대한기계학회 주최

제9회 전국학생설계경진대회(2019년)

설계 최종 보고서

참가부	고등부 (O)				
참가분야	공모주제				
참가팀명	북한산의 인연				
설계제목	IoT를 통한 지역의 기상상황 자동 수집, 물기를 제거하는 의자				
지도교수/교사	(소속) 하나고등학교 (성명) 김명진 (연락처) (이메일) wahrheit93@hana.hs.kr				
대표자 (신청인)	성명	소속	연락처 (휴대폰)	E-mail	주소
	고도형	하나고등학교		kodh49@naver.com	

참가팀원 인적사항

NO	성명	소속 / 학년	E-MAIL
1	김마린	하나고등학교 1학년	kimmarin0420@naver.com
2	유호준	하나고등학교 2학년	jayyoo2002@gmail.com
3	이원재	하나고등학교 2학년	wjlee0421@gmail.com
4	이정원	하나고등학교 1학년	jungwonhill@naver.com
5	이준서	하나고등학교 1학년	junseo0344@naver.com
6			

설계 요약문

참가분야	**공모주제**
참가팀명	북한산의 인연
설계제목	IoT를 통한 지역의 기상 상황 자동 수집 및 물기를 제거하는 의자
대표자명	고도형
요약문	4차 산업 혁명을 겪으며 각종 IT 기술이 생활 속 불편함을 해소하는 데 활발히 이용되고 있다. 그런데 이런 기술은 R&D 현장에서뿐만 아니라 공공 시설물의 상태와 관리를 효율적으로 구축하는 서비스 측면에서도 필요하다. 이에 본 연구진은 공공 시설물 중 우리가 가장 보편적으로 이용하는 시설물로 공원이나 아파트 단지 등의 의자를 선정하였고, 이에 이 시설물의 상태를 실시간으로 파악하고 즉각적인 관리까지 할 수 있는 시스템을 구축하였다. 본 제품은 하나의 시스템으로 이루어져 있으며 세부적으로 3개의 파트로 구성되어 있다; 습기가 표면에 흡수되지 않는 의자, 의자에 설치될 물기 제거 장치, 그리고 온습도 측정 센서다. 우선 의자에 설치된 물기 제거 장치에는 Arduino를 이용하여 중앙 네트워크와 상호 통신할 수 있도록 LoRa 수신기를 설치한다. 여기에는 와이퍼 블레이드를 부착하여 장치가 의자의 좌우로 움직이면서 의자 표면에 묻은 물기를 제거한다. 중앙 네트워크에서는 기상청 API를 파싱하여 특정 알고리즘을 수행하여 LoRa 네트워크 게이트웨이를 사용하여 클라이언트에 Arduino 작동 명령을 내린다. 이러한 상호작용을 통하여 모든 클라이언트(의자)에서 물기를 제거하는 동작을 수행할 수 있도록 하였다. 본 제품은 네트워크 스스로 기상 상황을 특정 시간 간격에 맞추어 파악함으로서 굉장히 많은 수의 기기들을 한 번에 편리하게 사용할 수 있도록 한다는 점에서 사물인터넷의 강력함을 잘 보여주며 기존 기기들보다 시간적, 에너지 효율 면에서 훨씬 뛰어나다. 또한 저 전력, 장거리 무선 통신이 가능한 LoRa 기술을 활용하기 때문에 다른 기술에 비하여 에너지 소모가 적고 더욱 오랜 시간 동안 클라이언트와의 연결을 유지할 수 있기 때문에 유지 보수 측면에서도 뛰어나다. 넓은 야외 공간에 여러 대의 의자를 배치해야 하는 상황을 고려한다면 이 제품은 상용화될 수 있을 것이다. 특히, 기존에는 의자가 젖어 있을 때 직접 물기를 닦는 불편함을 감수하지 않는 이상 앉을 수 없었으나 이 제품을 도입함으로서 그러한 불편함을 해소할 수 있을 것이다.

1. 설계의 필요성 및 목적

이른 새벽이나 비가 갠 후에 맑은 공기를 마시며 운동이나 산책을 하면, 가끔씩 피로를 느낄 때가 있다. 그러나 앉아서 쉬기에는 공원의 의자가 젖어 있는 경우가 많아 앉기 힘들다는 불편함이 존재한다. 그리하여 이러한 불편함을 일괄적으로 사람 대신 해결해주고자 물기를 닦아주는 기계장치가 결합된 의자를 개발하고 이를 통합적으로 컨트롤할 수 있는 네트워크 시스템을 설계하고자 하였다. 이를 위하여 사람의 인위적인 개입 없이 네트워크상으로 의자의 상태를 파악하고 물기를 제거할 수 있도록 기상 상황과 온습도 데이터를 인자로 받아 작동하는 알고리즘을 구성하고 이러한 알고리즘을 각각의 의자가 연결된 네트워크에 적용함으로서 특정한 공간에 존재하는 여러 개의 의자들을 우천 시에 통합적으로 관리할 수 있는 모델을 제시하고자 한다.

2. 설계 핵심 내용

(1) 설계 문제의 정의

길을 지나다 보면, 시민의 편의와 수요를 충족하기 위해 설치되어 있는 공공 시설물이나 편의 시설을 쉽게 발견할 수 있고, 이는 대부분 누구나 이용할 수 있는 시설이다. 그런데 이런 시설물은 명확하게 사유 재산으로 규정되어 있지 않은 경우가 많기에 상태 관리와 유지/보수의 기준 역시 불명확하다. 원칙적으로는 환경 미화 공무원이나 시설물을 설치한 단체에 관리의 책임이 부여되지만, 사실상 실시간으로 시설물들의 오염 상태와 사용 가능 여부 등을 확인하는 등 즉각적인 관리 체계는 잡혀있지 않아 어려움을 겪는 경우도 많다. 본 설계자들도 비가 오는 등 기상 상태에 변화가 생겨 공원의 의자가 모두 젖어버리거나, 공공 화장실의 변기가 막히거나 심하게 오염되어 사용하지 못하는 상황 등 문제 상황에 봉착하며 이것을 즉각적으로 대응할 수 있는 시스템이나 최소한 사용 가능 여부에 대한 정확한 정보를 실시간으로 전달해 주는 시스템의 필요성에 대해 생각해 보게 되었다. 실제로 경기도 하남시의 경우 '스마트시티 서비스'를 도입하여 교통제어정보, 실시간 신호 제어, 돌발 상황 관리, 대중교통정보 제공, 주/정차 위반 단속, 공공지역 안전감시 서비스를 제공하고자 하는 계획을 실천 중에 있다. 또, 은평구 진관동의 쓰레기 처리 시스템의 경우 종량제 봉투에 새겨진 디지털 홀로그램 무늬를 인식시켜 분리수거를 관리하는 등 공공 시설물에도 디지털 및 기술이 첨가되어 관리가 이루어지고 있다. 이러한 정보 제공 및 편의 개선에 더불어, 우리가 사용하는 공공 시설물의 상태와 관리를 효율적으로 구축하는 서비스 역시 필요하다고 생각한다. 공공 시설물 중 우리가 가장 보편적으로 이용하는 시설물로 공원이나 아파트 단지 등의 의자를 선정하였고, 이에 이 시설물의 상태를 실시간으로 파악하고 즉각적인 관리까지 할 수 있는 시스템을 구축하고자 한다. 의자의 젖은 정도나 오염 정도, 청결 상태 및 고장 여부 등을 파악한 후, 이것의 상태를 정보로 저장하여 표시하고 물기 제거 및 오염물 청소 등의 피드백을 제공하는 장치를 설계하여 불명확하게 규정된 공공 시설물 관리에 대한 문제를 해결하고자 본 설계를 시작하게 되었다.

