Energy Research and Development Division

FINAL PROJECT REPORT

XBOS-DR: Customer-Controlled, Price Mediated, Automated Demand Response for Commercial Buildings Final Report

California Energy Commission

Gavin Newsom, Governor

September 2019 | CEC-EPC-2019-XXX

PREPARED BY:

Primary Author(s):

Therese Peffer
Gabe Fierro
Marco Pritoni, LBNL
Moustafa AbdelBaky
Daniel Lengyel
Greg Thomson
Anand Prakash, LBNL
Pranav Gupta, LBNL
Callie Clark, LBNL
Aris Athanasios (Thanos) Panagopoulos, CSUF
Jack Hodges, Siemens
Steve Ray, CMU-SV
Irina Krishpinovich, QuEST
Carl Blumstein

California Institute for Energy & Environment, UC Berkeley

2150 Allston Way, #280, Berkeley, CA 94704

(510) 289-4278 http://uc-ciee.org/

Contract Number: EPC-15-057

PREPARED FOR:
California Energy Commission

Jackson Thach
Project Manager

Mikhail Haramati
Office Manager
BUILDING EFFICIENCY RESEARCH PROGRAM

Laurie ten Hope
Deputy Director
ENERGY RESEARCH AND DEVELOPMENT DIVISION

Drew Bohan
Executive Director

ACKNOWLEDGEMENTS

The research team would like to express their deep appreciation to several people.

UC Berkeley: BETS graduate students, including Michael Andersen (creator of several software components including BTrDB and BOSSWAVE/WAVEMQ), Sam Kumar, Kaifei Chen, Jack Kolb, Lukas Spangher, SIMS student Sameer Bajaj; undergraduate students John Leyden, Brandon Berookhim.

Professor David Culler and CIEE researcher Karl Brown.

CMU-SV: Students Prabhu Saitu, Ankit Jain, Rajat Pandey, Dimitris Tzannetos,

Siemens: Student Jason Koh

Two student teams used an early prototype of XBOS as the basis of student projects—one for a user interface course and one for a cleantech to market business course. The interviews conducted for those projects provided a framework for much of this memo. The two projects were:

1. OpenBAS Usability Research for the INFO 214: Needs and Usability Assessment course in Spring 2015. Over the course of the semester the team conducted a heuristic evaluation, diary study, and usability tests on the current web interface, features and functionality. The team also conducted interviews with market experts, software engineers, current users and potential users of the system. The team members were:

- Wenqin Chen
- Bobby Davis
- April Dawn Kester

2. Cleantech to Market course at the UC Berkeley Haas Business School also in Spring 2015 that devoted a semester to researching and developing a tech to market plan for XBOS. The Cleantech to Market team was tasked with identifying the best applications for XBOS as well as a market entry strategy for these applications. The team used design thinking, the lean start-up methodology, desk research, and over 40 interviews to generate an initial list of more than 40 potential applications and a phased market strategy. The team members were:

- Dan Curran, MBA 2016
- Roel Dobbe, PhD EECS 2017
- Robbie Heath, MBA 2016
- Jared Landsman, M.S. Architecture 2016
- John Maus, MBA 2016

PREFACE

The California Energy Commission's Energy Research and Development Division supports energy research and development programs to spur innovation in energy efficiency, renewable energy and advanced clean generation, energy-related environmental protection, energy transmission and distribution and transportation.

In 2012, the Electric Program Investment Charge (EPIC) was established by the California Public Utilities Commission to fund public investments in research to create and advance new energy solutions, foster regional innovation and bring ideas from the lab to the marketplace. The California Energy Commission and the state's three largest investor-owned utilities—Pacific Gas and Electric Company, San Diego Gas & Electric Company and Southern California Edison Company—were selected to administer the EPIC funds and advance novel technologies, tools, and strategies that provide benefits to their electric ratepayers.

The Energy Commission is committed to ensuring public participation in its research and development programs that promote greater reliability, lower costs, and increase safety for the California electric ratepayer and include:

- Providing societal benefits.
- Reducing greenhouse gas emission in the electricity sector at the lowest possible cost.
- Supporting California's loading order to meet energy needs first with energy efficiency and demand response, next with renewable energy (distributed generation and utility scale), and finally with clean, conventional electricity supply.
- Supporting low-emission vehicles and transportation.
- Providing economic development.
- Using ratepayer funds efficiently.

XBOS-DR: Customer-Controlled, Price Mediated, Automated Demand Response for Commercial Buildings Final Report is the final report for the XBOS-DR project (Contract Number EPC-15-057) conducted by the California Institute for Energy and Environment, a representative of the Regents of the University of California (Berkeley campus). The information from this project contributes to the Energy Research and Development Division's EPIC Program.

For more information about the Energy Research and Development Division, please visit the Energy Commission's website at www.energy.ca.gov/research/ or contact the Energy Commission at 916-327-1551.

ABSTRACT

Small commercial buildings, such as retail and offices less than 50,000 square feet, make up the vast majority of commercial buildings in California, and use energy for heating, cooling, ventilating, lighting, computing, and so on. However, small commercial customers (consuming less than 100 kiloWatt (kW)), typically do not have Building Management Systems (BMS) found in large commercial buildings, and thus cannot easily manage energy nor participate in utility reliability events designed to reduce peak electricity consumption.

The goal of the XBOS-DR project was to improve small commercial customer participation in demand response by providing a cost-effective energy management system that allows a wide range of hardware and service offerings as well as effective and automated price-based management. Researchers from Siemens, Carnegie Mellon University's Silicon Valley campus and QuEST joined UC Berkeley in this project to enable integrated customer-controlled, price-based demand-response management. The research team developed a user interface, developed the system architecture, recruited about 20 small commercial buildings, installed the platform (connected thermostats, gateway to energy meter, and miniature computer) in the small commercial buildings, and tested cost-savings strategies in 16 buildings in PG&E and SCE territory in Summer 2018 and 2019.

The platform itself successfully communicated price signals, provided aggregated data (weather, multiple thermostats, whole building data), and allowed different control algorithms to act upon the systems. Installing the networked thermostats alone saved up to 29%. Expanding temperature setpoints during price event days reduced demand 5-25% and reduced costs.

Keywords: small commercial buildings, energy efficiency, demand response, building automation systems, building energy management systems, open-source software, smart thermostats

Please use the following citation for this report:

Peffer, Therese, Gabe Fierro, Marco Pritoni, Moustafa Abdelbaky, Anand Prakash, Irina Krishpinovich, Derrick Rebello, Jack Hodges, Steve Ray. 2019. *XBOS-DR: Customer-Controlled, Price Mediated, Automated Demand Response for Commercial Buildings Final Report*. California Energy Commission. Publication Number: CEC-EPC-2019-XXX.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Improved comfort, fewer complaints, reduced demand charges, convenience of scheduling multiple devices—these are some of the things commercial electricity customers desire. Customers typically do not understand electricity tariffs and do not have the time to manipulate their electrical load, yet electric utilities increasingly need improved integrated distributed resources to maintain a resilient grid. This project proposed to develop a standards-compliant open-source management system that promotes customer choice as well as provides predictable demand.

Introduction

Small commercial buildings, such as retail and offices less than 50,000 square feet, make up the vast majority of commercial buildings in California, and use energy for heating, cooling, ventilating, lighting, computing, and so on. However, small commercial customers (consuming less than 100 kiloWatt (kW)), typically do not have Building Management Systems (BMSs) found in large commercial buildings, and thus cannot easily manage energy nor participate in utility reliability events designed to reduce peak electricity consumption. Large commercial customers have BMSs that can control Heating, Ventilation, and Air-Conditioning (HVAC) equipment and often lighting in order to respond to utility price or event signals, to reduce the electricity demand especially during hot weather. While demand response solutions abound for residential customers—communicating thermostats, for example—few solutions address the complexity and heterogeneity of the needs of small commercial customers.

The increasing shift from natural gas-fueled to electric-powered appliances and growing number of solar electric generation has created the need for greater flexibility in electrical loads to provide grid stability and reliability. The goal of this project matches the goal of California's Public Utilities Code 8360 to develop and incorporate cost-effective smart technologies toward demand response and energy-efficient resources, and provide consumers with timely information and control options. This project aligns with the integration of distributed energy resources outlined in the CPUC Section 769 and R.14-08-013, addressing the problem of lack of access to data and providing customers choices. In providing the underserved small commercial building sector with energy efficiency and demand response capability along with customer choice to improve participation rate, this project overcomes a major barrier to achieving several of the state's energy goals. This project provides a cost-effective solution towards both AB 758 and SB 350, which govern energy efficiency in existing buildings.

Most services—heating, lighting, security—in a building cannot communicate with others because they do not share the same communications protocol, vendor variants are incompatible, or no interfaces exist. The market is already transitioning to open protocols, since they allow more manufacturers to sell and service the market and Internet Protocol (IP)-enabled devices can ease this transition. The technology associated with the XBOS-DR system is not likely to be provided by the competitive market because the code is open source—anyone can use, modify and redistribute it without charge. This makes it difficult for a private interest to capture the benefits from an initial successful development. Thus public funds from the Energy Commission were necessary for this research.

Project Purpose

The goal of the project was to improve commercial customer participation in providing electric grid resilience by enabling cost-effective management and integration of demand response with other building services in small commercial buildings. To achieve this goal the researchers built and pilot tested an energy management system on the eXtensible Building Operating System (XBOS) software platform. The system connected networked thermostats and electrical interval meters with a price signal, and developed a user interface that allowed notification, monitoring and control. The research included the development of a messaging system, an information exchange platform, and a negotiation mediator.

The high cost of technological upgrades—such as replacing equipment with energy efficient products—represent a major barrier to energy efficiency goals; a software solution that relies on simple hardware and can leverage existing systems is a much less expensive and nimble approach to achieve energy goals. Another major barrier is single-vendor, single-communication-protocol integrated solutions. The open architecture of XBOS-DR, augmented by the price-response messaging system, information exchange, and price-based optimization components, can foster technical innovation by third-party vendors and HVAC, lighting, and other manufacturers in providing energy services, and can provide a mechanism for evaluating and optimizing building demand strategies based on pricing information.

Researchers from Siemens, Carnegie Mellon University's Silicon Valley campus and QuEST joined UC Berkeley in this project to enable integrated customer-controlled, price-based demand-response management. The research team hoped to identify and address technical and social barriers to enabling demand response in small commercial buildings. The team hoped to develop the platform XBOS to provide a virtual BMS for small commercial buildings by networking multiple devices such as thermostats, plugload controllers, EV charging, and interval meters. The team also hoped to develop the messaging system and information exchange components, which ensure standards compliance and system interoperability with various hardware devices or management systems.

The ratepayers of California will appreciate the research targeting the underserved small commercial customer market, especially to support energy efficiency and demand response in existing buildings, which can reduce energy costs, improve comfort, reduce greenhouse gases, and improve grid resiliency.

The anticipated audience of this research are small commercial customers, energy service providers, utility program developers, energy data analysts, and other researchers working to improve HVAC control systems.

The first technical project objective was to design and develop customer value, using user surveys and use cases to develop applications and the user interface. The second objective was to define, design, and develop five major components of the system architecture: 1) the negotiation mediator that aggregates and coordinates price signals—whether none (e.g., TOU), one way or two way—and energy usage among many sites, 2) a messaging system to convey pricing schedules to the BMS and the BMS demand requirements to the rate supplier, 3) an

information exchange module to manage disparate message content, 4) developments within the XBOS context to support a variety of networked devices and support price-responsive building controls, and 5) applications such as optimization to manage demand based on pricing schedules and software to manage load diversity. In the third objective, the project tested at the laboratory scale in order to develop a version that is robust and secure. For the fourth objective, the research team pilot tested XBOS/DR in 13-19 commercial buildings in California in order to measure and verify energy savings and evaluate the project, especially customer satisfaction. Pilot test data was analyzed both to evaluate the performance of the software and to identify strategies that can improve performance. Continuous objectives included finding and fixing software bugs, refining control algorithms based on experience with pilot tests, evaluation of project benefits and technology, and knowledge transfer activities. The final objective was to develop a product and market readiness plan.

Project Approach

The California Institute of Energy and Environment at UC Berkeley was the prime recipient and managed the project and developed the user interface. The UC Berkeley Computer Science research organization (the Software Defined Buildings group later renamed the Berkeley Energy and Transportation Systems (BETS) research group) developed the XBOS-DR platform including the microservices, building models, and advanced controls. Researchers at Siemens and Carnegie Mellon University-Silicon Valley worked closely to develop the price messenger, information exchange module, and negotiation mediator. QuEST recruited the small commercial customers, installed the system, and managed the relationship with the customer.

In identifying and developing value to the customer, the research team employed a literature review and interviews to understand the types of users of the platform (e.g., business owner, employee, janitorial staff) and a range of values (e.g., comfort, convenience, cost-savings). The team then developed use cases, and developed a user interface that showed the energy consumption, indoor temperature, provided a slider bar for the customer to select the desired balance of cost savings versus comfortable conditions, and provided simulations so the user could understand the implications of the cost-comfort selection.

The system architecture of the XBOS-DR and EPIC platforms were modular, so that the various components could be developed and tested separately. The Siemens and CMU team developed the price messenger and tested it with both simulated and real utility event signals. The information exchange module included many standards-based information models. The BETS team developed the XBOS-DR platform: creating the drivers (interfaces) for various hardware, such as the thermostats, outlining and developing more than 10 microservices, and upgrading security services. They helped install, integrate, and maintain the hardware and software, developed an improved method of storing, categorizing, and acquiring the data through the Brick schema, and developed models for each building, and tested advanced control algorithms such as Model Predictive Control.

The research team used laboratory spaces on campus and three offices off campus as initial laboratory testing of the networked thermostats, and electric utility meter, and in one case, Electric Vehicle charging and various kitchen appliance loads. After recruitment and initial

building audit of several buildings, the team installed the platform consisting of a miniature computer, networked thermostats, and metering in 16 small commercial buildings: 14 in Pacific Gas and Electric territory and two in Southern California Edison territory. For each building, the team procured historical electricity usage for the previous 24 months. After the system was installed, the team commissioned the equipment and monitored the data. In the summer of 2018, the team conducted several tests during peak day events in many of the buildings; some buildings were not able to participate. The team worked to analyze the data, correct data quality issues, build models, develop simulation tools, and develop zone-level analysis tools. In summer 2019, the team conducted a couple of tests to refine the procedure.

The technical barriers included intermittent Internet and other networking issues, data quality, insufficient Application Programming Interface for the networked thermostats, and the short range of the environmental sensors. Non-technical barriers included misunderstandings and miscommunication between the subject customers and the project team, unanticipated length of time to develop the user interface, and difficulty in obtaining the utility price signal directly from the utility.

The networking issues caused the XBOS-DR development team to reconstruct the secure data bus to accommodate local control; a couple of times, the customer's IT team made changes to the internal networked which removed the connection of the research team's system. The Green Button data was continuously collected throughout the project, but often had missing data; the team developed a script to detect the missing data. The team also had to find ways to work around the thermostat API to achieve the needed functionality (e.g., access the programmed schedule, force the system into first stage cooling). One of the TED meters was wired backwards and needed to be rewired; another TED meter malfunctioned and needed to be replaced. One of the customers informed us that the building was a designated cooling center, so the team could only control six of the HVAC zones; two customers (firestations) asked that the team not change the temperature setpoints, which essentially eliminated those buildings from the study. The project team found a couple of ways of receiving the price signal, including scraping the data from the internet.

The research team presented the project in a number of venues to industry that included technical advice. The feedback included the need for security, desire to use other databases, ability to add other diagnostic and control modules, and need for some training to use the system.

Project Results

In general, the researchers achieved project goals in using networked thermostats to reduce energy consumption, demonstrating the prototype XBOS-DR platform to reduce peak loads on event days in small commercial buildings, and demonstrating the price messaging and information exchange module functionality. The team was able to integrate lighting reduction in one building and EV charging and appliance loads in another building. The research team was not able to integrate photovoltaics and storage in the commercial buildings due to the insufficiency of the interface of the existing systems.

The research team identified several major lessons learned. There is an incredible value of incorporating real-time building energy data with thermostat data: one can achieve building system identification, conduct diagnostics, and improve control. The research team learned the difficulties in commissioning and managing multiple buildings. The team acknowledges more time is required to develop a user interface and address data quality issues.

Additional research that would further the goals of the project would be further testing of the user interface, testing the system with a different networked thermostat with an API that provides the functionality desired, and to continue to deploy and test different control and diagnostic strategies.

Technology/Knowledge Transfer/Market Adoption (Advancing the Research to Market)

The research team published seven papers so far on the project, have presented over ten times to various audiences, and have released the software on public sites (Github). The team has developed four patents. The platform is currently included in three ongoing research projects, and the team research intends to obtain more funding to continue to develop the platform in order to prepare for uptake into manufacturers' products. In addition, the project trained and employed 20 students and two post-docs.

The near-term target market is research groups who need access to data for analytics or for testing control. Mid-term target market is utility programs, startup companies, and other early adopters who see value in secure access to building and systems data. The long-term target market is adoption by control companies to advance small and large commercial building controls.

Benefits to California

This project is anticipated to result in the ratepayer benefits of greater electricity reliability and lower electricity costs through enabling more effective use of DR and distributed generation resources, to help manage issues anticipated from: 1) the increasing integration of intermittent power generation into California's grid under the state Renewable Portfolio Standard program, and 2) related issues anticipated from increasing electric vehicle charging. The project technology is also anticipated to increase penetration of building management systems in smaller commercial buildings, resulting in energy cost reduction through improved end-use energy efficiency. In addition, increased safety is expected for energy end-use equipment through increased capability for remote monitoring and potential integration with alarm services.

The technology could be adapted to the residential sector at minimal costs; different drivers would need to be developed depending on the types of hardware included.

The project estimated the following specific impacts and benefits of the proposed aggregated demand-response and integrated energy management program, with supporting rate design and open-architecture software platform for large and small commercial sectors:

- Greater reliability of the electricity infrastructure, reducing frequency of outages.
- $260 million per year reduction in energy costs for ratepayers in 2024—derived from: lower demand charges (reflected for the utility as lower electricity infrastructure upgrade

and operating costs), increased electric grid energy efficiency, reduced energy end-use from persistent efficiency in parallel with DR, and lower electricity generation costs (from lower-cost intermittent greenhouse gas (GHG)-free electricity generation with less need for spinning reserve, storage or high-cost supplemental peaking generation).

