabilities.
REQ 7.1.5-PoC	The pcap files shall be kept permanently on cloud storage until there is a request to delete it.
REQ 7.1.6-PoC	The pcap file must be deleted permanently from the system if requested.
REQ 7.1.7-PoC	All modules in the system shall continuously pass all defined penetration tests.
REQ 7.1.8-PoC	Any SSH access must only accept RSA/DSA private keys and not password.
REQ 7.1.9-FP	Pcap data must be anonymous to clients' infrastructure maintainers.

Table 7.1: Safety Requirements

8 Conclusion

Oakion Systems' Distributed Computing Network provides a solution to alleviate high traffic bottlenecks situations in systems built for network data. It stands apart from other optimization competitors in the field for its robust security measures and cross platform abilities while still being lightweight. The solution demonstrates its cost efficiency through its adherence to maximization of software for prime performance without the need of upgrading existing hardware. When compared to other architectures systems in place, Oakion Systems' Microservice Architecture not only complies with the current trends of other large market platforms, but is also provides scalability and security like no other. The solution also allows for better flexibility for future development.

This document outlines the requirement specifications of Oakion Systems' proposed middleware solution of a Distributed Computing Network to solve the bottleneck issues facing Optigo Networks in the future. Covered here are all three phases of development:

Requirements have been tactically organized by system type and module for the ease of the reader and are summarized in the following:

High Level Requirements	• The system will be scalable and able to process all incoming BACnet data then integrate with the current Optigo system. • The system will not disclose any customer or commercial information beyond the scope of Optigo.
Software Requirements	• The system contains the producer, splitter, consumer, and reducer module, which communicates with the DSMS.
Hardware Requirements	• Capture tool will match the same specifications as the current model deployed by Optigo and and communication to the DSMS will be encrypted.
Engineering Standards Requirements	• The software life cycle processes, systems, software, automation engineering, and coding standards published by acclaimed organizations that the team will follow as a guideline throughout the project.
Safety and Sustainability	• The Cradle-to-Cradle Model will be followed, aiming towards eliminating waste and provide Technical Nutrients.

Appendix

Acceptance Test Plan

The following is a test plan for Oakion Systems Distributed Computing Network PoC design. As this test plan will be created before any major implementation of the project has commenced, the plan will act as a primitive guideline focusing on principle points of testing for the PoC product. The Performance Checks are mostly to determine whether the microservice solution that is implemented has signs of optimization (timing, scaling, or computational). Further details on verification will be provided in future documentation in the design specification.

Hardware Checks

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Capture Tool Switch	The Capture tool should correspond to the switch	Capture tool powers on when the switch is on and vice versa.				
Capture Tool Temperature	The Capture Tool should not overheat	The Capture Tool should operate below 82 ℃.				
Marking Instructor Initials:				**Date:**		

Security Checks (Pre-scan Reports)

Will not be a real time scan, will be a pre-scan report:

Checks	Description	Expected Outcome	Actual Outcome	Comments		

Vulnerability Scan	The system should not be open to software vulnerabilities. Rapid7 Nexpose shall be used for scanning.	No Vulnerabilities should be detected.				
Penetration Test	All ports that are not used by our system or not authenticated should not pass.	The ports used by the system are the only ports open.				
Marking Instructor Initials:				**Date:**		

Module Checks

DSMS Module

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Alert when Service Down	The modules cannot connect to DSMS. This is reported through an APM	The APM should clearly indicate service failure.				
Monitor Transaction Rate	The data that is going through the DSMS can be monitored as a rate. This is reported through an APM.	The transaction rate of the DSMS should clearly be shown from the APM.				
Monitor System Load	The DSMS should be constantly measured for data loadage. This is reported through an APM.	The system load of the DSMS should clearly be shown from the APM.				

Monitor the Queue size of the DSMS	The amount of data that is in queue on the DSMS should be monitored. This is reported through an APM.	The queue length and size of the DSMS should clearly be shown on the APM.				
Unplanned shutdown	In the event of an unplanned shutdown, the system will not lose any data.	The existence of all data before the shutdown should still all exist after the recovery of the system.				
Marking Instructor Initials:				**Date:**		

Splitter Module

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Monitor memory usage	The Splitter should have its memory memory usage monitored through Linux terminal commands (top, htop.. etc).	The memory usage is displayed clearly from the terminal.				
Send pcap data	The Splitter should be able to send data to the DSMS	The DSMS queue must increase in size with respect to the Splitter.				
Receive raw pcap data	The Splitter should be able to receive raw pcap data from the DSMS	The DSMS queue must decrease in size with respect to the Splitter.				
Marking Instructor Initials:			**Date:**			

Consumer Module

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Monitor memory usage	The Reducer should have its memory memory usage monitored through Linux terminal commands (top, htop.. etc).	The memory usage is displayed clearly from the terminal.				
Process PCs	The Consumer should be able to process the data for analysis	The PC is structured with metadata.				
Send JSON data	The Consumer should be able to send processed JSON data to the DSMS	The DSMS queue must increase in size with respect to the Consumer.				
Receive PC data	The Consumer should be able to receive raw pcap data from the DSMS	The DSMS queue must decrease in size with the respect to the Consumer.				
Marking Instructor Initials:				**Date:**		

Reducer Module

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Monitor memory usage	The Reducer should have its memory memory usage monitored through Linux terminal commands (top, htop.. etc).	The memory usage is displayed clearly from the terminal.				