(2) 설계의 독창성 및 접근 방법

1) 설계 방법 및 배경

본 연구에서는 (1)에서 정의한 바와 같은 설계문제를 해결하고자 다음과 같은 시스템을 설계하였다. 시스템은 크게 3가지 세부 시스템인 (a), (b), 그리고 (c)로 구성된다. 다음 그림 II-1은 앞서 서술한 3가지 시스템의 연결구조를 나타낸 다이어그램이다. 1개의 시스템, 즉 공원에 총 12개의 의자가 존재할 때의 네트워크의 작동 원리를 단순화한 것으로 각 시스템에 1개의 온습도 측정 센서(Humidity sensor)가 존재하면 LoRa 네트워크 수신기(LoRa Receiver)와 중앙 네

(a) 의자에 설치될 물기 제거 장치
(b) 시스템 전체의 습도 측정 센서
(c) 중앙에서 (a)의 각 장치를 제어하는 네트워크

그림 II-2

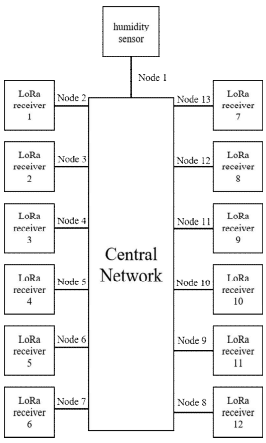

그림 II-1

트워크(Central Network) 가 연결되어 중앙 네트워크의 판단을 즉각적으로 수신하고 이를 바탕으로 각각의 의자에 설치된 물기 제거 장치가 작동하게 되는 구조이다. 따라서 본 시스템 내 총 Node의 개수는 의자의 개수를 n개라고 할 때 습도 측정 센서 1개를 추가로 고려한 $n+1$개가 된다.

(a) 의자에 설치할 물기 제거 장치

Arduino를 이용하여 중앙 네트워크와 상호 통신할 수 있도록 LoRa 수신기를 설치하여 의자에 결합한다. 공원에 한정되지 않다고 가정한다면 일반적인 야외시설의 경우 최소 한 개 이상의 의자가 존재한다. 그리고 한국과 같이 국토가 그리 넓지 않은 국가의 경우, 비가 아주 좁은 지역이 아닌 상당히 넓은 지역에 고루 내린다. 반면, 의자들이 배치되어 있는 장소의 면적은 대부분 이에 비해 괄목할 정도로 작다. 그렇기에 공원의 서로 다른 어떠한 두 지점에 대하여 해당 두 지점의 날씨가 서로 다를 확률이 무시할 정도로 낮다. 이에 클라이언트로서 작용할 각각의 의자에는 네트워크로부터 신호를 수신할 수 있는 LoRa 수신기와 Arduino 모듈을 부착한다. 다음 그림 II-3은 물기 제거 장치의 예상 설계 도안이다. 고무로 이루어진 와이퍼 블레이드와 동일한 장치를 그림 II-3의 A 부분에 부착하여 금속 의자의 표면에 묻은 물방울을 제거한다. 와이퍼 블레이드의 아랫부분에 LoRa 수신기와 모터를 추가로 부착하여 자동으로 블레이드를 움직여 작동시킬 수 있도록 유도하였다.

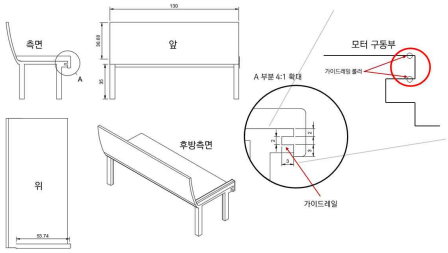

그림 II-3

(b) 시스템 전체의 습도 측정 센서

한 시스템에 한 개로 충분한 습도 측정 모듈은 Arduino의 DHT11 모듈을 사용하여 습도를 측정한다. DHT11은 정전식 습도 센서가 내장되어 있는 온습도 센서로서 습도에 따라 저항 값이 변하는 특징을 바탕으로 한 장치이다. 습도 측정 센서는 주변 인파와 독립적으로 항상 정확한 값을 측정해야 하기 때문에 지상으로부터 약 2.5m ~ 3m 높이에 떨어진 곳에 설치하여 사람들의 손이 닿을 수 없도록 설계한다. 습도 측정 센서를 통해 측정한 값은 15분을 주기로 중앙 네트워크에 송신하여 주기적인 기상 상황 변화를 잡아낼 수 있도록 한다.