- 450 MW of avoided or shifted peak electric demand in 2024. This is a 150% increase beyond the 293 MW of DR from a combination of nonevent-based programs, critical peak pricing, and peak-time rebates estimated by the California Energy Demand 2016-2026 Revised Forecast [15].
- 180 million kWh per year and 18 million therms per year of reduced energy use in 2024 from persistent end-use energy efficiency achieved in parallel with demand-management.
- 930,000 metric tons of carbon dioxide (CO2e) emissions per year avoided in 2024 from: increased electric grid energy efficiency, increased end-use energy efficiency in parallel with demand-management, and increased fraction of intermittent operationally GHG-free renewable electricity generation (and decreased need for GHG-intensive supplemental peaking generation).

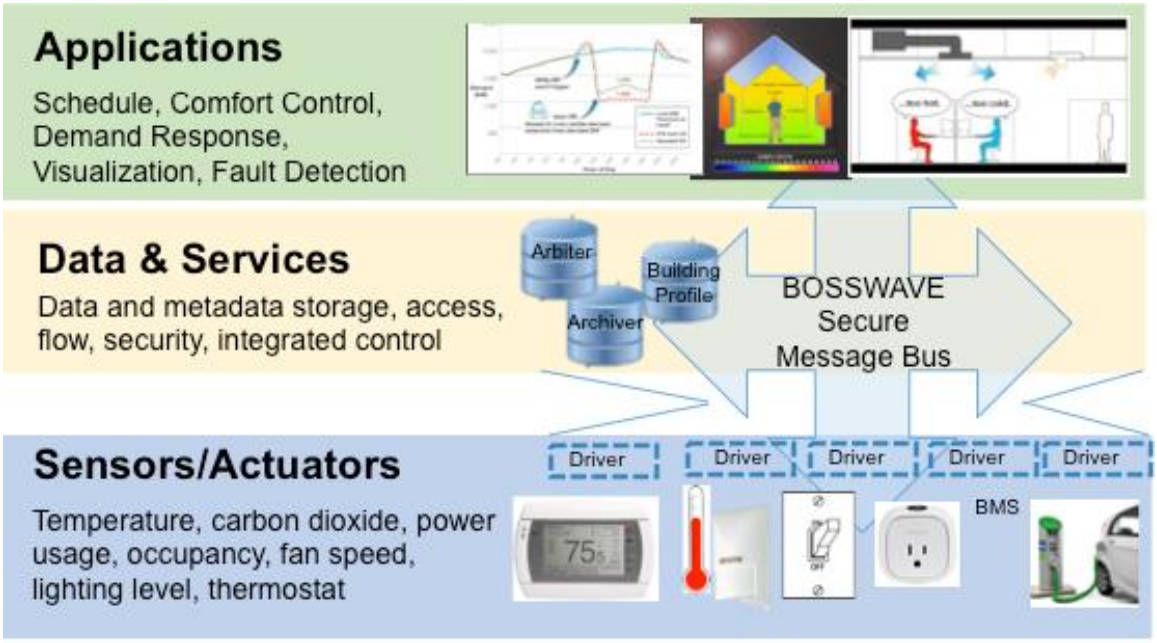
CHAPTER 1: Introduction

The services demanded of commercial building customers—heating, cooling, ventilating, lighting, computing, and so on—require significant energy and contribute to peak energy demand. In fact, while demand response solutions abound for residential customers—communicating thermostats, for example—few solutions address the complexity and heterogeneity of the needs of commercial customers. Large commercial customers (consuming 1 megawatt (MW) and greater than 50,000 square feet (sf) typically have a Building Management System (BMS) that can control Heating, Ventilation, and Air-Conditioning equipment and often lighting in order to respond to price signals, although this requires some communication and controls infrastructure. Small commercial customers (consuming less than 100 kW and smaller than 50,000 sf, however, typically do not have BMS, and thus cannot easily participate in demand response.

The team of researchers led by UC Berkeley's California Institute for Energy and Environment includes the Building Energy Transportation Systems research group in Computer Science, Siemens, Carnegie Mellon University—Silicon Valley, and Quantum Energy Services & Technology (QuEST). The team proposed to create a *Customer-controlled, Price Mediated, Automated Demand Response for Commercial Buildings* (XBOS-DR). This project proposed to develop a Demand Response manager based on the eXtensible Building Operating System (XBOS/DR) (an open source and open architecture platform) that can interface with multiple hardware devices from different vendors as well as include software applications from various vendors. With its ability to create a virtual BMS for small commercial buildings by networking thermostats and other controllers, XBOS-DR can provide large and small commercial customers with a variety of choices for demand response capability.

The XBOS platform provides an open software "layered" architecture, similar to that found in a smart-phone, where different applications (Apps) are supported by an Operating System and data management services and interacts with the hardware. With this open-source tool, any time-series data stream—whether online weather, third party sensors, hardware devices, or data from the BAS—can be labeled or tagged and stored in a fast, easily queried database. One can add new data streams by writing a simple "driver" interface. Applications can access this rich database and provide improved actuation, visualization, or optimization. An open source and open architecture enabling platform runs counter to the business model of many companies, who want to maintain a single vendor, proprietary solution.

Figure 1: Horizontal Layers of an Open Software Architecture



Horizontal layers define interoperability compared to vertical silos.

Credit: Carl Blumstein, Therese Peffer

This final report describes the development and testing of XBOS-DR in multiple buildings. With its ability to create a virtual BMS for small commercial buildings by networking thermostats and other controllers, XBOS-DR can provide small commercial customers with a variety of choices for demand response capability.

Goals of the Project

The goal of the project was to improve commercial customer participation in demand response by providing customer value: engaging service choices such as lowered demand charges and improved comfort and convenience and an easy-to-use interface at a low cost.

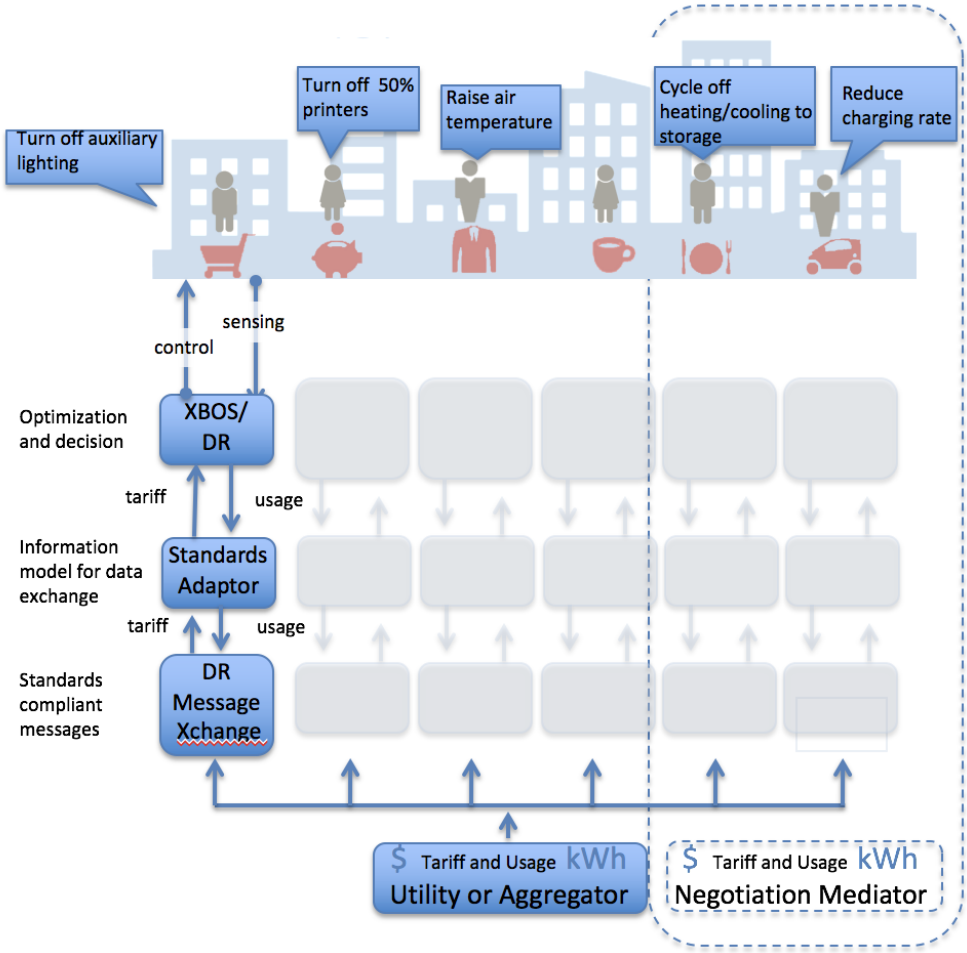
The objectives of this project were to:

- Design and develop customer value, with user surveys and use cases to develop applications and the user interface.
- Define, design, and develop all components of the system architecture:
    - A negotiation mediator that coordinates among several building sites.
    - a messaging system to convey pricing schedules to the BMS and the BMS demand requirements to the rate supplier,
    - an information exchange module to manage disparate message content,
    - developments within the XBOS context to support a variety of networked devices, and

- applications such as optimization to manage demand based on pricing schedules.

- Test the alpha version at the laboratory scale in order to develop a beta version that is robust and secure.
- Pilot test XBOS/DR in at least 20 commercial buildings in California in order to measure and verify energy savings.
- Evaluate project benefits and technology,
- Develop knowledge transfer activities and
- Produce a product and market readiness plan.

Figure 2: Components of XBOS-DR Platform



Small commercial customers (top) have different needs and different values regarding reducing peak electrical loads. Price information (tariff and demand response event signal) from the utility is received by the price messaging system, and relayed to the XBOS-DR platform, which sends control signals and receives information (e.g., electrical power consumption, indoor temperature). The Negotiation Mediator can take a subset of buildings and control based on a desired aggregated load profile.

Credit: Therese Peffer and Jack Hodges

This project provides a holistic solution in addressing nine key requirements that a customer-controlled, price mediated, automated demand response manager should:

1. Receive pricing signals, evaluate energy demand, and respond with managed demand requirements, while complying with regulatory requirements,
2. Enable heterogeneous customers to adapt demand response with individual preferences for each commercial building,
3. Track, evaluate, and control multiple heterogeneous devices,
4. Interoperate with various building systems, such as Building Management System (BMS) in large commercial and networked thermostats in small commercial to enable coordination of building management with the operation of other building systems and to take advantage of sensors connected to the BMS, enhanced by the use of emerging smart grid standards,
5. Retain the electrical usage history of connected devices. An electrical energy data historian, or means of storing energy data, provides the basis for understanding usage patterns, which is essential for effective demand optimization and management,
6. Provide pricing-based load management algorithms based on a variety of metrics including load type, existing schedules, service prioritization, and historical demand,
7. Coordinate to maintain load diversity. Coordination can be achieved through price signals but if many consumers automatically and simultaneously respond to the same price (e.g., a TOU price), loads may become synchronized—switching on and off simultaneously in ways that destabilize the system.
8. Provide security. The system must be secure from disruptive intrusions or theft of confidential information, and
9. Provide customer value. Because demand-response is implemented at a relatively small scale in many commercial buildings, software must be inexpensive (preferably free) and easy to install and maintain. To enhance this value the software should also provide other services such as management of demand charges and turning off equipment during non-business hours.

The project developed the customer value; define, design and develop the system architecture, modules, and applications; make the software robust through testing, and then pilot test in at least 20 small and large commercial buildings.

Report Organization

The chapters of this report follow the task structure of the project. The second chapter describes the **Project Approach** (**Customer Value**: the needs and preferences of various commercial customers, garnered through surveys, interviews, and other means, in order to develop the software tools and interfaces necessary to provide customer value in managing energy demand, the **Front End Services** of the **System Architecture**: the price messaging system, the information and compliance transparency module, and the mediator, and the

efforts to harden or making these software elements robust and secure, the development within the **XBOS platform** to support a variety of networked devices, and applications, and the hardening to make the XBOS-DR platform robust and secure, and the **Pilot Testing** of the platform in several buildings to test the ability of the system to control the operation of the devices, to save energy, to reduce peak electrical load, and to engage customers. The third chapter describes the **Project Results**. The fourth chapter covers the Technology Knowledge Market Transfer Activities. The fifth chapter provides a **Conclusion and Recommendations**. The final chapter outlines the benefits to California ratepayers.

CHAPTER 2:
Project Approach

This chapter describes the approach of the project: discussion of **Customer Value**, **System Architecture** (both the **Front End Services** of the **System Architecture** and the XBOS platform), and the **Pilot Testing** of the platform in several buildings to test the ability of the system to control the operation of the devices, to save energy, to reduce peak electrical load, and to engage customers.

Customer Value

This section outlines the needs and preferences of various commercial customers, garnered through surveys, interviews, and other means, in order to develop the software tools and interfaces necessary to provide customer value in managing energy demand).

Electricity customers typically do not understand tariffs and do not have the time to engage in bidding, shedding or shifting load, yet the purpose of this project is to help electric utilities who need improved integrated distributed resources to maintain a resilient grid. For the electrical utility to engage small and large commercial buildings in demand response programs to reduce electrical peak power demand periods and improve grid reliability, the demand needs, values, and priorities of these various businesses must be considered. Small commercial buildings include grocery stores, banks, retail, cafes/restaurants, offices, and others. Each business has different appliances, HVAC, or lighting loads, a different load profile over the course of the day, and different needs with respect to a business model. This chapter describes efforts in understanding who potential customers are, what are their values, needs and preferences, and how can the researchers include these in design and outreach decisions. More detail may be found in Appendix A.

Customer Values, Needs and Preferences

The purpose of this subsection is to briefly outline the types of customers that might use the XBOS-DR platform, and their values and preferences. Two student teams worked to develop information on potential customers; this subsection draws in part on their work both in developing usable interfaces as well as path to market. The researchers used these values to develop use cases for the XBOS-DR platform and inform the development of the user interface that promotes customer choice in responding to price signals.

The **target users** identified for this type of technology are primary users (business owner, building owner, office manager, property manager, building manager, bill payer, occupants such as employees and clients) and secondary users (janitorial staff, contractor (service provider for maintenance), vendor/installer of equipment, or energy services provider.

Key issues identified are: split incentives (owner/tenant), relationship between the building owner and service contractor that maintains the building, labor-intensive requirements to install new equipment, proprietary and rigid existing Building Management Systems, lack of

knowledge and training of installing contractor (especially with regard to information technology e.g., network systems, Internet of Things connected devices, firewalls, Internet provider).

Each user has his or her own values related to this new service. These values include:

- **Safety:** adequate light, fresh air/ventilation, appropriate thermal conditions, furnishings
- **Security:** doors lock, controlled access
- **Privacy:** restricted access to building data and control
- **Low risk:** in terms of insurance for physical premises
- **Comfort**: thermal comfort, good light, little noise, good air quality
- **Convenience:** time savings measure (remote control, ease of maintenance)
- **Cost:** rent, salaries (productive employees!), demand charges, equipment, maintenance, energy, and balance of all of these.

Customer Use Cases

The researchers used these values to develop use cases that represent the ways these customers might use the XBOS-DR platform to meet their needs and preferences. Reviewing the various types of businesses represented in the potential customers, three types of buildings—office, retail, and education—have shown the greatest energy efficiency potential (16-33%), followed by healthcare and lodging. Lighting, HVAC, and computers showed the most promise as targets of energy efficiency measures; hot water was another end use that showed potential for energy savings. Finally, a prioritized list of use cases provides scenarios using XBOS-DR to meet these needs.

The ultimate goal of the project is to provide price signals to the XBOS-DR platform in various buildings, and have the XBOS-DR platform of each respond with a forecasted demand (e.g., not to exceed power demand per hour, 90% confidence interval for demand). The system uses machine learning for predicting and forecasting load. The team brainstormed potential functions of the XBOS-DR platform for both small commercial and larger commercial buildings. Categories of functions included HVAC, lighting, plugload, process load, storage, and human activities. The major difference between small and commercial buildings is in the HVAC system.

The primary control strategy was to change thermostat setpoints to reduce air conditioning runtime. The team also explored reducing the runtime of second stage cooling and alternating multiple Roof Top Units (RTUs) to reduce peak demand.