Receive PC data	The Reducer should be able to receive pcap data from the DSMS	The DSMS queue must decrease in size with the respect to the Reducer.				
Analyze PCs	The Reducer should be able to a do a full analysis on a pcap check.	The output of the Reducer should be meaningful JSON on command terminal.				
Marking Instructor Initials:				**Date:**		

Performance Checks

Checks	Description	Expected Outcome	Actual Outcome	Comments	PASS	FAIL
Time spent by splitter on a single Pcap	Determine the time spent on the Splitter from a single pcap data file.	The job will finish in a relative timeframe. It will act as an initial performance benchmark				
Time spent by consumer on a single Pcap	Determine the time spent on the Consumer from a single pcap data file.	The job will finish in a relative timeframe. It will act as an initial performance benchmark				
Time spent by reducer on a single Pcap	Determine the time spent on the Reducer from a single pcap data file.	The job will finish in a relative timeframe. It will act as an initial performance benchmark				
Memory Usage on Splitter	Determine the real-time memory	The Splitter should not bottleneck the				

	usage of the Splitter on a single pcap data file.	memory.				
Memory Usage on Consumer	Determine the real-time memory usage of the Consumer on a single pcap data file.	The Consumer should not bottleneck the memory.				
Memory Usage on Reducer	Determine the real-time memory usage of the Reducer on a single pcap data file.	The Reducer should not bottleneck the memory.				
Overall System Ingestion	Compare the ingestion timing of Optigo's current architecture and the microservice architecture using a single pcap data file.	The microservice solution should provide better results if scaled with more machines.				
System Storage	Compare the system storage between Optigo's current architecture and the microservice architecture using a single pcap data file.	The microservice solution should consume less overall non-volatile storage.				
Marking Instructor Initials:				**Date:**		

References

[1] "Accelerated PCAP - an architecture for precision packet capture and analysis," *Napatech*. [Online]. Available: https://www.napatech.com/support/resources/white-papers/accelerated-pcap-an-architecture-for-precision-packet-capture-and-analysis-on-high-speed-networks/. [Accessed: 01-Feb-2019].

[2] N. Peck, "Microservice Principles: Smart Endpoints and Dumb Pipes," *medium.com*, 01-Sep-2017. [Online]. Available: https://medium.com/@nathankpeck/microservice-principles-smart-endpoints-and-dumb-pipes-5691d410700f. [Accessed: 02-Feb-2019].

[3] M. Bangert, "Quality management: The Importance of ISO," *Quality Magazine RSS*, 14-Sep-2012. [Online]. Available: https://www.qualitymag.com/articles/84855-quality-management-the-importance-of-iso. [Accessed: 04-Feb-2019].

[4] "Solutions for Compute-Intensive Environments," *Distributed Computing and Grid Computing Solutions from Digipede*. [Online]. Available: http://www.digipede.net/solutions/distributed-computing.html. [Accessed: 04-Feb-2019].

[5] P. D. Francesco, P. Lago, and I. Malavolta, "Architecting with microservices: A systematic mapping study," *Journal of Systems and Software*, vol. 150, pp. 77–97, 2019.

[6] A. Kharenko, "Monolithic vs. Microservices Architecture – Microservices Practitioner Articles," *Microservices Practitioner Articles*, 09-Oct-2015. [Online]. Available: https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59. [Accessed: 06-Feb-2019].

[7] Bestofmedia Team, "Client/Server Versus Peer Networks - LAN 101: Networking Basics," *Tom's Hardware*, 15-Sep-2011. [Online]. Available: https://www.tomshardware.co.uk/local-area-network-wi-fi-wireless,review-32275-2.html. [Accessed: 06-Feb-2019].

[8] "Mobile endpoint security: What enterprise infosec pros must know now", IoT Agenda, n.d. [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT. [Accessed: February 6, 2019]

[9] "OI Engine, an innovation management software built on design thinking," *OpenIDEO* -

How might we get products to people without generating plastic waste? - Replace plastic straws with paper straws, 12-Jun-2012. [Online]. Available: https://challenges.openideo.com/challenge/e-waste/inspiration/cradle-to-cradle. [Accessed: 06-Feb-2019].

[10] "IEEE Trial-Use Standard--Adoption of ISO/IEC TR 15026-1:2014 Systems and Software Engineering--Systems and Software Assurance--Part 1: Concepts and Vocabulary."

[11] "ISO/IEC/IEEE International Standard Systems and software engineering -- Content of life-cycle information items (documentation)."

[12] "Building automation and control systems (BACS) -- Part 5: Data communication protocol' ."

[13] "Technology Dictionary," *Techopedia*. [Online]. Available: https://www.techopedia.com/dictionary. [Accessed: 04-Feb-2019].

[14] "Kafka, A distributed streaming platform," *Apache*. [Online]. Available: https://kafka.apache.org/intro. [Accessed: 04-Feb-2019]

[15] "IEEE Advancing Technology for Humanity," *IEEE*. [Online]. Available: https://www.ieee.org/. [Accessed: 02-Feb-2019]

[16] "International Organization for Standardization," *ISO*. [Online]. Available: https://www.iso.org/home.html. [Accessed: 02-Feb-2019]

[17] IEEE Standard Adoption of ISO/IEC 15026-1--Systems and Software Engineering--Systems and Software Assurance--Part 1: Concepts and Vocabulary," in IEEE Std 15026-1-2014, vol., no., pp.1-45, 4 Nov. 2014 doi: 10.1109/IEEESTD.2014.6948215

[18] *Home - Cradle to Cradle Products Innovation Institute*. [Online]. Available: https://www.c2ccertified.org/resources/collection-page/cradle-to-cradle-certified-resources-public. [Accessed: 08-Feb-2019].