(c) 중앙에서 (a)의 각 장치를 제어하는 네트워크

최근 IT관련 연구와 상품들로부터 각광을 받고 있는 IoT 기술을 완성시키기 위하여 안정적인 전송 기술이 필요하다. 본 연구에서 제안하는 설계안 또한 실시간으로 데이터를 제공받아 각 Node에 신호를 보내야 하기 때문에 이러한 요건이 필수적이다. 이때 필요한 기술이 차세대 무선 네트워크 무선 표준이라 불리는 SemTech의 LoRa(Long Range) 기술이다. LoRa는 스마트폰에서 주로 사용되는 3G나 LTE(Long Term Evolution), 5G와는 달리 최소한의 전력을 사용하며 10km ~21km에 달하는 장거리 통신이 가능하다. 또한 별도의 기지국이나 중계장비가 필요하지 않아 최초 설비 제공 비용이나 지속적 관리비용을 최소화할 수 있는 방안이다. 또한 LoRa는 단 하나의 네트워크에 여러 개의 센서를 연결할 수 있는 기술(다중 센서 통신)이 가능하고 AES128의 보안 방식(미국 표준 기술 연구소 NIST에 의하여 지정된 128bit의 Data Block를 사용하는 방식)을 채택함으로서 IoT와 함께 주목받고 있는 데이터 보안의 측면에서도 탁월하다. 이러한 특징에서 알 수 있듯, 센서와 센서 간의 지속적인 통신이 필요할 뿐만 아니라 여러 개의 Node와 동시 통신을 시도해야 하는 본 연구와 가장 적합한 통신 기술이라 판단하였다. 본 연구에서는 다음과 같은 원리로 작동하는 네트워크를 구현하고자 한다. 기상청으로부터

Conectivity	Topology	Power	Datarate(mbps)	Coverage(Km)	Application	Cost
RFID	P2P	Very Low	<0.4	<0.003	Unit Tag	Low
Bluetooth	Star	Low	<0.7	<0.03	데이터교환	Low
Zigbee	Mesh, Star	Very Low	<0.25	0.0.1~0.3	Sensing	Mid
Wi-Fi	Star	Low ~ High	11~100	0.004~0.02	Internet	Mid
WiHART	Mesh, Star	Very Low	<0.25	0.2	Sensing	Mid
6LoWPAN	Mesh, Star	Very Low	<0.25	0.8	Sensing	Mid
LoRa	Mesh, Star	Low	<0.3	<21	Sensing	Low
Wi-MAX	Mesh	High	11~100	<50	Broadband	High
3G/4G	Mesh	High	1.8~7.2	Cell Unit	Cell Phone	High

그림 II-4

현재 Node가 설치된 해당 지역의 날씨를 실시간으로 받아 오기 위하여 중앙 네트워크에는 업데이트되는 기상청 API (JSON Data)를 Auto Parsing할 수 있는 기능을 추가적으로 설치한다. 중앙 네트워크는 다음의 원리를 기반으로 작동한다.

1. 현재 시간대의 기상청 날씨정보를 API Parsing을 통하여 가져온다.
3. 시스템에 설치한 Arduino 습도 측정 모듈로부터 현재 시간대의 습도 데이터를 받아온다.
4. 만약, 습도 데이터의 상대습도 수치가 100%이고 강수형태 코드가 1, 즉 비가 올 경우에는 의자 위에 물기가 있으므로 이를 닦아 내기 위하여 곧바로 기계를 작동시킨다.
5. 100% 이하로 습도가 떨어질 때, 즉 두 시간대의 습도 데이터를 받아 와서 시간에 따른 습도의 그래프의 기울기가 음수가 될 때부터 기계를 작동시킨다. 기상청의 강수형태 예보 Code가 1이 아니게 되는 측정 시점까지 이를 재귀하다가 위 조건을 만족시킬 경우 기계의 작동을 중지한다.
6. 현재 시간이 8:00~24:00이고 만약, 습도 데이터의 상대습도 수치가 80% 이하이고 강수형태 코드가 1이 아닐 경우에는 기계를 작동시키지 않는다. 다음 측정 데이터를 기다린다.
7. 만약, 다음 측정 데이터에서 습도가 빠르게 증가하여 다른 조건을 만족할 때는 해당 코드를 따른다.
8. 현재 시간이 24:00~8:00이고 만약, 습도 데이터의 상대습도 수치가 80% 이상일 경우 30분 간격으로 기계를 작동시킨다.
9. 현재 시간이 (8) 조건을 만족하고 이전 측정 데이터와 상대 습도를 비교하였을 때 변화율이 양수일 경우 20분 간격으로 기계를 작동시키며 다음 측정 데이터를 기다린다.

2) 설계의 독창성

(1) 선행 연구 (Literature Review)

연구를 진행하기에 앞서 본 연구에서 제시하고자 하는 설계안과 이미 존재하는 특허나 연구 논문들에 대한 비교분석을 시도하여 기존 연구와의 회피전략을 연구하고자 하였다.

(a) 특허 분석

현재 특허에 방수 의자와 관련된 등록되어있는 특허의 개수는 1개이며 특허 신청이 있었던 사례는 총 2개가 있다. 우선, 순수한 방수 의자: 특허 등록이 거절되어 있다. 의자를 방수 코팅으로 덮어서 비가 오더라도 깨끗하게 한 번만 물을 쓸어도 의자의 물기가 사라진다. 다음으로 기울기가 있는 방수 의자: 의자에 다소 볼록한 경사면을 설치함으로써 외부에 의한 아무런 개입 없이도 빗물이 알아서 쓸려나간다. 본 연구에서 제시하고자 하는 의자의 형태는 현재 특허청에 등록된 사례 중에는 해당하는 것이 전혀 없으며, LoRa 네트워크를 사용할 경우, 특정 지역의 날씨 정보를 입력하여 외부의 개입 없이 Arduino를 기반으로 한 물기 제거 장치를 작동시킴으로서 의자에 묻은 물기를 본 연구에서 제시한 장치로 깔끔하게 청소할 수 있다. 앞서 언급한 특허 (2)가 현실에 반영되지 않는 이유는 의자의 목적과 의자의 형태가 실질적으로 맞지 않을 뿐 아니라 재정적인 문제까지 겹친 것이라고 볼 수 있다. 의자는 사람이 앉기 위해서 만들어진 것이다. 그러나 의자의 바닥면에 경사가 생긴다면 사람들이 그 의자에 앉을 때 불편함을 느낄 수 있어서 현실에서 실행이 되지 않는다. 또한, 재정적인 측면에서 의자에 경사면과 방수기술이 결

합 된다면 그것을 만드는 데에 드는 비용이 만만치 않아서 현실에서 실현되지 않는다. 본 연구에서 제시하는 기술은 비용적인 면에서 앞서 말한 방수 의자와 크게 차이가 나지 않는다는 점이다. 오히려 경사면을 어떤 각도로 조절할지 모르는 (2)의 의자와 달리 의자의 앉는 부분이 평평하더라도, 기계로 확실하게 청소를 하는 것이 더 효율적이다.