Customer Facing Functions

The researchers developed the use cases in functionality of the user interface and control platform. The most common use cases are: 1) monitor and control energy use manually, remotely, conveniently (such as check current temperature and modify temporarily during the

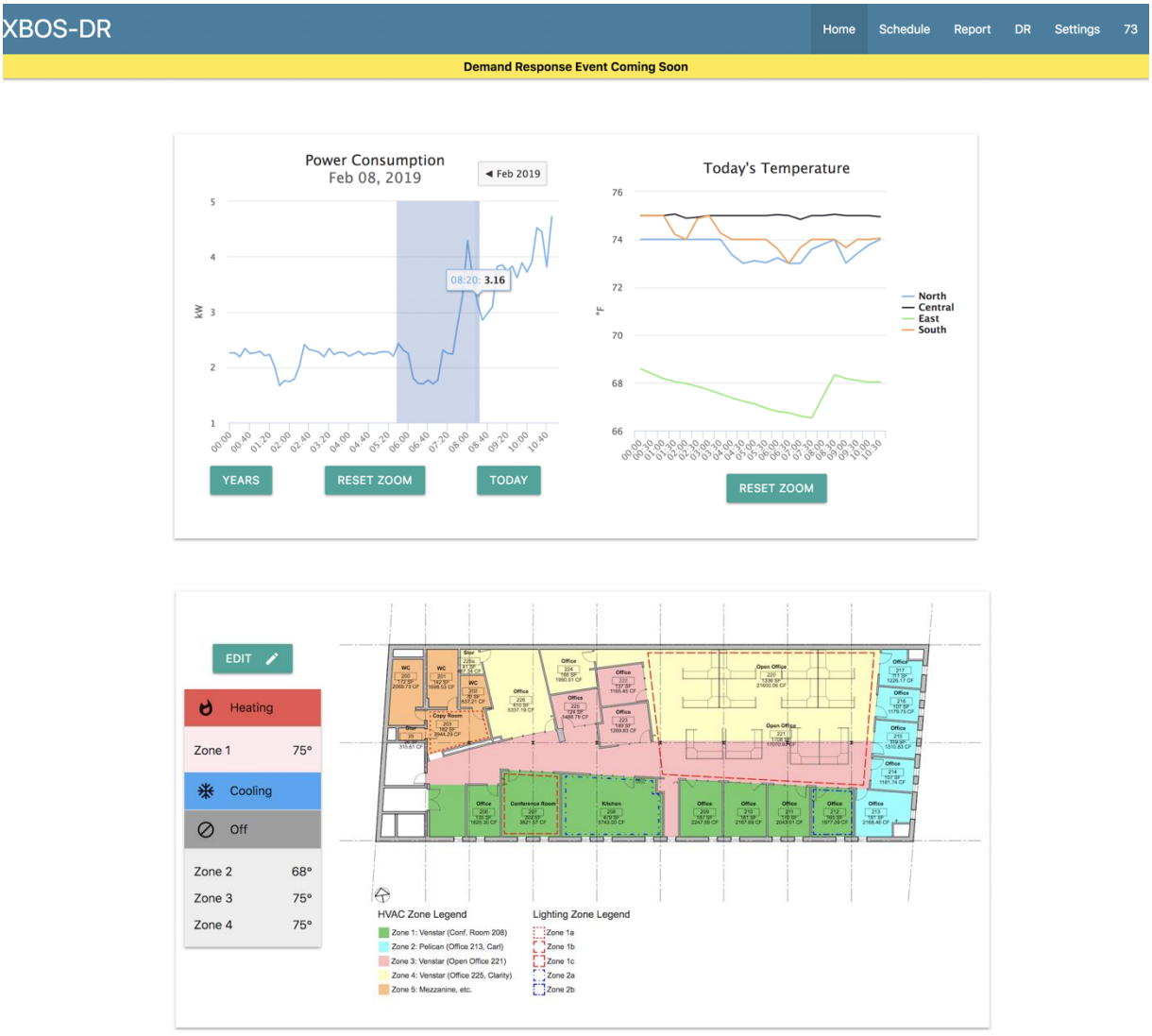
day, for holidays, or seasonally), 2) engage in demand response (receive alert, choose level of engagement at the zone level), 3) manage energy use to reduce energy cost, and 4) check diagnostics.

User interface

The research team developed a user interface for the project. The main sections of the user interface are the Home screen, Schedule screen, DR screen, and Settings.

The home screen shows the overall energy usage of the subject property, the temperatures in each zone, the floor plan of the area, and the ability to change the zone temperature setpoint temporarily.

Figure 3: Home Screen of User Interface

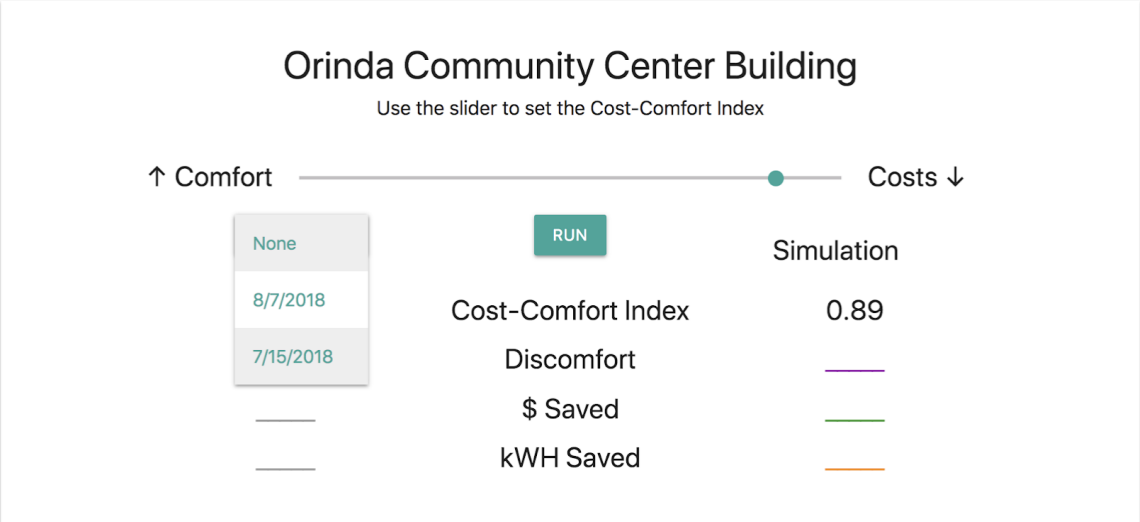


The home screen shows notification of events as well as basic energy and temperature information.

Credit: Therese Peffer and Brandon Berookhim

The Demand Response screen allows the user to simulate the loads of the building during a DR event, and select a level of disruption using the Cost-Comfort Index slider, which balances the temperature setpoint selection (within a safety range) from the base case (full comfort on the left) to reduced cost (and potentially slightly reduced comfort) to the left.

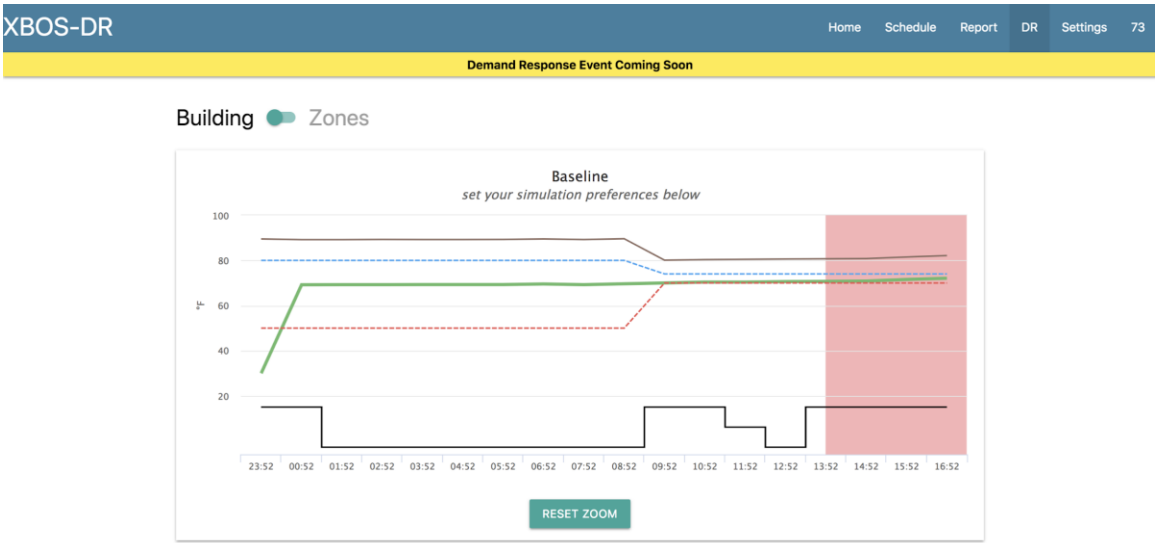
Figure 4: Cost-Comfort Index on DR screen of User Interface



The cost-comfort index allows one to select and simulate the desired level of balance between saving money or remaining comfortable during a demand response event.

Credit: Therese Peffer and Brandon Berookhim

Figure 5: Display of Simulated Data on DR screen of User Interface



The cost-comfort index allows one to simulate the effect on indoor temperatures, setpoints, and estimated energy consumption.

Credit: Therese Peffer and Brandon Berookhim

Front End Services of the System Architecture

This section describes the EPIC platform, which includes the price messaging system, the information and compliance transparency module, and the mediator. Full details are in Appendix B.

Develop the Price Messaging system

The Price Messaging module implements the default workflow of transactive signals from the utility to the building management system, and the forecasts back from the building management system to the utility. Since research partners generating price or DR event signals (EPRI, PG&E and SCE) were not interested in receiving any responses from our systems (such as demand forecasts), interactions with outside energy suppliers consisted solely of receiving (and not sending) price and demand response signals. Accordingly, the only compliance requirements were to support the communications, security and authentication specifications needed to poll for such signals. This was done using the OpenADR protocol version 2(b) and security keys provided by a third party.

The researchers subscribed to both the PG&E and SCE signals that EPRI generates, first with test signals and servers and later with their production servers where possible, and also implemented two additional signals: a signal generated from the commercial tariffs and the Peak Day Pricing (PDP) or Critical Peak Pricing event signal generated from PG&E or SCE respectively.

The researcher conducted several integration tests with XBOS-DR: 1) simple round trip communication, 2) live signals from Group 3 EPRI signal, followed by PG&E and, later, SCE, 3) pricing event forecasts, 4) test the Brick/data retrieval API for the Negotiation Mediator, and 5) the last was to integrate the Negotiation Mediator.

Develop the Information Exchange Module

The Information Exchange/Management module is the core component of the EPIC platform. It was designed to provide a system-agnostic information model that could enable interactions between the grid and building management systems, or between heterogeneous building management systems, which traditionally use different models. Model integrations serve the purpose of allowing the use of system-agnostic standards and other models to appear as a single database structure. The following information models were selected to integrate:

- FSGIM (ASHRAE 201) - Energy models (from the standard, in OWL)
- OpenADR 2.0b - Price messaging model (from the standard, but converted from XML Schema to OWL)
- QUDT - Standards-compliant quantities, units, and dimensions models (from the standard, version 1.2)
- IFC - Building models (from the standard, in OWL)
- SAREF - Building models (from the standard, in OWL)
- SSN/SOSA - Sensor/Actuator models based on the SSO pattern (from the standard, in OWL)

- SSF - Sensor and Actuator models (adhoc, in OWL)
- EPIC Bldg - A building model that includes services (e.g., electricity, plumbing, etc.) and tariffs (created from a combination of sources, including a pre-standard tariff model from NIST, and Schema.org)

Model integrations are done in two ways: (1) between ontologies that are part of the SAIM, and (2) between the models in the SAIM and legacy models such as BRICK. The mechanism for ontology mapping is the same in both cases.

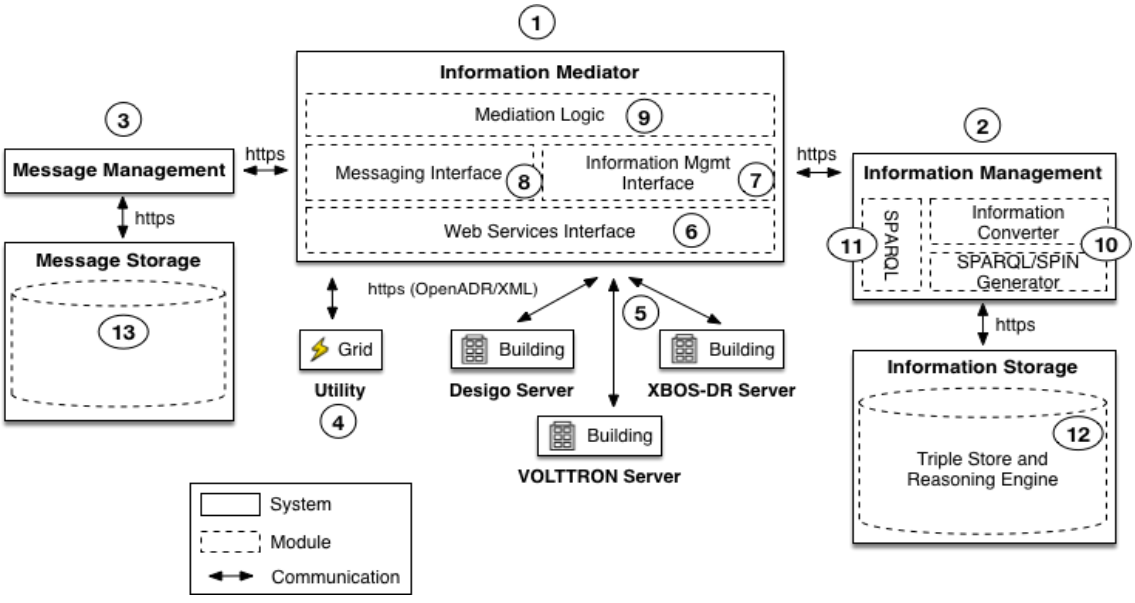
- FSGIM <-> OpenADR
- FSGIM <-> QUDT
- IFC <-> QUDT, SSN/SOSA, SSF, EPIC Bldg
- SAREF <-> QUDT, SSN/SOSA, SSF, EPIC Bldg
- BRICK <-> QUDT, SSN/SOSA, SSF, EPIC Bldg

The bi-directional arrows represent integration mappings or adapters.

Negotiation Mediator

The Negotiation Mediator was originally proposed as an autonomous computational unit that managed the interactions between the utility and buildings, but it was later changed to be a service running on the EPIC platform which, in the architecture diagram below, is now referred to as the Information Mediator. The Negotiation Mediator becomes active when a negotiation use case is identified. The Price Messaging System is the interaction manager. A high-level view of the architecture is shown in the figure below:

Figure 6: Components of Negotiation Mediator



The components of the Negotiation Mediator include the Information Mediator that looks at predicted energy consumption data from multiple buildings, Information Management and Storage, and Message Management and Storage.

Credit: Jack Hodges

A brief walkthrough of this architecture follows. The primary component of the platform is the Information Mediator (noted at 1 in the figure above), which serves as a common entry point and provides web service access points to the Information Exchange/Management services (at 2, through an interface at 7), and the Message Management (at 3, through an interface at 8). Interactions between the utility (at 4) or buildings (at 5) are mediated by the platform (at 6). Examples of different possible building management systems are shown in the figure, though in this project only XBOS-DR servers are interacting with the platform. The mediation logic component (at 9) is part of the platform. Other aspects of the architecture, namely security, storage/retrieval (shown at 10-13) will be discussed elsewhere in this document. For a full discussion of the architecture please see the EPIC Technical Specification document.

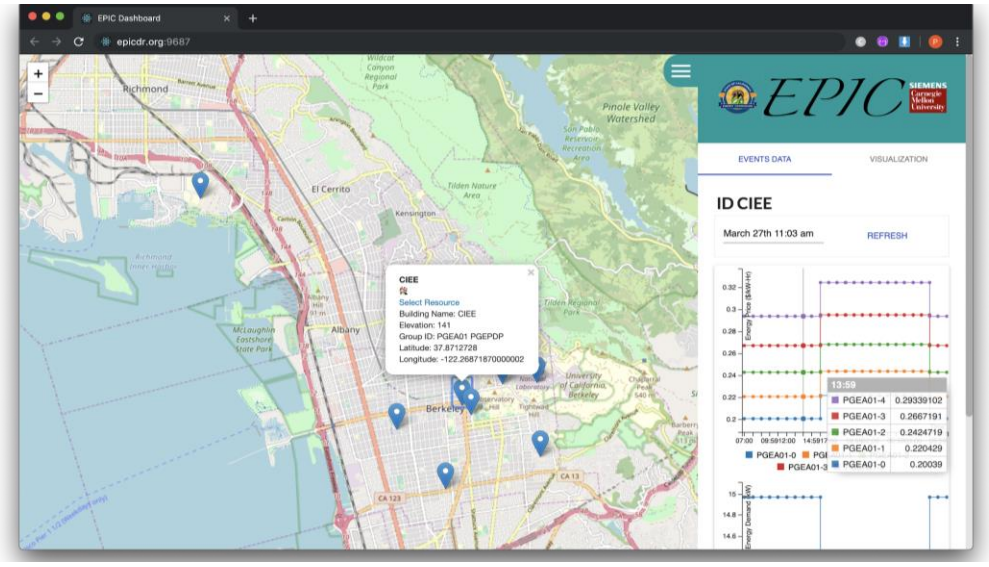
Develop Applications

The EPIC platform brings together interactions between the utility and buildings. As a platform, EPIC manages many services that can be deployed to different servers. It is important, for debugging and general monitoring, to know what is going on in the platform at any given time. At the same time, the EPIC platform provides a view of all of the buildings in the pilot study and how they are reacting to the pricing events. As such, having a map-based dashboard that supports all of the buildings and their interactions with the utility can be a useful tool.

Two applications were developed to aid in debugging the EPIC platform and visualizing both workflow and the pilot study. Both are integrated into the same dashboard:

- Messaging visualizer
- Pilot buildings on map / dashboard

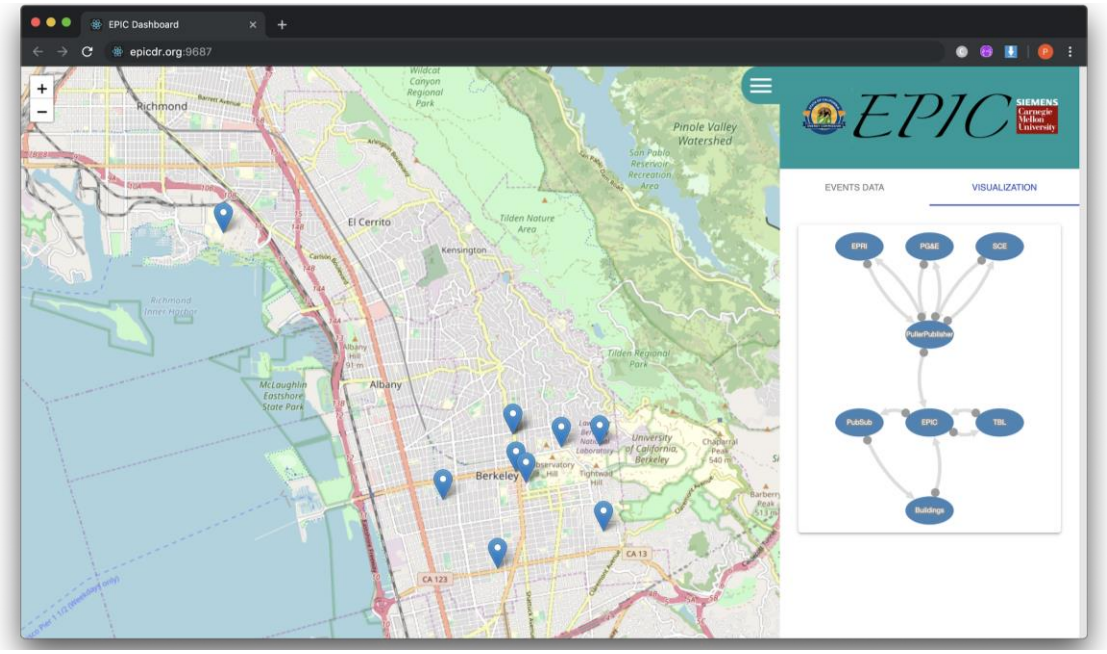
Figure 7: Demand Response Dashboard.