(b) 연구논문 분석

1. 효율적인 센싱을 위한 LoRa 기술 동향 및 비교분석, 한국통신학회 학술대회논문집 (2019), 장종욱

최근 IT관련 연구와 상품들에서 각광을 받고 있는 IOT기술을 완성시키기 위해서는 안정적인 전송기술이 필요하다. 이 때 필요한 기술이 차세대 무선 네트워크 무선 표준이라고 불리는 Semtech의 LoRa(Long Range)이다. LoRa는 스마트폰에서 주로 사용되는 3G나 LTE(Long Term Evolution), 5G와는 달리 최소한의 전력으로 10km 통신할 수 있고, 별도의 기지국이나 중계장비가 필요하지 않다. 또한 LoRa는 단 하나의 노드에 여러 개지의 샌서들을 연결할 수 있는 기술(다중 센서 통신)이 가능하고, IOT와 함께 주목받고 있는 보안또한 AES128(미국 표준 기술 연구소(NIST)에 의해서 지정된 데이터 블록의 길이가 128bit)이라는 암호화 방식이 적용된다.이러한 점을 보았을 때, 센서와 센서간의 지속적인 통신을 요구하는 프로젝트를 완성하는데 가장 적합한 통신기술은 LoRa라고 결론 내렸다.

2. 사물인터넷 기반 실내환경 측정 센서 모듈 개발, 한국생활환경학회지, 26(1), 92-100 (2019), 김태원 외 3인.

위 논문은 센서(Sensing), 인터페이스(Interface), 네트워크(Networking)로 이루어진 IOT기술을 사용하여 실내환경의 온/습도를 측정하는 작업에 대해서 언급하고 있다. 고성능이며 크기가 작은 아두이노 우노 R3(Arduino UNO R3) 모듈과 온습도 센서인 DHT11을 사용하여 측정하고 있다. 또한 센서들을 연결하기 위해서 유무선 네트워킹 기술로 WPAN(Wireless Personal Area Networks), WiFi(Wireless Fidelity)를 사용하고 있다. DHT11센서를 사용하여 발생한 습도 평균 오차율은 4.96%, 온도 평균 오차율은 4.08%로 온-습도 오차율 모두 5%미만의 값을 보였다. 이러한 온습도 센서의 측정값과 특성을 파악함으로서 본 연구에 활용할 수 있었다.

3) 설계의 제약조건 및 문제 해결 방법

(a) 의자에 설치할 물기 제거 장치

본래 모든 의자에 온습도 측정 센서를 설치하여 각각의 데이터를 중앙 네트워크에 송신, 이로부터 보다 정확한 판단이 가능하도록 설계하고자 하였다. 그러나 모든 의자에 해당 센서를 부착하고 실시간으로 데이터를 주고받는 데에 소모되는 에너지와 비용이 상당하기에 비효율적이라 판단하였다. 뿐만 아니라 비는 넓은 지역에 고루 내리는 반면, 의자들이 배치되어 있는 장소의 면적은 대부분 훨씬 작기 때문에 각 클라이언트 상에서 측정한 온습도 데이터에는 통계적으로 무의미한 수준의 차이가 발생할 것으로 기대하였다. 따라서 효용 측면에서 분석하여 볼 때, 온습도 센서를 통한 습도 측정은 한 네트워크 시스템에 한 개만으로도 충분하다 판단하여 설계 방향을 변경하였다. 공원에는 다양한 형태의 팔걸이가 있는 것부터 직사각형 형, 또는 곡선형 구조까지 매우 다양한 형태의 의자가 존재한다. 뿐만 아니라 플라스틱, 나무, 금속까지 의자를 이루는 재질도 굉장히 다양하다. 본 연구에서 제안하는 물기 제거 장치는 그 자체만으로는 이러한 다양한 형태의 의자에 적용하기에는 한계가 있다. 대표적으로, 나무로 만들어진 의자는 그 소재의 흡습성 때문에 표면에 존재하는 물방울을 쓸어 내는 구조로 작동하는 본 장치를 적용할 수 없다. 이러한 범용성을 확보하고자 물기 제거 장치가 본 기능의 효율을 높일 수 있도록 장치를 의자에 탈착 및 부착할 수 있도록 구성하는 것이 아닌, 장치가 의자의 한쪽 모서리에 부착된 일체형으로 설계하였다. 장치가 필요하지 않은 경우에 작동을 하거나 사람이 의자 위에 앉아 있을 경우에도 작동할 가능성이 있다. 이러한 문제를 해결하기 위하여 Arduino 장치의 하드웨어와 알고리즘에 추가적인 수정을 가하였다. 의자에 금속 표면 하단에 압력 센서를 부착하는 것 또한 사람이 앉은 상태에서 기계가 작동되지 않도록

방지하기 위하여 반드시 필요한 장치이다. 이를 실질적으로 구현하기 위하여 의자 밑에 LoRa 수신기와 함께 Arduino Module이 부착된 상태라면, 중앙 네트워크로부터 Wiping 명령이 내려왔으나 의자에 설치된 압력 센서가 $n$kg 이상의 중량을 감지하였을 경우에는 사람이 있다고 판단하여 즉시 그 동작을 멈추도록 코딩한다. 사용자의 편의성을 극대화하기 위한 조치이다.

(b) LoRa를 활용한 비동기 통신과 네트워크의 데이터 송수신 지연 시간 관리

Wi-Fi Direct나 Bluetooth 와 같은 무선 통신 기술의 경우 다중 연결이 간편하다는 장점이 있으나 LoRa의 경우 다소 복잡하다. 1-(a)에서 소개한 바와 같이 중앙의 네트워크는 여러 개의 다중 Node를 통하여 각 클라이언트의 LoRa Receiver로부터 데이터를 제공받고 해당 Node로 다시 요청을 전송하는 과정을 비동기적으로 진행하여야 한다. 본 연구에서 사용하는 LoRa 통신 모듈(SX1272)은 반드시 필요한 비동기적 프로세스를 지원하지 않기에 추가적인 코딩을 통하여 이를 구현하여야 한다. 이러한 문제를 해결하기 위하여 통신 주기와의 연결을 도입하였다. 기본적으로 네트워크 Node를 통한 데이터 송수신은 30분마다 이루어지며 중앙 네트워크가 받아온 날씨 정보에 따라 그 주기는 20분~30분 사이로 조정될 수 있다. 총 $n+1$개의 Node를 통해 통신한다고 한다면, 비동기적 프로세싱은 불가능하나 각 프로세스 사이의 간격을 3초 내외로 조정한다면 데이터 수신에 걸리는 시간은 $3(n+1)$초, 송수신의 양방향 통신에 소요되는 알짜 시간은 $2\times3(n+1)=6(n+1)$초가 된다. 이러한 계산에 따르면 $n \leq 300$개의 Node를 가진 시스템이 지연 없이 충분히 네트워크를 운용할 수 있을 것이라 판단된다.

(c) 서리나 이슬로 인한 네트워크의 취약점 보완 및 효용성 증대

1-(a)의 네트워크 알고리즘에는 2가지의 취약점이 존재한다. 1-(a)에서 6번 항목부터 9번 항목까지를 제외하고 작성한 알고리즘을 가동할 경우 맑은 날 새벽에 발생하는 이슬로 인해 필요 이상으로 비효율적인 운용을 하게 된다. 이슬은 맑은 날 이른 아침에 맺히는 물방울로 바람이 없는 맑은 날 밤에 습도가 높을 때 맺히는 경우가 많다. 이는 온습도 센서에 영향을 미친다. 따라서 이슬이 맺힐 수 있는 새벽 시간대에는 습도 또한 높아질 수 있기에 기존의 알고리즘으로는 적절히 대처할 수 없다. 온습도 데이터를 받는 모든 경우에 장치가 작동하게 되기 때문이다. 이러한 심야 시간대에 장치를 자주 운용하는 것은 이 장치의 주 수요자인 사람들이 거의 없는 시간에 많은 자원을 투자하는 것으로 굉장히 비효율적이다. 이러한 불필요한 에너지 및 작업을 낭비하기 위하여 특수한 알고리즘 처리가 필요하다. 비가 오거나 이슬이 맺히는 현상(서리가 끼는 현상)은 습도가 80% 이상인 상태에서부터 나타나므로 해당 수치인 80%와 시간대를 기준으로 하여 장치의 가동 시간 간격을 조절함으로서 보다 효율적으로 네트워크를 운용할 수 있다.

8. 현재 시간이 24:00~8:00이고 만약, 습도 데이터의 상대습도 수치가 80% 이상일 경우 30분 간격으로 기계를 작동시킨다.

위와 같은 경우에는 밤이기 때문에 습도가 상승하여 이슬이 맺힐 수 있는 환경을 감지할 수 있고 장치의 작동 간격을 30분으로 넓힘으로서 에너지 효용성을 높일 수 있다.

9. 현재 시간이 6:00~9:00이고 현재 온습도 수치가 80% 이상이며 이전 측정 데이터와 상대 습도를 비교하였을 때 변화율이 음이 아닌 실수일 경우 20분 간격으로 기계를 작동시키며 다음 측정 데이터를 기다린다. (서리가 내리는 시간을 고려하였다.)

이 알고리즘을 추가함으로서 아침 시간에 해가 뜨면서 의자에 남아 있던 수분이 증발하면서 사라지는 경우를 고려할 수 있다. 계절과 상관없이 해가 뜨는 시간은 대략 6:00 ~ 9:00 사이이기 때문에 해당 시간대에 있을 때 공기 중의 온습도 수치가 점점 낮아지고 있다면 의자에 남아 있는 수분이 증발하는 것으로 볼 수 있다. 뿐만 아니라 현재 온습도 수치가 80% 이상이라는 점을 추가함으로서 자동적으로 증발하는 수분을 굳이 닦지 않아도 되도록 보다 효율적으로 유도하였다.

(3) 설계 내용

(1) 하드웨어 설계

그림 III-1

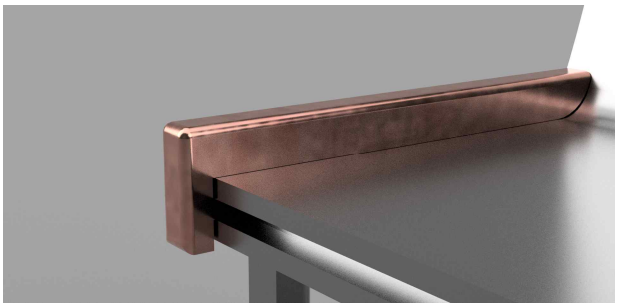

그림 III-2

그림 III-1, III-2 는 본 연구에서 제시하는 의자와 일체형으로 구성된 물기 제거 장치를 그림 II-1의 설계 도면을 바탕으로 3D 모델링한 것이다. 앞서 제시한 바와 같이 의자는 금속 재질로 구성하여 비나 눈이 올 때 그 수분을 흡수하지 않고 표면에 남겨 놓을 수 있도록 구성하였고 추가적으로 장착할 LoRa 수신기와 Arduino Module이 물기 제거 장치가 좌우로 이동할 때 함께 이동할 수 있도록 그림 III-2의 장치를 의자 표면을 덮는 와이퍼의 형태로 구성하였다. 물기 제거 장치는 플라스틱을 소재로 하여 제작하여 쉽게 수분에 노출될 수 있는 환경으로부터 보호할 수 있도록 구성하였다.

그림 III-3

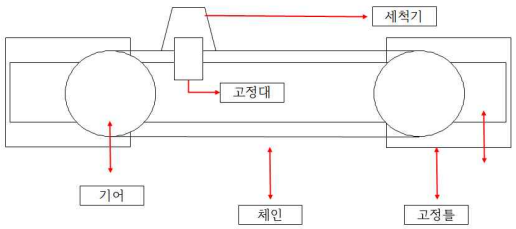

그림 III-4

초기에는 의자 아래에 기계를 움직이는 모터와 체인이 연결되어 있고 체인에 연결된 기어를 통해 장치를 구동시키는 방식을 의도하였다. 그러나 동력장치를 의자 밑에 고정하기 위해서는 의자 가운데에 나사를 고정하여야 하며, 안전에 위협이 되는 요소들이 상당수 존재한다. 이를 해결하기 위해 책상에 사용되는 모서리 보호대에서 착안하여 모터를 고정하는 틀을 고안하였다. 실질적인 형태는 그림 III-3의 책상용 모서리 보호대와 비슷하다. 물기 제거 장치의 모터를 고정하는 틀을 만들어 4개의 고정 틀을 의자 모서리에 붙이고, 고정 틀의 모터 전용 공간에 모터를 장착하였다. 체인은 의자 한쪽 면에만 부착하여 체인에 물기 제거 장치를 붙이고, 앞뒤로 모터를 회전시켜 의자의 물기를 제거하는 것으로 설계를 최종 수정하였다. 그림 III-4는 장치가 의자에 부착되었을 경우의 단면도이다. 그림 III-5은 실제 3D Printing에 사용할 수 있도록 제작한 설계도면이다. (모든 치수의 단위는 mm이다.) III-4에 나타난 것과 동일한 규격의 부품을 4개 제작하고 이를 서로 결합함으로서 최종 장치를 제작한다.