The Demand Response dashboard show the price and energy demand from multiple buildings in the DR program shown on the map on the left.

Credit: Jack Hodges and Steve Ray

Figure 8: Flow of Price Signal.



The diagram shows the flow of data.

Credit: Jack Hodges

Making the Software Tools Robust and Secure

The EPIC platform has a single endpoint to communicate with the utility companies and the XBOS servers. This endpoint is protected by the SSL encryption. In order to successfully establish a connection, the client will have to provide a valid signed certificate and the corresponding RSA key.

XBOS-DR Platform Development

This section covers the development within the **XBOS platform** to support a variety of networked devices, and applications, and the hardening to make the XBOS-DR platform robust and secure. More complete information lies in Appendix C.

The eXtensible Building Operating System (XBOS) platform is an open-source, secure, distributed operating system realized on top of a family of technologies developed by the Software Defined Buildings[1] (SDB) group, later renamed the Berkeley Energy and Transportation Systems[2] (BETS) research group, at UC Berkeley. XBOS-DR is an extension of the XBOS platform with a focus on integrating building management with demand response capability.

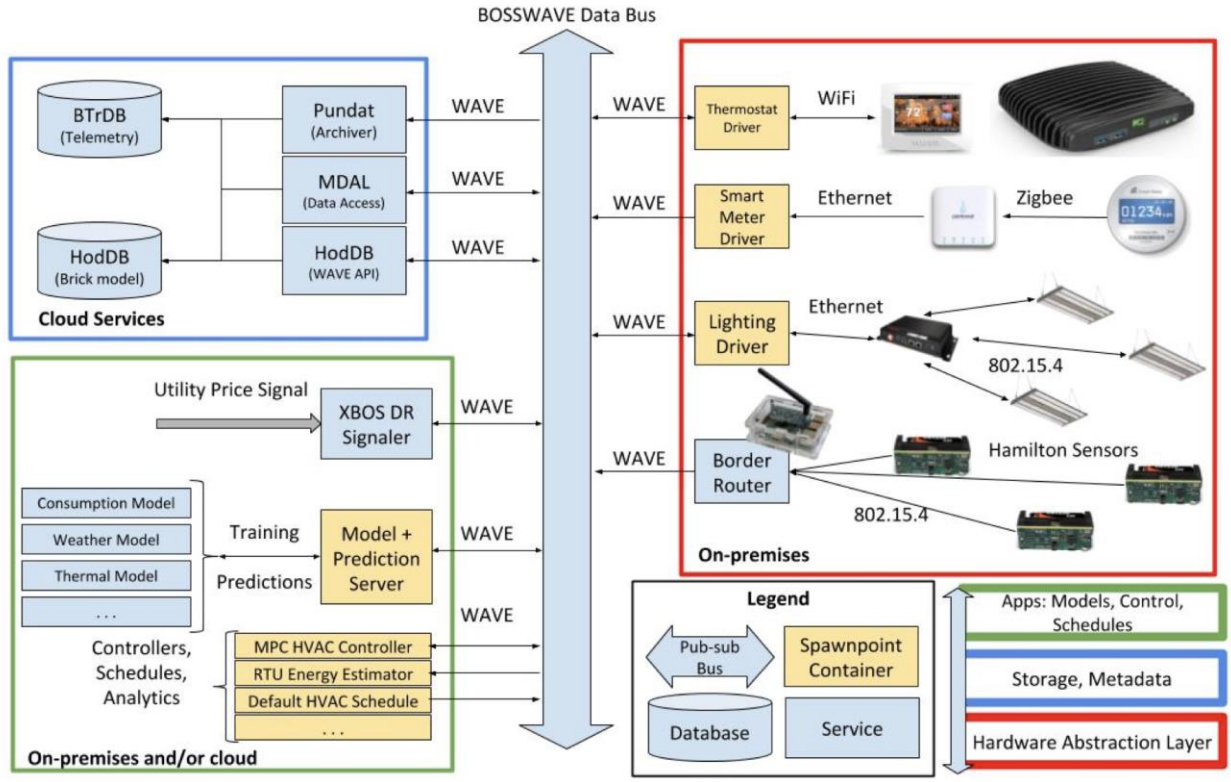
[1] http://sdb.cs.berkeley.edu/

[2] https://bets.cs.berkeley.edu/

Because XBOS was designed to be extended, these extensions can be implemented entirely within the XBOS architecture. As such, first is described the high-level architecture of XBOS before exploring each of its components, the interfaces between them, and finally how the required components of XBOS-DR can be implemented within this framework. Figure 5 provides a high level view of the overall XBOS system architecture that integrates building characteristics with individual control systems.

The figure below shows an overview of the software architecture: hardware abstraction, data management (storage/metadata), applications (price ingestion, models, controls, schedules). The architecture of the XBOS-DR platform: a secure data bus (Wave) is the underlying backbone of the platform. The upper left shows the databases for time series data as well as the metadata. The upper right shows the various hardware components, such as the thermostats, interval meter gateway, lighting controllers, and indoor environmental sensors. The lower left shows the transactive price signal, the model and prediction server, and various control schemes.

Figure 9: Schematic of XBOS Platform.



The diagram shows the components of XBOS.

Credit: Gabe Fierro.

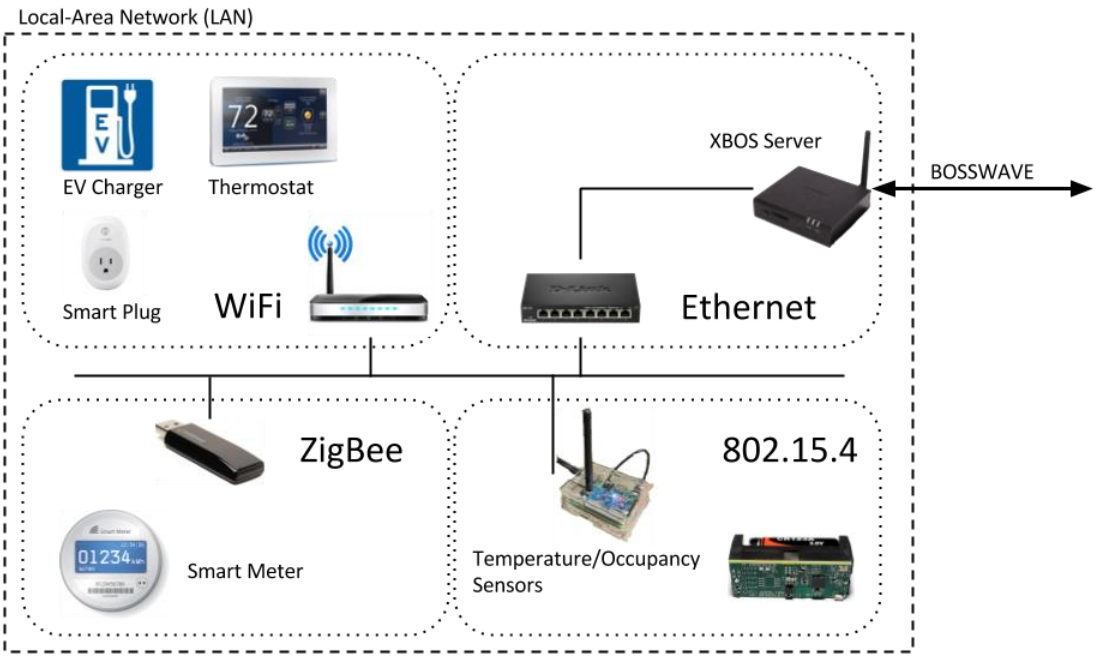
The XBOS architecture is extensible in that it allows for the integration of additional services, controllers, drivers and other components that may provide functionality beyond what is already contained in XBOS. For XBOS-DR, these additional components involve integration with

the utility grid, buildings, building automation systems, sensors, thermostats, and other devices such as electric vehicles. The BTrDB[3] database is a high-performance database developed by the BETS team. Also developed by the BETS team, the BOSSWAVE secure data bus was used in the project, but ultimately evolved to WAVEMQ[4] that will be used for future projects. HodDB[5] is the database for the Brick[6] metadata schema.

Communication

There are two considerations for communication in the XBOS-DR system: the physical medium (how to talk to a device or service) and the application details (what the device/service says and how to talk to it).

Figure 10: Schematic of XBOS Communication.



The diagram shows the communication components of XBOS.

Credit: Gabe Fierro.

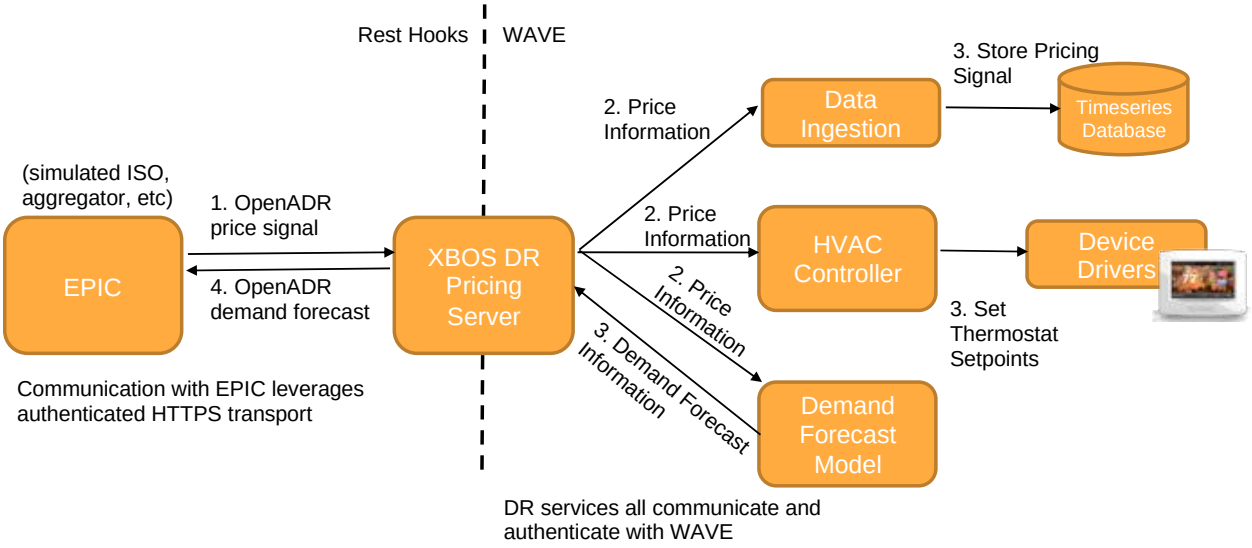
The EPIC platform and services generates three different pricing signals, which are then passed to the XBOS to control the building HVAC.

Figure 35 depicts the architecture of the XBOS pricing services and the flow of operation.

---

[3] http://btrdb.io/
[4] https://github.com/immesys/wavemq
[5] http://hoddb.org/
[6] https://brickschema.org/

Figure 11: XBOS Pricing Architecture



The diagram shows the architecture of the XBOS platform receiving price information.

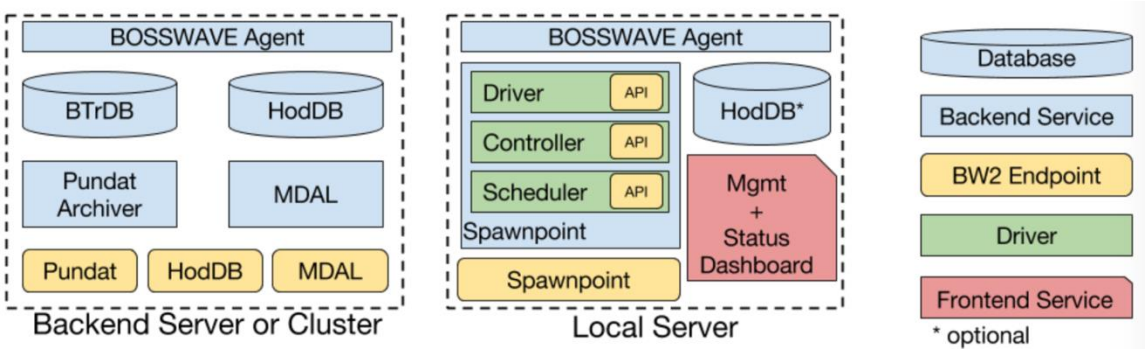
Credit: Moustafa AbdelBaky

The EPIC platform publishes a pricing signal for each tariff to the XBOS-DR Pricing Server using the openADR format. The XBOS-DR server extracts the pricing information from these tariffs and publishes this information using WAVE to the corresponding topics for each tariff. The data ingestion service, which is subscribed to all of the pricing topics, then stores the pricing information in the timeseries database for each of the tariffs. Similarly, the HVAC controller for each building, which is subscribed only to the tariff topic associated with that building, uses these prices to generate the schedule and control the HVAC systems. The schedule setpoints are then further relayed to device drivers in the building. The drivers are responsible for enforcing the scheduled setpoints. Finally, a demand forecast service, which models the total building power consumption, uses the pricing information to predict the overall power consumption of each building. The predicted consumption is then published to the XBOS-DR Pricing Server. The Pricing Server uses the demand forecast to generate an openADR compatible signal with the predicted forecast, which is then published to the EPIC platform.

XBOS follows a "microservice" architecture; that is, an instance of XBOS is composed of a set of distributed services connected by a secure bus (BOSSWAVE). As such, there are several ways to arrange these services. This guide (https://docs.xbos.io/install_overview.html), covers one common architectural configuration.

In this configuration, there is a local on-premises server and a set of remote services.

Figure 12: Schematic of Servers Needed to Host XBOS-DR.



The diagram shows the servers of XBOS.

Credit: Gabe Fierro.

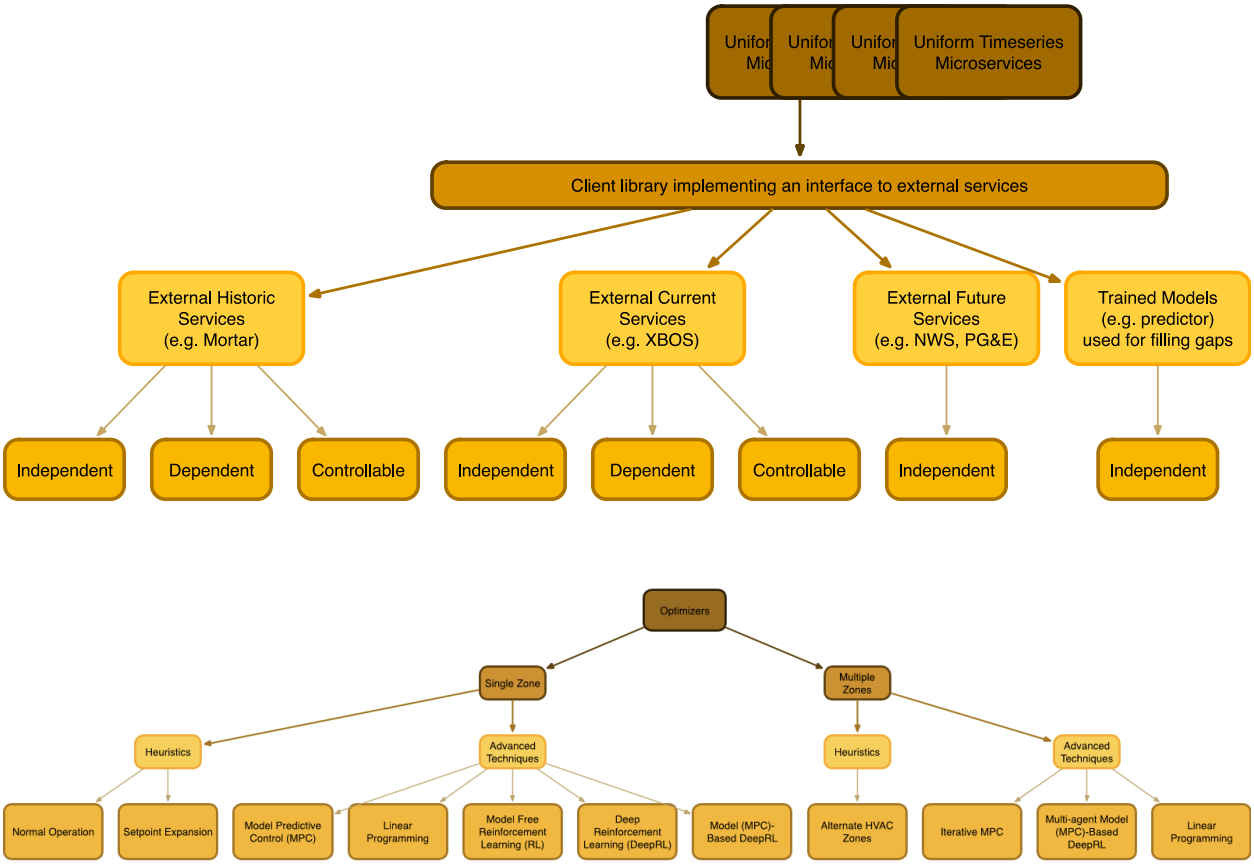
Microservices

A microservice is a small encapsulated service that provides a simple tool for anyone (especially non computer science experts) to develop applications on top of the XBOS platform. Each individual service provides a microfunctionality, such as establishing the price (given a building's utility territory and tariff), weather, HVAC, occupancy and so on. Each microservice can reason and perform independently; new services can combine multiple existing microservices. The modular nature allows improving or replacing each microservice independently. The microservices can be easily integrated with the user interface or other components.

For the XBOS project, several microservices were developed: to provide uniform access to timeseries data, provide timeseries forecast for models, provide an HVAC action based on different optimization techniques and objectives, and simulate HVAC control in buildings.