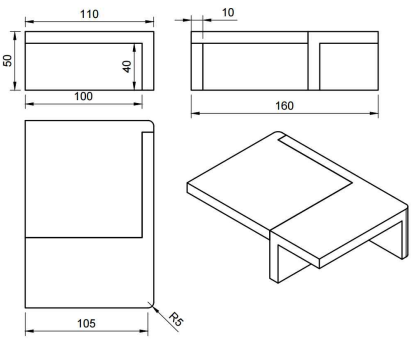

그림 III-5

(2) 네트워크 설계

그림 II-1에 서술한 바와 같이 각각의 Node에 대하여 LoRa 네트워크를 구축하고자 다음과 같은 Arduino 코드를 작성하였다. 아래는 LoRa 네트워크를 사용하기 위하여 Arduino의 사전 설정을 구현한 것으로 중앙 네트워크와의 통신 주기는 600초로 설정하였다.

```
static const PROGMEM u1_t NWKSKEY[16] = { 0xAE, 0xBD, 0x9B, 0xBC, 0x43, 0x41, 0xFB, 0x47, 0x3D, 0xA5, 0x8D,
```

```
0x0D, 0x48, 0x98, 0xEF, 0xCF };
static const u1_t PROGMEM APPSKEY[16] = { 0xE1, 0x51, 0x0F, 0xB1, 0x59, 0x4C, 0xC8, 0xD7, 0xAA, 0x43, 0xAC,
0xE7, 0xE2, 0xBC, 0x5F, 0x1B };
static const u4_t DEVADDR = { 0x260314F7 };
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }
static uint8_t mydata[] = API_Humidity;
static osjob_t sendjob;
const unsigned TX_INTERVAL = 600;
const lmic_pinmap lmic_pins = {
    .nss = 10,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 7,
    .dio = {2, 5, 6},
};
void onEvent (ev_t ev) {
    Serial.print(os_getTime());
    Serial.print(": ");
    switch(ev) {
        case EV_TXCOMPLETE:
            Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
            os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL), do_send);
            break;
        default:
            Serial.println(F("Unknown event"));
            break;
    }
}
void do_send(osjob_t* j){
    if (LMIC.opmode & OP_TXRXPEND) {
        Serial.println(F("OP_TXRXPEND, not sending"));
    } else {
        LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
        Serial.println(F("Packet queued"));
    }
}
void setup() {
    Serial.begin(115200);
    Serial.println(F("Starting"));
    os_init();
    LMIC_reset();
    uint8_t appskey[sizeof(APPSKEY)];
    uint8_t nwkskey[sizeof(NWKSKEY)];
    memcpy_P(appskey, APPSKEY, sizeof(APPSKEY));
    memcpy_P(nwkskey, NWKSKEY, sizeof(NWKSKEY));
```

```
    LMIC_setSession (0x1, DEVADDR, nwkskey, appskey);
    LMIC_selectSubBand(0);
    LMIC_setLinkCheckMode(0);
    LMIC.dn2Dr = DR_SF9;
    LMIC_setDrTxpow(DR_SF7,14);
    do_send(&sendjob);
}
void loop() {
    os_runloop_once();
}
```

아래 코드는 Java를 사용하여 LoRa 네트워크에 접속하고 네트워크 연결을 공유하고 있는 디바이스와의 통신을 시도하고자 차용한 것이다. LoRa 통신 기술을 활용하는 IoT 디바이스에서 사용되는 프레임워크를 활용하여 본 연구를 진행하고자 Libelium의 LoRa 게이트웨이 접속용 Java 코드를 활용하였다. 코드의 양이 너무 많아 아래의 Github Link를 통하여 첨부한다. 해당 소스 파일은 GNU General Public License (Version 2, Revised in June 1991) 의 라이센스를 따르며 본 연구에서는 해당 라이센스를 충분히 읽고 확인하였으며 이에 의거하여 허용된 목적으로 사용되고 있음을 명시한다. [1)]

https://github.com/PascalBod/iot-libelium-lora-gateway/blob/master/src/com/orange/pb/loragatewaycontroller/FrameHandler.java

아래에 첨부한 Java 코드는 중앙 네트워크에서 기상청의 날씨 데이터를 가져오기 위해 사용하는 것으로 JSON을 매개로 하여 기상청의 초단기예보 API를 통해 얻은 XML 데이터를 파싱하고 이 값을 Arduino를 통해 받아온 실제 측정 데이터와 대조, 적정 알고리즘을 따라 명령을 확정할 수 있도록 하는 코드이다. 현재 설계 상에서 의도하였던 알고리즘을 그대로 적용하였으며 Prototype의 실전 테스트를 통하여 알고리즘의 상세한 상수는 조정될 것이다. 표 III-1은 기상청에서 제공하는 기상예보 XML 데이터에 대한 설명으로 해당 데이터셋으로부터 장치 작동 알고리즘을 위한 4가지 데이터(하늘 상태 코드, 강수 상태 코드, 강수확률, 습도)를 받아 오도록 설계하였다.

XML 코드	XML 설명	비고
<?xml version="1.0" encoding="UTF-8" ?>	xml 선언부에 한글 처리(utf-8) 인코딩 선언	.
- <wid>	초단기예보 열기	.
- <header>	지역, 구역 헤더 열기	.
<tm>201003081400</tm>	발표시각:yyyymmddhhMM	.
<sky>4</sky>	하늘 상태 코드	① 1 : 맑음 ② 2 : 구름조금 ③ 3 : 구름많음 ④ 4 : 흐림
<pty>0</pty>	강수 상태 코드	① 0 : 없음 ② 1 : 비 ③ 2 : 비/눈 ④ 3 : 눈/비 ⑤ 4 : 눈
<pop>17</pop>	강수확률 (%)	.
<reh>49</reh>	습도 (%)	.

표 III-1

1) License에 대한 정확한 확인은 Appendix A를 참조하라.

```
public interface WeatherKeyInfo {
    String SERVICEKEY = "servicekey";
    int[][] REGION = {60,127};
    String[] LOCATION = {"Location"};
}
public class DateLoader {
    public String DateLoader(){
        Date date = new Date();
        SimpleDateFormat sformat = new SimpleDateFormat("yyyyMMdd HHmm");
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(date);
        calendar.add(calendar.HOUR, -1);
        String now = sformat.format(calendar.getTime());
        return now;
    }
}
public class HtmlParser {
    public String HtmlParser(String urlToRead) {
        StringBuffer result = new StringBuffer();
        try {
            URL url = new URL(urlToRead);
            InputStream is = url.openStream();
            int ch;
            while ((ch = is.read()) != -1) {
                System.out.print((char) ch);
                result.append((char) ch);
            }
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return result.toString();
    }
}
String now = new DateLoader().DateLoader();
System.out.println(now);
public int humidity; //습도
public int typeCode; //날씨상태코드
for (int j=0; j<REGION.length; j++) {
    weatherStrUrl.setLength(0);
weatherStrUrl.append("http://newsky2.kma.go.kr/service/SecndSrtpdFrcstInfoService2/ForecastGrib?serviceKey=" +
            SERVICEKEY + "&base_date=" + now.substring(0, 8) + "&base_time=" + now.substring(9, 13));
    weatherStrUrl.append("&nx=" + REGION[j][0] + "&ny=" + REGION[j][1] +
"&numOfRows=10&pageSize=10&pageNo=1&startPage=1&_type=json");
    String jsonSouce = new HtmlParser().HtmlParser(weatherStrUrl.toString());
```

```
    try {
        JSONParser jsonParser = new JSONParser();
        JSONObject jsonObject = (JSONObject) jsonParser.parse(jsonSouce);
        JSONObject response = (JSONObject) jsonObject.get("response");
        JSONObject body = (JSONObject) response.get("body");
        JSONObject items = (JSONObject) body.get("items");
        JSONArray item = (JSONArray) items.get("item");
        JSONObject weatherInfo = (JSONObject) item.get(0);
        weatherDto.setBaseDate(weatherInfo.get("baseDate").toString());
        weatherDto.setBaseTime(weatherInfo.get("baseTime").toString());
        weatherDto.setLOCATION(LOCATION[j]);
        for (int i = 0; i < item.size(); i++) {
            weatherInfo = (JSONObject) item.get(i);
            if (weatherInfo.get("category").equals("T1H")) {
                typeCode = weatherDto.setT1H(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("RN1")) {
                weatherDto.setRN1(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("SKY")) {
                weatherDto.setSKY(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("PTY")) {
                weatherDto.setPTY(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("LGT")) {
                weatherDto.setLGT(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("VEC")) {
                weatherDto.setVEC(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("WSD")) {
                weatherDto.setWSD(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            } else if (weatherInfo.get("category").equals("REH")) {
                weatherDto.setREH(Double.parseDouble(weatherInfo.get("obsrValue").toString()));
            }
        }
    } catch (Exception e) {
        e.getMessage();
    }
    weatherDao.insertinfo(weatherDto);
}
public String GetCurrentTime() {
    Date today = new Date();
    SimpleDateFormat format;
    format = new SimpleDateFormat("HH");
    return format;
        }
public int currentTime = GetCurrentTime();
public ExecutionGate(currentTime, humidity, typeCode) {
        if (humidity == 100) && (typeCode == 1) {
            boolean GetPause = false;
```

```
        }
        else if ((humidity < 100) && (typeCode !=0)) {
            boolean GetPause = false;
        }
        else if ((humidity <= 80) && (typeCode == 0)) {
            boolean GetPause = true;
        }
        else if ((8 <= currentTime <= 24) && (humidity >= 80)) {
            int serialEnds = 1800;
        }
        else if ((8 <= currentTime <= 24) && (humidity < 80)) {
            int serialEnds = 1200;
        }
}
        String json;
        StringBuilder sb = new StringBuilder();
        double v1 = 0.0;
        double v2 = 0.0;
        try {
        String location = "서울 은평구 진관동";
        String addr = "http://maps.google.com/maps/api/geocode/json?address=";
        URL url = new URL(addr + URLEncoder.encode(location, "UTF-8"));
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setConnectTimeout(10000);
        con.setUseCaches(false);
        BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream()));
        while (true) {
        String line = br.readLine();
        if (line == null)
        break;
        sb.append(line);
        }
        br.close();
        con.disconnect();
        } catch (Exception e) {
        System.out.println(e.getMessage());
        }
        json = sb.toString();
        JSONObject object = (JSONObject) JSONValue.parse(json);
        JSONArray array = (JSONArray) object.get("results");
        for (Object o : array) {
        JSONObject object2 = (JSONObject) o;
        String lat = ((JSONObject) (((JSONObject) object2.get("geometry")).get("location"))).get("lat").toString();
        String lng = ((JSONObject) (((JSONObject) object2.get("geometry")).get("location"))).get("lng").toString();
        v1 = Double.parseDouble(lat);
        v2 = Double.parseDouble(lng);
```

```
}
double RE = 6371.00877;
double GRID = 5.0;
double SLAT1 = 30.0;
double SLAT2 = 60.0;
double OLON = 126.0;
double OLAT = 38.0;
double XO = 43;
double YO = 136;
```

3. 설계 수행 일정

현재 3D printing을 통하여 물기를 제거하는 장치의 Prototype을 조립하였다. Arduino와 LoRa Network를 결합한 모듈을 통하여 네트워킹과 동작에 쓰일 통신모듈, 모터 장치를 개발 완료하였다. Multiple-node system에서의 테스트를 몇 차례 진행할 것이다. 사용자의 편의와 장치의 원활한 기동을 위하여 금속 의자 제작을 기획하였고 현재 제작 중이다. 해당 의자에 대한 테스트를 추가로 진행함으로서 장치의 효율성을 극대화할 수 있을 것이다.