The following figures show examples of microservices. A quite typical microservice is one that fetches the desired data stream: the timeseries microservice.

Figure 13: Uniform Timeseries Microservices.



Top: time series microservices. Bottom: the diagram shows different optimizers identified by the project.

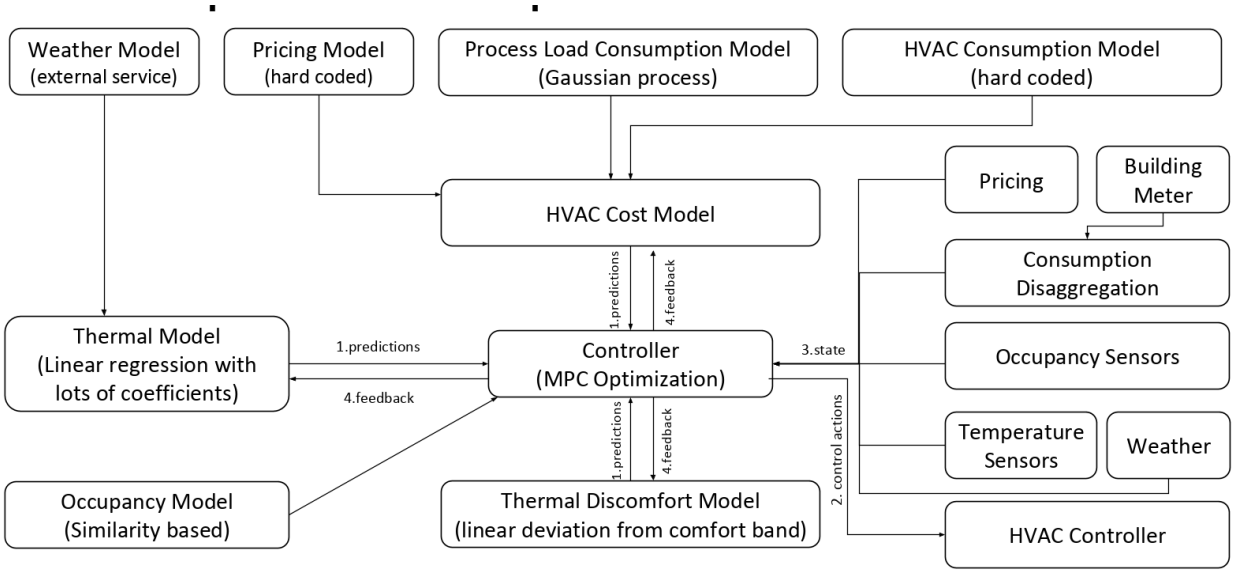
Credit: Moustafa AbdelBaky.

Microservices are further detailed in and available at https://pypi.org/project/xbos-services-getter/.

Control Code

The project looked at different control algorithms to reduce demand on event days: the simplest technique was to simply expand the temperature setpoints to reduce cooling runtime during the event. More complicated was Model Predictive Control (MPC) optimizer, shown in the figure below.

Figure 14: Diagram of Model Predictive Control Optimizer.



Top: time series microservices. Bottom: the diagram shows different optimizers identified by the project.

Credit: Moustafa AbdelBaky.

Pilot Testing

This section discusses the testing of the platform in several buildings to test the ability of the system to control the operation of the devices, to save energy, to reduce peak electrical load, and to engage customers. This is described more fully in Appendix D.

The research team developed a set of selection criteria, such as:

- Must be a commercial building less than 50,000 sf.
- Must have electrical service provided by an investor-owned utility (SCE, PG&E, or SDG&E)
- Must have individual meter for the building or space under consideration
- Not multi-tenant
- Must pay for electricity bill
- No additional planned retrofits or renovations between now and December 2018.
- Must commit to participating in the 12-18 month project and agree to interaction with the building control systems.
- Must agree to building energy audit.
- Has at least 12 months of interval meter data available.

Outreach was primarily conducted by QuEST through existing contacts and relationships with local government entities, regional energy efficiency bodies, and industry associations. The

recruitment process took place over several months, from April 2017 through Nov 2017. There were several stages, including a walk through and audit of the building.

Figure 15 shows images of the buildings and Table 1 describes the buildings, mostly in the PG&E territory, but two from SCE.

Figure 15: Images of the Project Buildings.

Photos of the buildings.

Credit: Greg Thomson and Google Earth

Table 1: Buildings Recruited for the Project.

Site Name	Area	Classification	Tariff	IOU	Climate Zone
CSU Dominguez Hills (SAC2)	15,548 SF	Business (Higher Ed., Offices/Classrooms)	Master Metered	SCE¥	8
Orinda Community Center	20,488 SF	Multi-use assembly spaces (Theater, Meeting rooms)	HA10SX	PG&E¥	12
North Berkeley Senior Center	20,834 SF	Senior center (banquet hall & kitchen)	HA10SX	PG&E	3
The Local Butcher Shop	2,850 SF	Mercantile (Commercial Mixed-Use)	HE19S	PG&E¥	3
Avenal: Animal Shelter	4,132 SF	Animal Shelter (with storage)	HA1X	PG&E¥	13
Avenal: Movie Theatre	15,820 SF	Assembly (Movie Theater, meeting rooms)	HE19S	PG&E¥	13
Avenal: Veterans Hall	8,683 SF	Senior center (banquet hall & kitchen)	HA1X	PG&E¥	13
Avenal: Recreation Center	2,417 SF	multi-use community center with IT training facility	HA1X	PG&E¥	13
Avenal: Public Works Department	12,700 SF	Moderate Hazard Storage	HA1X	PG&E¥	13
Fire station 1, Hayward	8,700 SF	Business (with storage, kitchen, and sleeping areas)	HA10SX	PG&E	3
Fire station 8, Hayward	6,500 SF	Business (with storage, kitchen, and sleeping areas)	A6	PG&E	3
Berkeley Corporation Yard	9,600 SF	Business (offices)	A10SX	PG&E	3
Richmond Field Station, Bdg 190	1,850 SF	Business (Higher Ed., Offices/Classrooms)	Master Metered	PG&E¥	3
South Berkeley Senior Center	10,427 SF	Senior center (banquet hall & kitchen)	HA1X	PG&E	3
Jesse Turner Fontana Community Center	43,193 SF	Assembly (Banquet Hall, Indoor Gymnasium)	Master metered	SCE	10
CIEE	8,424 SF	Business (offices)	A1X	PG&E¥	3

LBNL building 90C	18,500	Business (offices)	Master Metered	PG&E¥	3
Word of Faith Christian Center	19,733 SF	House of Worship and Accessory School Spaces	HA1X	PG&E¥	12
Orinda Library	24,250 SF	Library	HA10SX	PG&E	12

Each building was documented in a Revit 3-D model; the figure below showed an example of a developed floor plan showing the HVAC zones.

Figure 16: Example Floor Plan.

Fig. 2 - Avenal Movie Theatre HVAC Zone Plan

Floor plan generated in Revit.

Credit: Greg Thomson

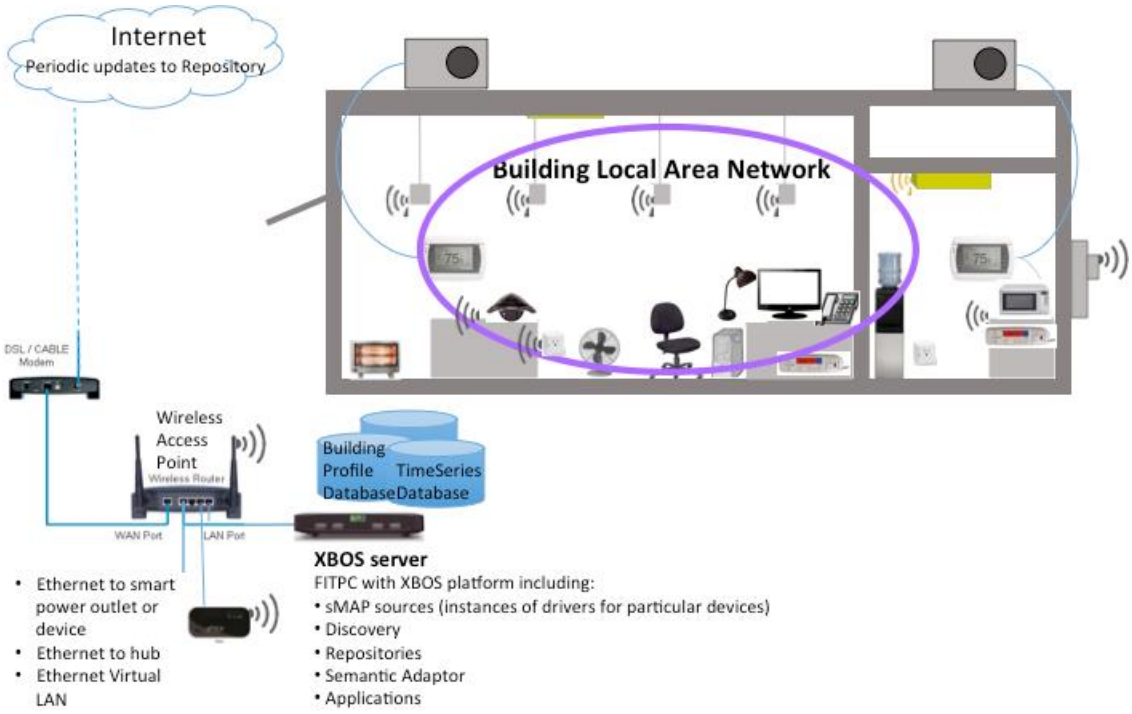
In order to determine the effect of energy efficiency and demand response event measures, the team worked to obtain historical electrical energy consumption from each participant.

Participating customers authorized QuEST to receive a daily interval data. The authorization was executed, electronically, through each customer's utility account management portal.

QuEST became a certified, third-party recipient of utility usage data and designed its system to communicate with utility Green Button or Share My Data API.

The XBOS-DR platform consisted of a number of components: miniature computer, Local Area Network, networked thermostats, and whole building energy meter gateway.

Figure 17: The XBOS-DR Platform Installed in a Small Commercial Building.



Schematic of XBOS in a small commercial building

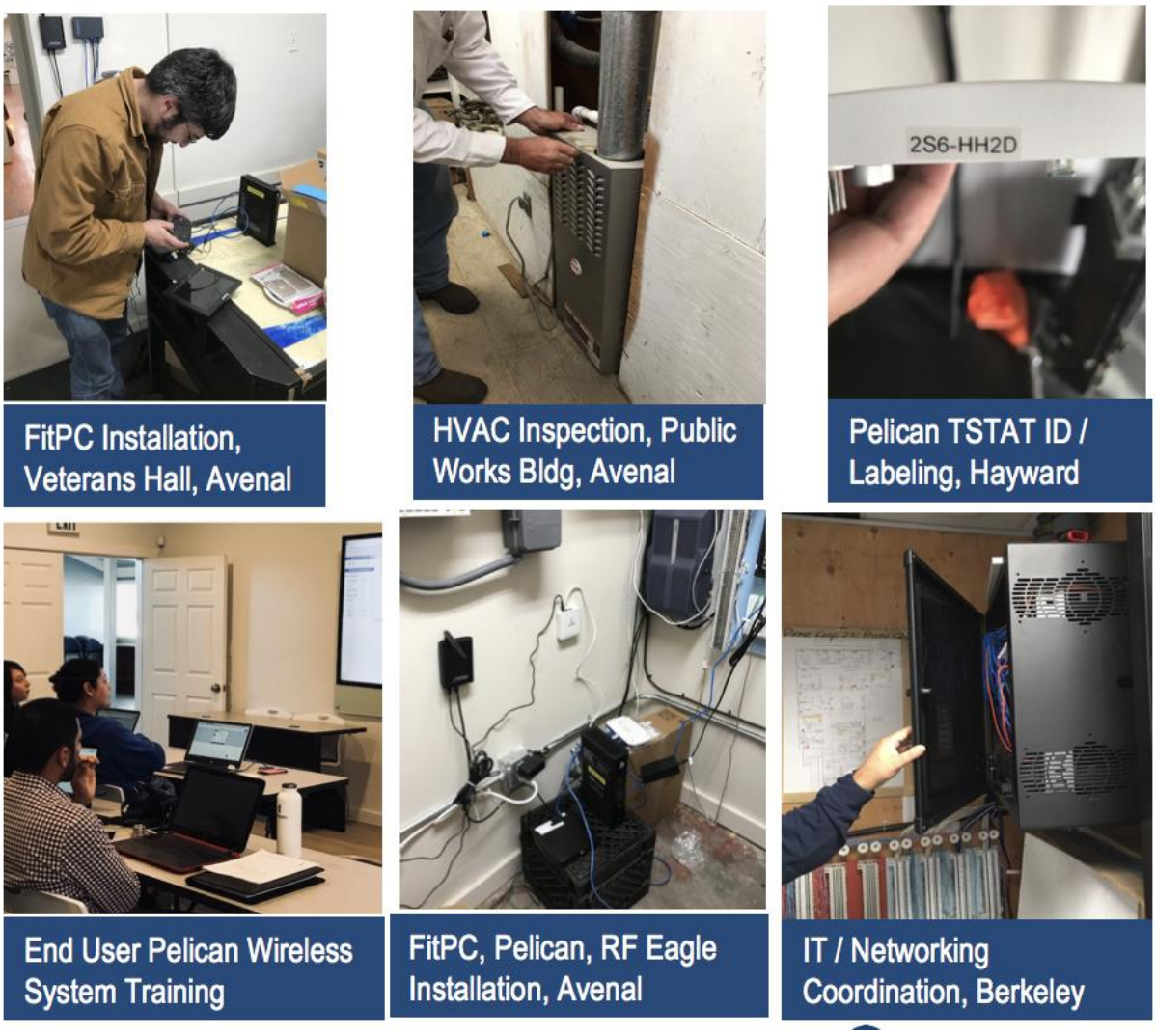
Credit: Therese Peffer

The XBOS-DR platform consists of:

- A miniature **computer** (Fit PC) either connected to the building's Local Area Network or creating its own LAN via multiple gateways. Requires connection to the **Internet**. Likely will need an **Ethernet port**.
- A gateway to the whole building interval **meter**: either with the Rainforest Eagle gateway or TED current transducers on the main circuit breaker panel to the building.
- Connection (e.g., via WiFi) or gateway to **connected thermostats** (whether through the Pelican gateway or via WiFi to Venstar)

- Connection to **lighting**, either through a gateway (e.g., Enlighted systems for controlling LEDs) or replacing the manual bipole switch with a connected switch (e.g., EnOcean device) (See Appendix E for installation procedure)
- Connection to sensors: one building used Hamilton **sensors**, with temperature, occupancy (PIR).

Figure 18: Photos of Installation at Different Sites.



Installation and training images from the sites.

Credit: Irina Krishpinovich

All data went to BTrDB and was accessible via a web-based plotting tool (Mr.Plotter) or through writing Python queries to access the data through HodDB. The research team discovered some problems with the data and worked to improve quality control.

In the commissioning phase, there were other issues uncovered. The city of Avenal was working to resolve poor Internet connection. The whole building energy metering device (TED) at the

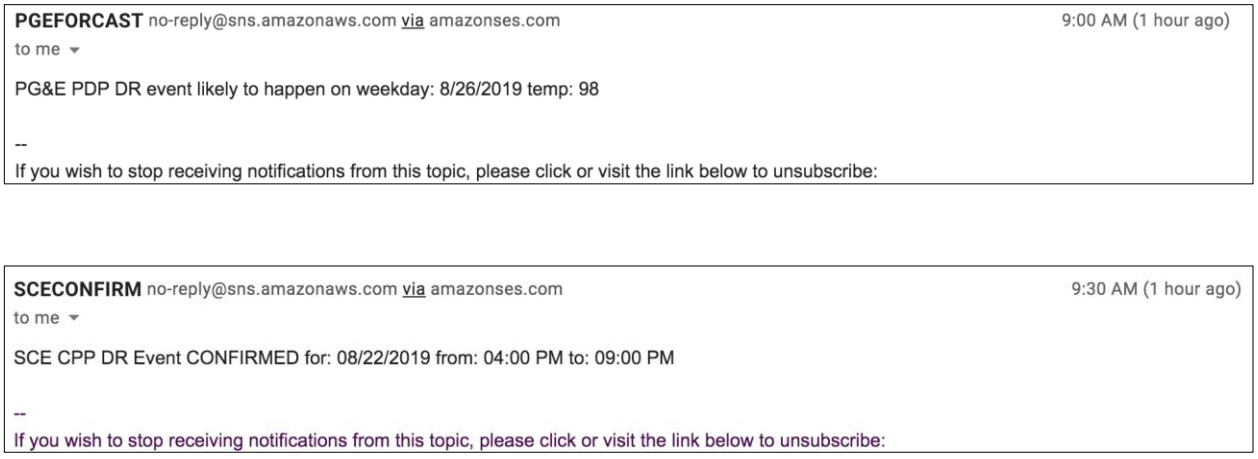
Jessie Turner Center did not work and had to be replaced. The electrician at the Richmond Field Station inadvertently wired the TED backwards and provided negative readings, and required rewiring. The plug to the miniature computer kept getting disconnected at the Local Butcher Shop.

Conduct Demand Response Tests

The research team conducted several events in multiple buildings in Summer 2018 and 2019. One test with multiple loads looked to reduce overall demand by balancing loads. The other tests occurred on utility demand response event days. The two main strategies were to expand setpoints, force the HVAC units to remain in first stage cooling, and Model Predictive Control (MPC).

The research team had difficulty obtaining the event signal directly from the utility, but managed a couple of work-arounds. One solution was to get one of the thermostats enrolled in the utility program (e.g. PG&E's Peak Day Pricing) and the other solution was to scrape the website and generate both a forecast of potential events and confirmed events. The forecast information was quite valuable.

Figure 19: Service that Provided Forecast and Confirmed Events by Email.



Top: email from an Amazon Web Service announcing the likelihood of an event in the PG&E territory; bottom: email from the AWS confirming an event in the SCE territory.

Credit: Moustafa AbdelBaky

Conduct Energy Efficiency Interventions

The main energy efficiency stemmed from installing networked thermostats, the Pelican thermostat system. Energy savings mainly stemmed from training the building managers to use the programmed schedule. The research team also deployed a Model Predictive Control scheme in the test building, CIEE.

CHAPTER 3:
Project Results

This section discusses project results, including the data quality issues, evaluation of Energy Efficiency (EE), load balancing, and evaluation of Demand Response (DR) results obtained during the field test. EE savings are estimated using industry standard practice for measurement and verification (ASHRAE Guideline 14), while DR savings are estimated using standard baselines used in DR programs in California (3 of 10 baseline) as well as other options to obtain more realistic results.

Data Quality Issues and Management

The primary data quality issue were periods of missing data. There was a large amount of missing data for varying durations across the various data streams. To address this, first the researchers identified the periods of missing data for the outdoor air temperature data stream, thermostat data stream and Green Button meter data stream. The researchers also flagged data with questionable quality, including outlier values below lower bounds and above upper bounds. Then the researchers evaluated the data collection process to determine if the error was in data generation or in the transmission of data from the sources to the BTrDB database. The recoverable data was retrieved, and the process was updated to avoid further errors.

Energy Efficiency Analysis

Developing a Baseline

The EE baselines are constructed using interval meter data from the local utility where possible, otherwise using sub-meters. Two buildings (Jessie Turner and Richmond Field Station) used The Energy Detective (TED) current transducers that measure current flow at the building circuit breaker panel; one building had an E-mon D-mon submeter (CSU-DH). There is at least one year of baseline data for each site; most of the buildings have two years of baseline data, that is, before the connected thermostats were installed.

Effect of Installing Networked Thermostats

Table 2 shows 12 buildings for which the researchers have thermostat and whole building energy data. The table shows the impacts of the change in energy consumption resulting from the installation of Pelican networked thermostats. Most buildings show modest savings 7-9%; one building achieved 18% savings. A few sites show minor negative savings; upon exploration, one site added a 1000 Watt baseload during the project. Note that over the project the North Berkeley Senior Center closed, which greatly increased the energy use of the South Berkeley Senior Center.

Table 2: Energy Impact from Installing Networked Thermostats.

Site	#Days since Pelican Installation	Energy Savings (%)	Dollar Savings (%)	Best Model	Adjusted R2	RMSE	MAPE
avenal-veterans-hall	597	18.23	21.04	RFR	0.96	0.75	12.01
hayward-station-8	585	-2.36	-3.52	RFR	0.97	0.46	112.71
hayward-station-1	585	7.65	7.14	RFR	0.95	0.98	4.28
avenal-animal-shelter	613	8.07	7.78	RFR	0.93	0.45	6.18
avenal-recreation-center	613	2.03	3.51	RFR	0.92	0.51	26.74
avenal-movie-theatre	620	9.50	13.39	RFR	0.94	0.77	10.26
berkeley-corporate-yard	654	8.02	6.54	RFR	0.93	3.52	8.63
ciee	780	7.91	4.79	RFR	0.92	0.28	6.75
avenal-public-works-yard	600	2.24	0.52	RFR	0.86	0.42	11.41
orinda-community-center	632	-2.80	-2.44	RFR	0.94	1.33	10.45
north-berkeley-senior-center	702	-2.87	-5.74	RFR	0.96	1.41	8.34
south-berkeley-senior-center	590	-29.15	-26.29	RFR	0.92	1.03	10.00

*** RFR - Random Forest Regressor, RMSE - Root Mean Squared Error, MAPE - Mean Absolute Percentage Error.**

The table shows the energy reduced (or increased) after the installation of networked thermostats, and shows the model used to develop the data compared to the baseline.

Credit: Pranav Gupta

A number of models were used to construct baselines for each site: Linear Regression, Lasso Regression, Ridge Regression, Elastic Net Regression & Random Forest Regressor. The adjusted R-squared, a metric that measures the proportion of variation in the dependent variable (which is the real energy usage of site) that is explained by the independent variable (which is the model predicted energy usage of site), was used to identify the best performing model for each site. As is evident in the table above, Random Forest Regressor performed better than all the other models for all of the sites, and therefore it was used to calculate the energy and cost savings for each site. Other metrics such as the Root Mean Squared Error (the standard deviation of model prediction errors) and the Mean Absolute Percentage Error (measure of prediction accuracy of a model) underline the accuracy of the models and accordingly, the correctness of the energy and cost savings of each site.

Zone Level Analysis

The research team developed code to generate an analysis per zone for most of the buildings. The intention was to determine:

- Data quality: percent of missing data, and which data stream has the highest percentage of missing data
- Size of the zone
- Annual heating and cooling hours
- Heating and cooling runtimes outside of typical business hours
- Heating and cooling runtimes during DR events
- Percent time that the zone does not reach heating or cooling temperature
- Percent hours HVAC running outside business hours
- Estimated cost per year

The tables below show examples from two buildings. This information proved to be quite valuable as the research team looked for reasons for excessive energy consumption, wasted energy (e.g., heating or cooling when the building is closed), and designed demand response

strategies. Table 3 shows a building in Avenal, California; note the fifth column from the right shows the percentage of HVAC energy (estimated from face-plate values on the equipment) compared to overall building energy (from either the interval meter or GreenButton data). Most of the buildings showed similar pattern: the majority of the load comes from cooling. From Table 3, the team discovered a storage zone that was inadvertently left in heating mode for weeks at a time, wasting energy. Table 4 shows the Public Works Yard in Avenal, which had a single HVAC zone; the other loads (forklift chargers and industrial equipment) dwarf the cooling load from this zone.

Table 3: Zone Energy Analysis of the Veterans Hall in Avenal, CA

Unnamed: 0	Data quality (% missing)	Location of Missing Data	Zone Area (sf)	Runtime-heating-annual (hours)	Runtime-cooling-annual (hours)	Runtime-heating-annual-Non-Business-period (hours)	Runtime-cooling-annual-Non-Business-period (hours)	Runtime-cooling-DR (event only hours)	Cooling Energy (runtime x faceplate)--annual (kWh)	Cooling Energy (runtime x faceplate)--DR (kWh)	% time did not reach target cool annual	% time did not reach target heat annual	% time did not reach target cool July	% time did not reach target heat July	Site Annual Energy (kWh)	% of cooling energy to total energy	Zone SF % of GSF	peak demand (kW)	GHG Emissions (lbs-CO2/Zone)	cost annual cooling per zone	Typ. Cooling Setpoint During Occupied Period
hvac_zone_ac_3	12.82%	hvac_zone_ac_1 State	962	1767	1752	1022	894	32	9636	176	20.24%	16.14%	33.06%	0.00%	33437.8	29%	13.2%	7	5049.26	2322.48	75.0859
hvac_zone_ac_1	12.82%	hvac_zone_ac_1 State	1989	1702	1474	988	729	28	8107	154	17.76%	18.00%	33.60%	0.00%	33437.8	24%	27.2%	7	4248.07	1971.75	75.4191
hvac_zone_ac_4	12.82%	hvac_zone_ac_1 State	2048	212	1872	94	933	29	10296	159.5	22.39%	1.95%	33.60%	0.40%	33437.8	31%	28.0%	7	5395.1	2446.18	75.3079
hvac_zone_ac_6	12.82%	hvac_zone_ac_1 State	672	841	941	523	503	1	5175.5	5.5	11.62%	3.45%	14.11%	0.27%	33437.8	15%	9.2%	7	2711.96	1203.01	76.9019
hvac_zone_ac_5	12.82%	hvac_zone_ac_1 State	862	2565	985	1565	526	15	2167	33	13.27%	14.67%	25.40%	0.00%	33437.8	6%	11.8%	7	1135.51	527.736	76.5154
hvac_zone_ac_2	12.82%	hvac_zone_ac_1 State	775	832	1251	482	601	32	6880.5	176	15.15%	3.55%	36.83%	0.00%	33437.8	21%	10.6%	7	3605.38	1712.64	75.1305

The table shows zone level data for Veterans Hall. The orange highlights the highest or lowest value in the column.

Credit: Greg Thomson, Moustafa AbdelBaky, Pranav Gupta

Table 4: Zone Energy Analysis of the Public Works Yard in Avenal, CA

Unnamed: 0	Data quality (% missing)	Location of Missing Data	Zone Area (sf)	Runtime-heating-annual (hours)	Runtime-cooling-annual (hours)	Runtime-heating-annual-Non-Business-period (hours)	Runtime-cooling-annual-Non-Business-period (hours)	Runtime-cooling-DR (event only hours)	Cooling Energy (runtime x faceplate)--annual (kWh)	Cooling Energy (runtime x faceplate)--DR (kWh)	% time did not reach target cool annual	% time did not reach target heat annual	% time did not reach target cool July	% time did not reach target heat July	Site Annual Energy (kWh)	% of cooling energy to total energy	Zone SF % of GSF	peak demand (kW)	GHG Emissions (lbs-CO2/Zone)	cost annual cooling per zone	Typ. Cooling Setpoint During Occupied Period
0 hvac_zone_public_works	2.34%	hvac_zone_public_works State	551	1233	1400	749	410	31	2745.6	68.2	15.21%	3.31%	36.96%	0.00%	21542.9	13%	100.0%	2	1438.69	696.696	77.0202

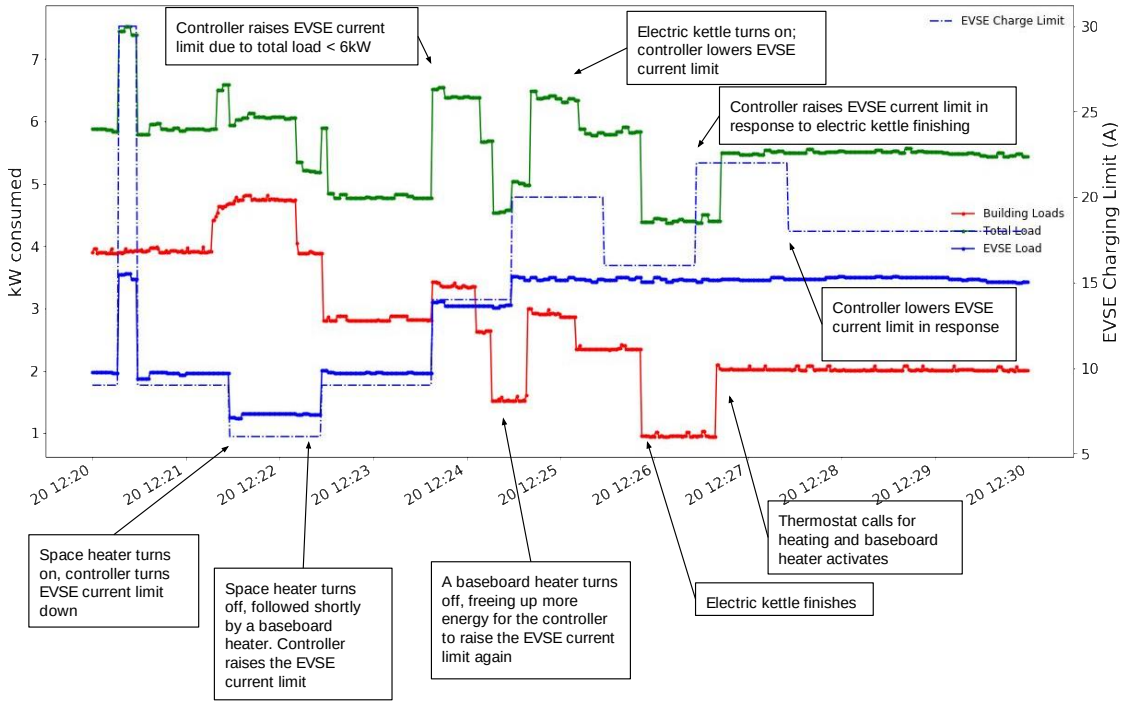
The table shows zone level data for the Public Works Yard in Avenal. Note the percentage cooling load to total building load is 13%.

Credit: Greg Thomson, Moustafa AbdelBaky, Pranav Gupta

Load Balancing Testing

One of the test sites (Richmond Field Station) had a Level 2 (240V) EV charger and was used for the XBOS-V project. The research team installed the XBOS platform and tested the coordination among the EV charger, three baseboard heaters, and five plug loads (e.g., a refrigerator, a microwave, a hotwater kettle, a space heater, and a fan). The inputs to the controller are 1) the maximum power threshold, 2) the priority of loads (which load has higher priority), and 3) an optional minimum charging rate for the EV. The controller continuously obtains the 1) overall power consumption of the building, 2) the current state and power consumption of the EV, and 3) the current state and power consumption of all controllable loads. If the total consumption is above the predefined threshold, the controller selects the loads that should remain on (based on the predefined priorities), turns off other loads, and adjusts the charging rate for the EVSE (if needed).

Figure 20: Coordinated Load Control with an EV Charger



The graphic shows a ten-minute test in modifying the EV charger in response to other loads.

Credit: Gabe Fierro and Moustafa AbdelBaky

DR Event Testing

The team ran 3-10 DR events in 2-13 buildings: Peak Day Pricing (PDP) events from 2-6pm in PG&E territory and Critical Peak Pricing (CPP) events from 2-6pm (2018) and 4-9pm (2019) in

SCE territory. Table 5 below shows the various strategies used for the buildings over the two seasons.

Table 5: Demand Response Event Dates

	PG&E PDP: 2018_06_12	SCE CPP: 7/6/2018	SCE CPP and PG&E PDP: 7/10/2018	SCE CPP 8/6/2018	SCE CPP 9/28	SCE CPP 10/18	SCE CPP 4-9; 7/12/2019	SCE CPP-4-9; 7/15/2019	SCE CPP-4-9; 7/16/2019	PG&E PDP 8/15/2019
Site Name										
Orinda Community Center	Temp expansion		MPC							
Avenal: Veterans Hall	MPC		MPC							
Avenal: Public Works Department	Temp expansion		MPC							
Avenal: Animal Shelter	Temp expansion		Temp expansion							
Avenal: Movie Theatre	Temp expansion		Temp expansion							Temp expansion
Avenal: Recreation Center	MPC		MPC							
South Berkeley Senior Center	Temp expansion		Temp expansion							
North Berkeley Senior Center	Temp expansion		MPC							
Berkeley Corporation Yard			MPC							
Word of Faith Christian Center (meter 1)	Temp expansion		MPC							
Word of Faith Christian Center (meter 2)	Temp expansion		MPC							
CIEE offices, 2nd floor	MPC		MPC							
CSU Dominguez Hills (SAC2)		Temp expansion		cycle rate change	precooling & temp expansion		temp expansion	temp expansion	temp expansion	
Jesse Turner Fontana Community Center		Temp expansion	Temp expansion	cycle rate change & temp expansion		precooling & temp expansion		temp expansion	temp expansion	

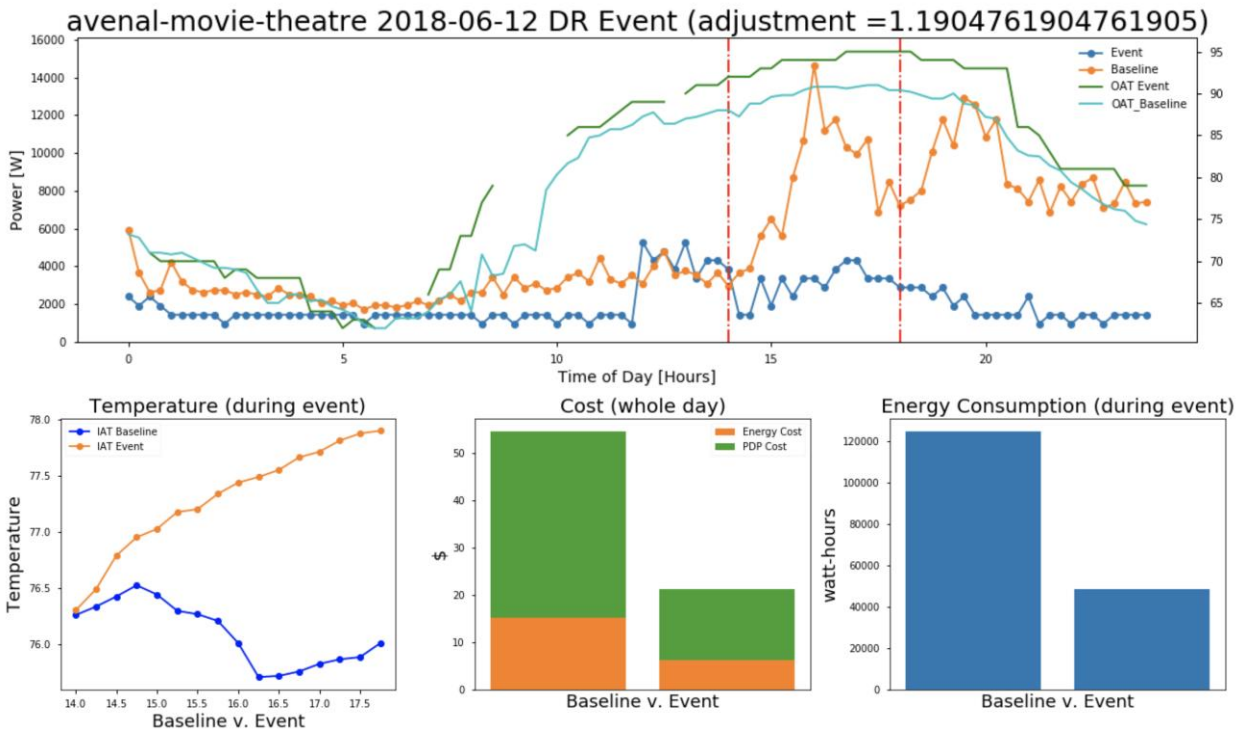
The table shows a summary of DR events in 13 buildings over two years in two territories using either temperature expansion (raise the cooling setpoint 4-5F), Model Predictive Control, changing the cycling rate of the RoofTop Unit, and precooling.