설계 진행 내용	4월	5월	6월	7월	8월	9월
초기 단계 설계 및 피드백						
선행 연구 조사 및 최종 설계 보완						
설계의 구체화 및 중간보고서 작성						
3D Modeling / Printing 중앙 네트워크 Pseudocode 제작						
Arduino 회로 구성 및 Prototype 제작						
Multiple-Node Test 및 피드백						
최종 보고서 작성						

4. 설계 결과물

(1) 최종 결과물 형상 및 작동원리

그림 4-I to 4-III는 소규모 Prototype의 실물 사진이다. 3에서 서술한 바와 같이 최종 결과물은 의자에 부착되어 있는 일체형 디바이스인데 3D 프린팅을 통해 실제 사이즈의 의자를 제작하기에는 무리가 있어 첨부한 사진과 같이 의자에 부착하게 될 물기 제거

그림 4-I

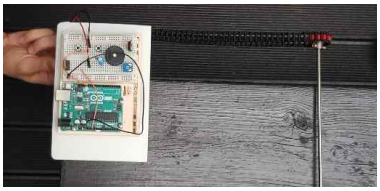

그림 4-II

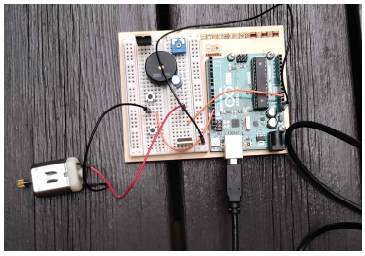

그림 4-III

장치를 제작하였다. 그림 4-I에서 확인할 수 있듯 Arduino Uno와 Breadboard, DC Motor를 사용하여 물기 제거 장치가 좌우로 이동하는 데 필요한 동력을 제공하고자 하였다. 추가적으로 장치에 부착될 의자에는 금속 축을 추가하여 하단에 기어와 체인을 부착하고 이를 바탕으로 장치가 안전하게 이동할 수 있도록 하였다. 그림 4-II는 이를 위에서 본 것이다. 기어의 위치는 의자의 반대쪽 끝이 되어 장치가 체인을 통해 이동할 수 있도록 유도하였다. 이때, Arduino 클라이언트가 네트워크를 통해 적절한 실행 명령을 받을 경우 모터의 회전 방향과 회전 속도가 바뀌면서 장치를 좌우로 이동시키며 의자에 존재하는 물기를 제거하게 된다.

(2) 최종설계 결과물의 장단점 및 의의

최종적으로 설계한 장치는 네트워크를 통해 원활히 통신하며 기상 조건에 따라 정상적으로 동작한다. 이는 네트워크 스스로가 기상 상황을 특정 시간 간격에 맞추어 파악함으로서 굉장히 많은 수의 기기들을 한 번에 편리하게 사용할 수 있도록 한다는 점에서 사물인터넷의 강력함을 잘 보여준다. 기존에 존재하던 특허나 아이디어들은 기계장치를 추가하는 대신 의자의 구조 자체에 변형을 가하였는데 본 연구에서 제시한 제품은 기계적 요소가 포함된 만큼 훨씬 정확하고 안전하게 목표하는 작업을 수행할 수 있을 뿐만 아니라 넓은 범위에 퍼져 있는 기기들을 한 번에 관리할 수 있기 때문에 기존 기기들보다 시간적, 에너지 효율 면에서 훨씬 적합하다. 또한 저전력, 장거리 무선 통신이 가능한 LoRa 기술을 활용하기 때문에 다른 기술에 비하여 에너지 소모가 적고 더욱 오랜 시간 동안 클라이언트와의 연결을 유지할 수 있기 때문에 유지 보수 측면에서도 뛰어나다. 다만, 클라이언트에 사용되는 전력 장치가 전지이기 때문에 주기적인 교체가 필요하다는 점에서는 운용 효율이 그리 높지 않다. 의자와 일체형으로 제공하므로 물기 제거 장치만을 제작할 경우에 발생할 수 있는 표준 규격화 문제의 위험을 줄일 수 있으며 넓은 야외 공간에 여러 대의 의자를 배치해야 하는 상황을 고려한다면 이 제품은 충분히 상용화될 수 있을 것이다. 특히, 기존에는 의자가 젖어 있을 때 직접 물기를 닦는 불편함을 감수하지 않는 이상 앉을 수 없었으나 이 제품을 도입함으로서 그러한 불편함이 해소될 수 있을 것이다.

5. 활용방안 및 기대효과

시내의 공원을 방문하여 잠깐 의자에 앉아서 쉬려고 해도 빗물이 아직 의자에 남아 있어서 앉을 수 없었거나 손으로 직접 물기를 치워야 했던 경험이 존재할 것이다. 본 연구에서 제공하는 장치를 통하여 의자의 물기를 제거함으로써 시민들의 수고가 한결 감소될 수 있을 것으로 예상한다. 의자 위의 중량을 고려하여 일정 무게 이상의 물체가 의자 위에 있다고 판단하여 물기 제거 프로세스를 자동으로 중단하기 때문에 장치의 오작동이나 고장을 사전에 예방할 수 있다. 센서를 통해서 의자 위에 물기가 존재하는지 파악하고 습도 등을 종합적으로 고려하여 청소 여부를 결정하기 때문에 기존에 비하여 훨씬 효율적으로 의자 위의 물기를 제거할 수 있을 것으로 판단된다. 또한, LoRa 네트워크를 통신에 활용하기 때문에 전기를 적게 쓰면서 오랫동안 시스템을 사용할 수 있고 중앙 네트워크와 연결된 한 곳에서만 습도 센서를 설치하여 기상 데이터를 받아 오고 이를 기상청의 값과 비교하여 복합적으로 기상 상황을 판단하기 때문에 각각의 의자에 센서를 달아 습도를 체크하는 것보다 훨씬 효율적으로 시스템을 운용할 수 있다. 추가적으로 기상청의 날씨 정보와 서리가 내리는 시간대까지 종합적으로 고려하기 때문에 날씨를 파악하는데 실패함으로서 발생할 수 있는 오작동을 최소화할 수 있어 공원 의자 등의 공공시설을 관리하는 데 널리 사용될 것이라 기대한다.

<참고문헌>

1. 효율적인 센싱을 위한 LoRa 기술 동향 및 비교분석. 장종욱(2019) 한국통신학회 학술대회논문집, 1384-1385.
2. Event Detection Method for Moving Object Data Stream using Continuous Query Processing based on Grid Index. Jiwon Wee, Seokil Song(2017) 한국콘텐츠학회 ICCC 논문집, 345-346.
3. 실내 콘크리트 격실에서 위치에 따른 LoRa 성능 분석. 임준영, 김동현, 김종덕(2018) 한국통신학회논문지 43(8), 1337-1346.
4. LoRa를 이용한 무선 통신의 성능 측정 연구. 최광호, 전성배, 이충완, Anthony Smith, 이민선(2018). 한국정보과학회 학술발표논문집 1869-1871.

APPENDIX A. GNU General Public License Version 2, June 1991.[2)]

2) License Text Available: https://github.com/PascalBod/iot-libelium-lora-gateway/blob/master/LICENSE.txt