Credit: Therese Peffer

One of the first issues was to develop an appropriate baseline to compare peak demand and energy consumption during the non-event periods with event periods. The team discovered that the suggested methodology used by utilities (e.g., comparing to the past 10 days) was consistently not a good match since it typically underestimates the savings due to lower outdoor air temperature. As a result of the wide range of power consumption profiles and site characteristics, the research team found that different baseline constructions performed with different levels of accuracy at each site and a uniformly best baseline across all sites could not be determined. To address the variability in baseline performance, the research team developed a baseline formulation process. Each site was tested on eight different baselines, including traditional power baselines with various hyperparameters, weather mapping baselines and a ridge regression baseline. These baselines were evaluated across a set of test days with high temperatures and no actuation. The coefficient of variation of the root mean squared error metric (CVRMSE) was calculated for each test day and aggregated to determine the best baseline per site. Each site was assigned a best baseline attribute and the baseline was used for evaluation of DR days at the identified site. The framework enables evaluation of the baseline accuracy at user-chosen intervals to incorporate recently generated data in the baseline models.

Each event was analyzed for each building. Figure 21 shows an example of data generated for a June 12, 2018 PDP event from 2 to 6pm at the Avenal Movie Theater. The top plot shows Outside Air Temperature (OAT) for the Event day and a generated Baseline, and the corresponding electrical power consumed during the Event day versus the generated Baseline. The graphic on the bottom show the indoor temperature differential between the hours of the Event compared to the generated Baseline (left), the total energy consumption during the Event and the generated Baseline (right) and the cost comparison for the whole day, identifying the cost associated with the standard tariff versus the cost associated with the peak pricing tariff for the Event and the Baseline (center).

Figure 21: Example of Graphic Generated for Event.

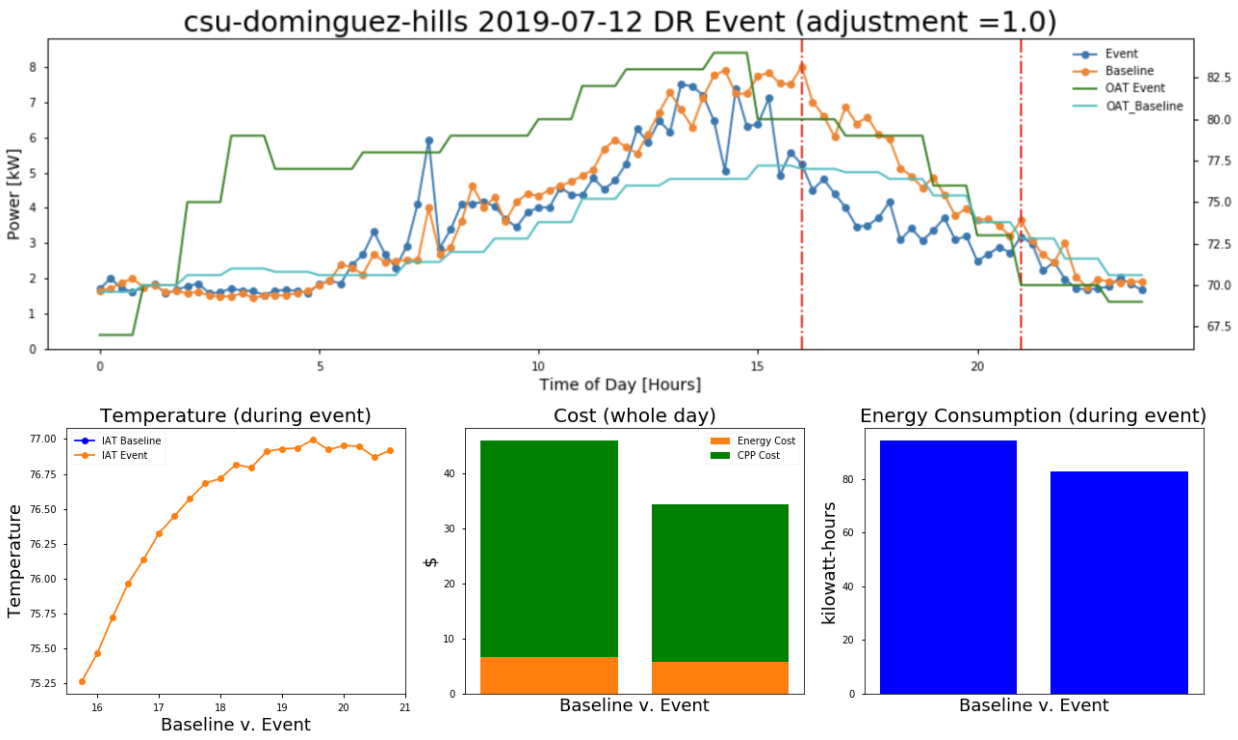


Data generated for a June 12, 2018 PDP event from 2 to 6pm at the Avenal Movie Theater.

Credit: Callie Clark

Figure 22 shows another example from a different building.

Figure 22: Another example of Graphic Generated for Event.



Data generated for a July 12, 2019 CPP event from 4 to 9pm at CSUDH.

Credit: Callie Clark

The following table summarizes the average savings in dollars and in energy across all event days.

Table 6: Summary Table of Demand Response Savings

	Average Savings[$]	Average Savings [kWh]
berkeley-corporate-yard	31.68	122.01
hayward-station-1	29.27	73.19
north-berkeley-senior-center	17.48	37.73
orinda-community-center	9.26	26.57
avenal-movie-theatre	7.75	39.51
csu-dominguez-hills	7.49	12.03
avenal-public-works-yard	7.39	21.49
local-butcher-shop	5.14	12.40
avenal-recreation-center	3.44	16.56
hayward-station-8	-0.84	2.57
ciee	-2.13	1.72
orinda-public-library	-3.91	-28.75
avenal-animal-shelter	-4.03	-10.77
south-berkeley-senior-center	-5.19	-7.38
avenal-veterans-hall	-7.02	25.81
word-of-faith-cc	-7.37	-0.83

Average Daily Savings across all event days where actuation occurred

Credit: Callie Clark

User Interface Testing

This section briefly describes the development and testing of the user interface: summarizing the needs and preferences of various commercial customers, garnered through surveys, interviews, and other means, describing the use cases, and outlining the functions and design of the user interface necessary to provide customer value in managing energy demand, and the user interface testing.

A section in Chapter 2 summarizes the user interface design, and Appendix F describes the testing protocol and results in more detail. The research team developed a user interface that provided:

- a means of receiving demand response or price signals from the utility, including notification of future events
- a means of prioritizing and managing demand response strategies (e.g., increasing thermostat setpoint on hot days to reduce air conditioning) according to the specific needs and loads of the customer. This can include HVAC, lighting, plug loads, EVs.
- feedback to the customer of the effectiveness of that demand response strategy (e.g., did the strategy save money? Did it negatively impact the business (e.g., productivity, sales))

- single place to manage (group and schedule) multiple thermostats for open business periods, closed times, vacations/holidays, including the potential of remote control
- visualization of temperatures in the space by zone
- visualization of whole building energy data
- usability
- reports including fault detection and diagnostics (e.g., are systems using energy during closed periods?)

This task took much longer than expected, partly due to student matriculation and partly due to its complex nature, as it involved front end and back end development to create the live interface. The team developed a testing protocol (Appendix F), and walked three subjects through more than 7 tasks that reflect the goals in the above bullet points.

The results of the user interface indicated that the users were able to successfully accomplish some tasks fairly easily, but suggested the need for changes to terminology and some basic functionality. The user interface testing showed users were able to accomplish: receiving and understanding the demand response signal, managing DR events, managing multiple thermostats, visualizing temperatures in the space by zone, and visualizing whole building energy. Scheduling and reporting could be made more usable through simple changes to the interface. The research team hopes to continue to use and develop the user interface.

Lessons Learned

The research team learned many lessons over the project.

Reflection on the Pelican Thermostats

The research team used thermostats sold by Pelican Wireless Systems in nearly all of the sites. Pelican allows user-developed software to interact with thermostats via a cloud-based web API. The team built a thermostat driver on top of this API, which allowed the team to query thermostat state, to retrieve historical data, and to actuate thermostats. The research team was able to cleanly integrate Pelican's thermostats into the larger XBOS system because of this API, but the team also encountered several difficulties that motivate improvements for future thermostats.

First, because any interaction with a thermostat must go through Pelican's cloud service, a thermostat must be connected to the Internet to receive commands or to service requests for data, even when the client that is issuing the command or making the request is on the same local network. A better design would allow client software, such as the driver, to interact directly with the thermostat over the local network, meaning many routine operations would continue to work correctly even if a site loses its connection to the Internet.

Second, Pelican's API allowed the team to schedule future changes to thermostat setpoints but offered no means of querying a thermostat for all currently scheduled changes. This asymmetry forced the team to engineer a workaround in which the researchers manually scraped a thermostat's schedule from Pelican's web browser interface.

Finally, the reseearchers encountered particular difficulties with Pelican's treatment of multi-stage cooling. A Pelican thermostat makes no distinction between the number of stages it is currently allowed to use versus the number of stages supported by the underlying HVAC system. During DR events, the researchers manually reconfigured thermostats as if the HVAC machinery did not offer second-stage cooling in order to prevent them from calling for second-stage cooling. At the end of the event, the team manually reverted this change. A future, more "DR-friendly" thermostat might explicitly distinguish between the number of permitted cooling stages and the number of supported cooling stages or perhaps support a temporary DR-only configuration.

Other lessons included the following:

Due to a miscommunication, the Fire Stations determined that they could not participate in events since it might affect comfort levels for the firefighters.

Jessie Turner Center is a designated cooling center and except for the basketball courts, could not participate in events that changed the temperature.

The researchers found that the typical recommended baseline methodology was flawed, especially for the first hot day of the season.

Orinda, Berkeley and Avenal had very interested facilities managers who were interested in energy savings and could provide multiple buildings for the pilot test.

Intermittent Internet led to problems for most of the buildings in Avenal.

Both the Berkeley and Avenal Corporation Yards (Table 4) had very large loads due to equipment (forklifts, machinery and so on); the HVAC load was tiny by comparison, so the DR events just couldn't provide much relief.

CHAPTER 4: Technology/Knowledge/Market Transfer Activities

This chapter documents technology, knowledge or other market transfer activities to the public from this project; these include

- Papers Submitted or Published
- Patents Submitted or Awarded
- Open-Source Software
- Students Hired
- Presentations
- Other Media Presentations
- Other research projects

The project team has had interest from the CBRE (real estate), Schatz Energy Research Center, at Humboldt State University, and Blue Lake Rancheria for XBOS, and Johnson Controls and ASHRAE BACnet committee for Brick.

Papers Submitted or Published

Fierro, Gabe, David E. Culler. HodDB: a Query Processor for Brick. In Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environment (BuildSys 2017). DOI: https://doi.org/10.1145/3137133.3141449

Fierro, Gabe, and David E. Culler. Design and Analysis of a Query Processor for Brick. ACM Transactions on Sensor Networks 14, 3-4, Article 18 (November 2018), 25 pages. DOI: https://doi.org/10.1145/3199666

Fierro, Gabe, Marco Pritoni, Moustafa AbdelBaky, Paul Raftery, Therese Peffer, Greg Thomson, and David E. Culler. Mortar: An Open Testbed for Portable Building Analytics. In Proceedings of the 5th Conference on Systems for Built Environments (BuildSys 2018). DOI: https://doi.org/10.1145/3276774.3276796

Panagopoulos, Athanasios Aris, Michail Katsigiannis, Marco Pritoni, Gabe Fierro, Daniel Lengyel, Therese E Peffer, Georgios Chalkiadakis, David Culler. 2018. Dealing with Expected Thermal Discomfort. *Proceedings of the ACEEE Summer Study on Energy Efficiency in Buildings*. 1:1-12. American Council for an Energy Efficient Economy: Washington DC.

Hodges, J., Garcia, K., Ray, S. Semantic Development and Integration of Standards for Adoption and Interoperability, Computer, Volume 50, Number 11, Pages 26-36, 2017.

Koh, J., Ray, S., Hodges, J. Information Mediator for Demand Response in Electrical Grids and Buildings. 2017 IEEE 11th International Conference on Semantic Computing (ICSC), San Diego, CA, pp. 73-76. 2017.

Hodges, J., Yu, D., Mayer, S., Hodges, J., Yu, D., Kritzler, M., Michahelles, F. An Open Semantic Framework for the Industrial Internet of Things. IEEE Intelligent Systems. Volume 32. Number 1. Pages 96-101. 201

Open-Source Software

Brick: https://brickschema.org/

Mortar Website: https://mortardata.org/

Tutorial Page https://tutorial.mortardata.org/

Brick Explore Tool https://querybuilder.mortardata.org/

XBOS-DR (BOSSWAVE version): https://github.com/SoftwareDefinedBuildings/XBOS

https://github.com/immesys/wave

Patents Submitted or Awarded

Four invention disclosures have been submitted before or during the course of this project:

- 2018E16692 US, A methodology and approach to assisting in ontology integration for improved domain representation and interoperability
- 2017E02411 AT, End to end semantic logic development in smart grid applications for non ontologists
- 2016E22295 US, A semantic negotiation mediator for smart grid demand response
- 2016E10272 US, A general semantic building block to support multi-domain semantic applications

Students and Post-Doctoral Researchers Hired

Six undergraduate students, 14 graduate students, and two post docs conducted research as part of this project

Presentations

Software Defined Buildings industry retreat, February 6, 2017

Siemens corporate leadership, Sept 27, 2017

Plenary talk at the Consortium for Energy Efficiency (CEE)'s Winter Meeting, San Francisco, January 18, 2018.

Electric Power Research Institute (EPRI) annual workshop on Advanced Energy Communities, July 10, 2018.

Workgroup 7, Mexico Smart Grid Workshop, Cuernavaca, Mexico, September 19, 2018.

XBOS-Microservices: SDB internal meeting, July 19, 2019

Writing Portable Building Analytics with the Brick Metadata Schema. Presented at ACM E-Energy 2019.

Demand Response Research Symposium for EPIC GFO-15-311, July 22, 2019.

Included as a research project in presentations made to the Energy Foundation, research colleagues from Jiangsu, Leuven, Denmark, and Mexico.

Other Media Presentations

George, Alexandra, Let's talk: information interoperability for the smart electrical grid,

Carnegie Mellon University Silicon Valley News, March 21, 2019

EPRI Webinar: Solutions for Customer to Manage their Energy Demand, April 25, 2018

https://groups.google.com/forum/#!forum/brickschema.

Additional Research Projects

The XBOS platform was the basis for a successful TRC-led proposal to NYSERDA, entitled OpenBOS-NY. The platform will be used in a small commercial building in New York.

The XBOS platform is used for the current DOE-funded ENERGISE project led by the California Institute for Energy and Environment (CIEE at UC Berkeley) to control loads with microsynchrophasor signals.

The XBOS platform is used in the current CEC-funded Solar+ project led by Humboldt State University (subcontract to Lawrence Berkeley National Lab) to control loads in a microgrid at a convenience store on a reservation.

The XBOS platform is used in the current DOE-funded Hamilton project to link the Hamilton environmental sensors with the Building Automation System.

A component of XBOS, Brick, was further developed during this project and became the focus of a separate three year DOE-funded project to begin October 1, 2019.

CHAPTER 5: Conclusions/Recommendations

The goal of the project was to improve commercial customer participation in providing electric grid resilience by enabling cost-effective management and integration of demand response with other building services in small commercial buildings. To achieve this goal the researchers built and pilot tested an energy management system on the eXtensible Building Operating System (XBOS) software platform. The system connected networked thermostats and electrical interval meters with a price signal, and developed a user interface that allowed notification, monitoring and control. The research included the development of a messaging system, an information exchange platform, and a negotiation mediator.

Researchers from Siemens, Carnegie Mellon University's Silicon Valley campus and QuEST joined UC Berkeley in this project to enable integrated customer-controlled, price-based demand-response management.

In identifying and developing value to the customer, the research team employed a literature review and interviews to understand the types of users of the platform (e.g., business owner, employee, janitorial staff) and a range of values (e.g., comfort, convenience, cost-savings). The team then developed use cases, and developed a user interface that showed the energy consumption, indoor temperature, provided a slider bar for the customer to select the desired balance of cost savings versus comfortable conditions, and provided simulations so the user could understand the implications of the cost-comfort selection.

The system architecture of the XBOS-DR and EPIC platforms were modular, so that the various components could be developed and tested separately. The Siemens and CMU team developed the price messenger and tested it with both simulated and real utility event signals. The information exchange module included many standards-based information models. The BETS team developed the XBOS-DR platform: creating the drivers (interfaces) for various hardware, such as the thermostats, outlining and developing more than 10 microservices, and upgrading security services. They helped install, integrate, and maintain the hardware and software, developed an improved method of storing, categorizing, and acquiring the data through the Brick schema, and developed models for each building, and tested advanced control algorithms such as Model Predictive Control.

The research team used laboratory spaces on campus and three offices off campus as initial laboratory testing of the networked thermostats, and electric utility meter, and in one case, Electric Vehicle charging and various kitchen appliance loads. After recruitment and initial building audit of several buildings, the team installed the platform consisting of a miniature computer, networked thermostats, and metering in 16 small commercial buildings: 14 in Pacific Gas and Electric territory and two in Southern California Edison territory. For each building, the team procured historical electricity usage for the previous 24 months. After the system was installed, the team commissioned the equipment and monitored the data. In the summer of

2018, the team conducted several tests during peak day events in many of the buildings; some buildings were not able to participate. The team worked to analyze the data, correct data quality issues, build models, develop simulation tools, and develop zone-level analysis tools. In summer 2019, the team conducted a couple of tests to refine the procedure.

The technical barriers included intermittent Internet and other networking issues, data quality, insufficient Application Programming Interface for the networked thermostats, and the short range of the environmental sensors. Non-technical barriers included misunderstandings and miscommunication between the subject customers and the project team, unanticipated length of time to develop the user interface, and difficulty in obtaining the utility price signal directly from the utility.

The networking issues caused the XBOS-DR development team to reconstruct the secure data bus to accommodate local control; a couple of times, the customer's IT team made changes to the internal networked which removed the connection of the research team's system. The Green Button data was continuously collected throughout the project, but often had missing data; the team developed a script to detect the missing data. The team also had to find ways to work around the thermostat API to achieve the needed functionality (e.g., access the programmed schedule, force the system into first stage cooling). One of the TED meters was wired backwards and needed to be rewired; another TED meter malfunctioned and needed to be replaced. One of the customers informed us that the building was a designated cooling center, so the team could only control six of the HVAC zones; two customers (fire-stations) asked that the team not change the temperature setpoints, which essentially eliminated those buildings from the study. The project team found a couple of ways of receiving the price signal, including scraping the data from the internet.

In general, the researchers achieved project goals in using networked thermostats to reduce energy consumption, demonstrating the prototype XBOS-DR platform to reduce peak loads on event days in small commercial buildings, and demonstrating the price messaging and information exchange module functionality. The team was able to integrate lighting reduction in one building and EV charging and appliance loads in another building. The research team was not able to integrate photovoltaics and storage in the commercial buildings due to the insufficiency of the interface of the existing systems.

The research team identified several major lessons learned. There is an incredible value of incorporating real-time building energy data with thermostat data: one can achieve building system identification, conduct diagnostics, and improve control. The research team learned the difficulties in commissioning and managing multiple buildings. The team acknowledges more time is required to develop a user interface and address data quality issues.

Additional research that would further the goals of the project would be further testing of the user interface, testing the system with a different networked thermostat with an API that provides the functionality desired, and to continue to deploy and test different control and diagnostic strategies.

CHAPTER 6:
Benefits to Ratepayers

This project is anticipated to result in the ratepayer benefits of greater electricity reliability and lower electricity costs through enabling more effective use of DR and distributed generation resources, to help manage issues anticipated from: 1) the increasing integration of intermittent power generation into California's grid under the state Renewable Portfolio Standard program, and 2) related issues anticipated from increasing electric vehicle charging. The project technology is also anticipated to increase penetration of building management systems in smaller commercial buildings, resulting in energy cost reduction through improved end-use energy efficiency. In addition, increased safety is expected for energy end-use equipment through increased capability for remote monitoring and potential integration with alarm services.

The technology could be adapted to the residential sector at minimal costs; different drivers would need to be developed depending on the types of hardware included.

The project estimated the following specific impacts and benefits of the proposed aggregated demand-response and integrated energy management program, with supporting rate design and open-architecture software platform for large and small commercial sectors:

- Greater reliability of the electricity infrastructure, reducing frequency of outages.
- $260 million per year reduction in energy costs for ratepayers in 2024—derived from: lower demand charges (reflected for the utility as lower electricity infrastructure upgrade and operating costs), increased electric grid energy efficiency, reduced energy end-use from persistent efficiency in parallel with DR, and lower electricity generation costs (from lower-cost intermittent greenhouse gas (GHG)-free electricity generation with less need for spinning reserve, storage or high-cost supplemental peaking generation).
- 450 MW of avoided or shifted peak electric demand in 2024. This is a 150% increase beyond the 293 MW of DR from a combination of nonevent-based programs, critical peak pricing, and peak-time rebates estimated by the California Energy Demand 2016-2026 Revised Forecast [15].
- 180 million kWh per year and 18 million therms per year of reduced energy use in 2024 from persistent end-use energy efficiency achieved in parallel with demand-management.
- 930,000 metric tons of carbon dioxide (CO2e) emissions per year avoided in 2024 from: increased electric grid energy efficiency, increased end-use energy efficiency in parallel with demand-management, and increased fraction of intermittent operationally GHG-free renewable electricity generation (and decreased need for GHG-intensive supplemental peaking generation).

GLOSSARY

Term/Acronym	Definition
DR	Demand Response
XBOS	eXtensible Building Operating System: a platform for managing applications and data, and providing means for controlling systems in the buildings

REFERENCES

Azar, Elie and Carol C. Menassa. 2014. “A comprehensive framework to quantify energy savings potential from improved operations of commercial building stocks.” Energy Policy 67: 459-472.

Brook, Martha. 2012. California Residential and Commercial Energy Use Characteristics. http://www.energy.ca.gov/ab758/documents/2012-10-08-09_workshop/presentations/Day-2/AB-758_Market_Characterization_and_Scenarios_2012-10-09.pdf

Ehrhardt-Martinez, Karen. 2016. Meta-Review of Behavior-based Energy-Savings Potential Estimates for commercial buildings. Presented at the Behavior Energy and Climate Change conference. Baltimore, Oct 18-22.

Ehrhardt-Martinez, Karen. 2016. “Behavior-based Energy Savings Opportunities in Commercial Buildings: Estimates for Four U.S. Cities.” In the proceedings of the ACEEE Summer Study on Energy Efficiency in Buildings. Washington, DC: ACEEE.

Hong, Tianzhen and Hung-Wen Lin. 2013. “Occupant Behavior: Impact on Energy Use of Private Offices.” (LBNL-6128E) Berkeley, CA: Lawrence Berkeley National Lab.

Wikler, Greg; Sathe, Amul; Swamy, Surya; Ehrhardt-Martinez, Karen; Daftari, Aayush; Oztreves, Semih; Pierce, Julie; Menon, Carishma; and Jack Cullen. 2016. “AB802 Technical Analysis: Potential Savings Analysis.” (Ref No.: 174655) Prepared for the California Public Utilities Commission.

APPENDIX A:
Customer Value

Electricity customers typically do not understand tariffs and do not have the time to engage in bidding, shedding or shifting load, yet the purpose of this project is to help electric utilities who need improved integrated distributed resources to maintain a resilient grid. For the electrical utility to engage small and large commercial buildings in demand response programs to reduce electrical peak power demand periods and improve grid reliability, the demand needs, values, and priorities of these various businesses must be considered. Small commercial buildings include grocery stores, banks, retail, cafes/restaurants, offices, and others. Each business has different appliances, HVAC, or lighting loads, a different load profile over the course of the day, and different needs with respect to a business model. This chapter describes efforts in understanding who potential customers are, what are their values, needs and preferences, and how can the researchers include these in design and outreach decisions.

Customer Values, Needs and Preferences

The purpose of this subsection is to briefly outline the types of customers that might use the XBOS-DR platform, and their values and preferences. Two student teams worked to develop information on potential customers; this subsection draws in part on their work both in developing usable interfaces as well as path to market. The researchers used these values to develop use cases for the XBOS-DR platform and inform the development of the user interface that promotes customer choice in responding to price signals.

2.1.1 Target customers

A precursor project called OpenBAS funded by the Department of Energy (summarized in Peffer et al 2015) led to two UC Berkeley student projects: a Cleantech to Market business course and a user interface study. The interviews conducted for those projects provided a framework for much of this memo.

Cleantech to Market business course project

The 2015 UC Berkeley Haas Business School Cleantech to Market team devoted a semester to researching and developing a tech to market plan for XBOS. The Cleantech to Market team was tasked with identifying the best applications for XBOS as well as a market entry strategy for these applications. The team used design thinking, the lean start-up methodology, desk research, and over 40 interviews to generate an initial list of more than 40 potential applications and a phased market strategy. The team members were Dan Curran, Roel Dobbe, Robbie Heath, Jared Landsman, John Maus.

Some of the key findings were about the building manager, the equipment manufacturers, vendors, and service providers. Large commercial buildings often have complex and proprietary BMS; if the Building Manager cannot operate or make changes to the BMS, he/she must call a service contractor or provider to perform updates, repair or maintenance (Curran et al 2015). The Building Manager often has to take the time to work closely with the service provider or technician since this technician is not usually the same person who installed, named, or programmed the system and its many components.

Another issue is split incentives between owners and tenants for energy efficient equipment. Sometimes the Building Owner pays the utility bills; more often the Tenant pays the bills, so there is often a split incentive in upgrading to more energy efficient equipment, since the tenant will reap the reward of lower bills while the owner fronts the cost.

Contracts for a particular Building Management Systems (BMS) are won by the firm with the cheapest bid that meets the specifications outlined by the general contractor or architect (p. 15, Curran 2015). Most of the work is in labor—in connecting and configuring hardware, understanding and writing software, and troubleshooting and verifying systems. Currently it is difficult to connect new devices, programming languages are inefficient, data is stored in proprietary systems, and there is no automatic updating of BMS (Curran et al 2015).

User Interface course project

The OpenBAS Usability Research project was conducted for the INFO 214: Needs and Usability Assessment course in Spring 2015, led by students Wenqin Chen, Bobby Davis, and April Dawn Kester. Over the course of the semester the team conducted a heuristic evaluation, diary study, and usability tests on the current OpenBAS web interface, features and functionality. The team also conducted interviews with market experts, software engineers, current users and potential users of the system. With the initial stages of the research complete, the team was able to make User Interface recommendations using card sorting and participatory design methods. These methods were conducted with current users of the OpenBAS web interface.

The following are the findings from preliminary research studies on key end users using the OpenBAS system:

Office Manager:

The office managers indicated that the existing manual methods of controlling these systems adequately perform the required functions. However, the office managers did indicate that the concept of a digital dashboard would be appealing if it could present status from all systems and provide control. The XBOS controls would need to be easier to manipulate than the existing manual solutions.

The office managers felt that the existing data visualization interface showing energy consumption is hard to interpret or translate to action. Moreover, energy consumption is not a high-priority issue in the context of the day-to-day job function of an office manager.

Property Manager:

The users had difficulty navigating the interface, could not easily distinguish interactive elements of the User Interface from static elements, and did not understand the value of many of the features.

Building Manager:

The building managers found the temperature control as the most important feature followed by lighting control. Monitoring energy usage is desirable but is challenging to present in an understandable and actionable way for a building manager.

List of potential users

Typically, the end users can be classified into two categories – Primary users and Secondary users.

Type of end users:

Primary Users

- Business Owner
- Building Owner
- Property Manager
- Building Manager
- Office Manager
- Bill Payer
- Occupants (Employees, Clients)

Secondary Users

- Janitorial (e.g., regular cleaning, often after hours)
- Contractor—service provider for maintenance (change lightbulbs, fix leaks, repair HVAC system)
- Manufacturer/Vendor/Installer of building components or equipment
- Energy services provider

One energy service provider commented that most of small and medium businesses have just the business owner and a janitor, who works for the building owner. In a few cases, there may be a service contractor. In 10,000-20,000 sf buildings there is an office manager. For typical small buildings, HVAC and lighting are managed by the owner or

the employee that happens to be working at the time. The medium size businesses have an office manager that probably manages energy and possibly reviews the energy bills, though not always. There may be a service contractor working for the owner that is available when there are problems.

2.1.2 Target Values and Needs

All of these users have different relationships with the buildings. These different concerns about building operation equate to the need for different information and control of the system to meet each stakeholder's need.

Business Owner:

The Business Owner who occupies a small commercial building often performs the role of property manager, building manager, and the office manager. Convenience, cost, low-risk, comfort of employees, and easy maintenance are important values.

For Business Owners who occupy large commercial buildings, the owners care about the high-level operations within the building. Their primary value needs revolve around the security and safety of employees, and potentially the cost.

Building Owner:

The owner of the building sometimes influences initial equipment and service choices in a building (often these are made by the architect or engineer in the construction phase), but more typically makes decisions about major equipment upgrades. Cost and payback is a key value for building owners, as well as low risk.

Property Manager:

The property manager seeks to ensure that the building is operating smoothly and efficiently to meet the client's needs. Both cost and safety conditions of the building are values.

Building Manager:

The building manager's primary concern is to keep the building operations (e.g., HVAC, lighting, cleaning, supplies, water, waste) running on a daily basis so that the people in the building are comfortable and able to perform their functions in a safe environment. The building manager is often the person that receives complaints from the occupants if the building is not operating satisfactorily. Thus values include an easy to understand system, easy and low cost maintenance, and an easy to use system. Building Managers also value low risk solutions.

Office Manager:

The office manager's primary objective is to ensure that the employees are productive and can do their job, such as with appropriate equipment, furniture, and with adequate indoor environmental quality, which includes light, temperature, air quality, noise. The

office manager also seeks convenience in modifying these conditions. The cost savings and easy maintenance are additional values to the office manager.

Bill Payer:

The person who pays the utility bills wants to ensure that there are no discrepancies with the bill, it is clear what is paid for, and the value of the service (such as energy) is appropriate to the cost. Some businesses recently are complaining about the high cost of demand charges. If a business exceeds a certain demand threshold (e.g., greater than 200 kW) within a certain time period (e.g., three months), this can trigger entering a new higher tariff for the next time period (e.g., 12 months). So providing means for these businesses to avoid entering higher tariffs is quite valuable.

Employees or Occupants:

The employees and occupants want good indoor environmental quality (e.g., adequate lighting and temperature conditions) within the building. Another value for the employees is the convenience of changing these lighting and temperature conditions.

Janitor:

The cleaning staff are often the last ones to exit commercial buildings. Janitorial staff require adequate light and thermal conditions to perform their tasks, often turning on systems that were turned off after hours. They need easy entrance to the building after hours, but do not want to be responsible for security issues when entering or exiting the building.

Contractor:

The service and maintenance contractors would like steady and reliable work, hence the popularity of typical service contracts for proprietary BMS systems (p. 17 Curran et al 2015). They value more detailed information about type of repair or replacement (broken fan belt, damper or valve). Their objective is to find the solution to the problem as fast as possible, and come prepared with the right material to make the repair. These stakeholders value increasing their value proposition by combining various services (e.g., security with maintenance).

Equipment Manufacturers, Vendors and Installers:

Manufacturers, Vendors, and Installers of building components and equipment would like to grown their market and sell more product. Thus a key value for these stakeholders is easy to install, easy to diagnose and verify, and inexpensive.

Energy Services Providers:

Companies that provide energy services (e.g., reduced demand charges, demand response, reduced energy costs through energy efficiency and management) require convenient and low cost access to energy data such as that from whole building interval “smart” electric meters.

In summary, values include:

- **Safety:** adequate light, fresh air/ventilation, appropriate thermal conditions, furnishings
- **Security:** doors lock, controlled access
- **Privacy:** restricted access to building data and control
- **Low risk:** in terms of insurance for physical premises
- **Comfort**: thermal comfort, good light, little noise, good air quality
- **Convenience:** time savings measure (remote control, ease of maintenance)
- **Cost:** rent, salaries (productive employees!), demand charges, equipment, maintenance, energy, and balance of all of these.

2.2 Customer Use Cases

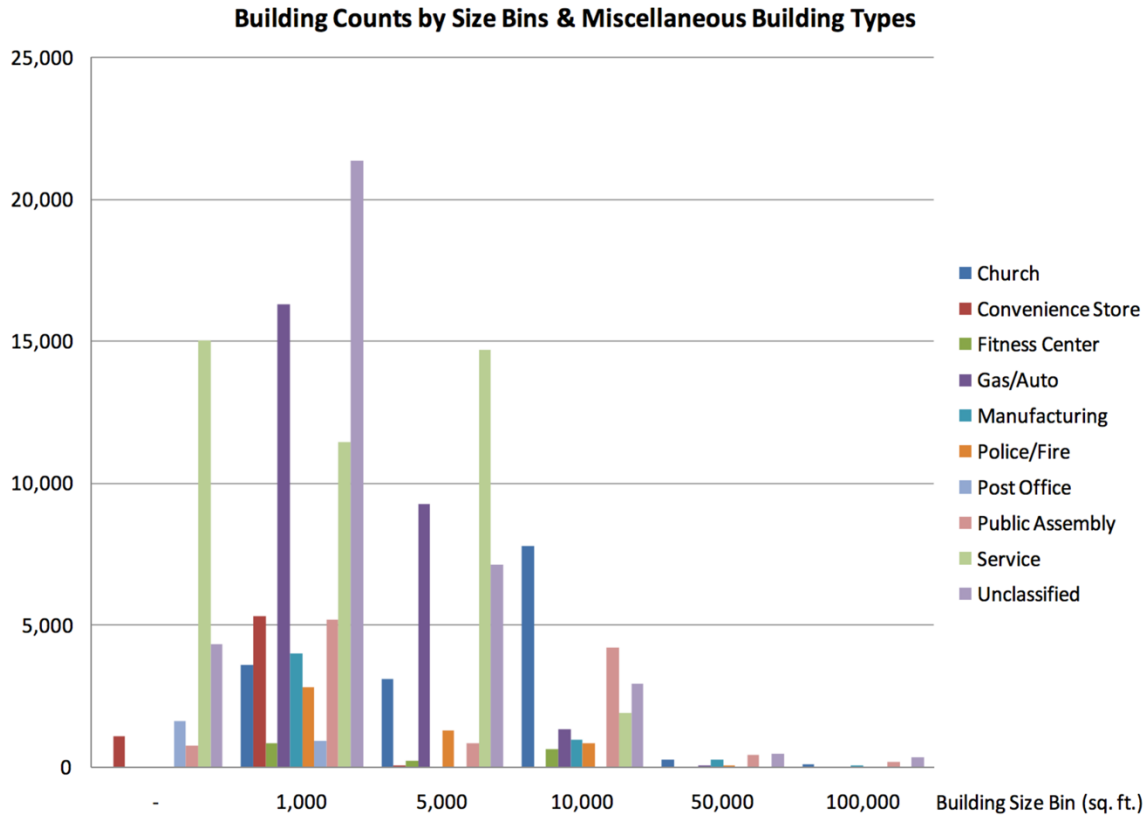
The researchers used these values to develop use cases that represent the ways these customers might use the XBOS-DR platform to meet their needs and preferences. Reviewing the various types of businesses represented in the potential customers, three types of buildings—office, retail, and education—have shown the greatest energy efficiency potential (16-33%), followed by healthcare and lodging. Lighting, HVAC, and computers showed the most promise as targets of energy efficiency measures; hot water was another end use that showed potential for energy savings. Finally, a prioritized list of use cases provides scenarios using XBOS-DR to meet these needs.

2.2.1 Types of Customers and their loads

This section defines the heterogeneous nature of potential customers. A myriad of businesses and organizations are housed by commercial buildings: churches, retail stores, restaurants, auto dealerships, education and so on.

In the US in 2012, there were 5.6 million commercial buildings, comprising 87.4 billion square feet of floorspace (CBECS, 2012). Most of the commercial buildings are relatively small: less than half are less than 5,000 square feet, and nearly three-fourths are smaller than 10,000 square feet (CBECS). Figure 1 below shows the same pattern of lots of small buildings across a variety of building types in California.

Figure 23: Numbers and types of commercial buildings in California

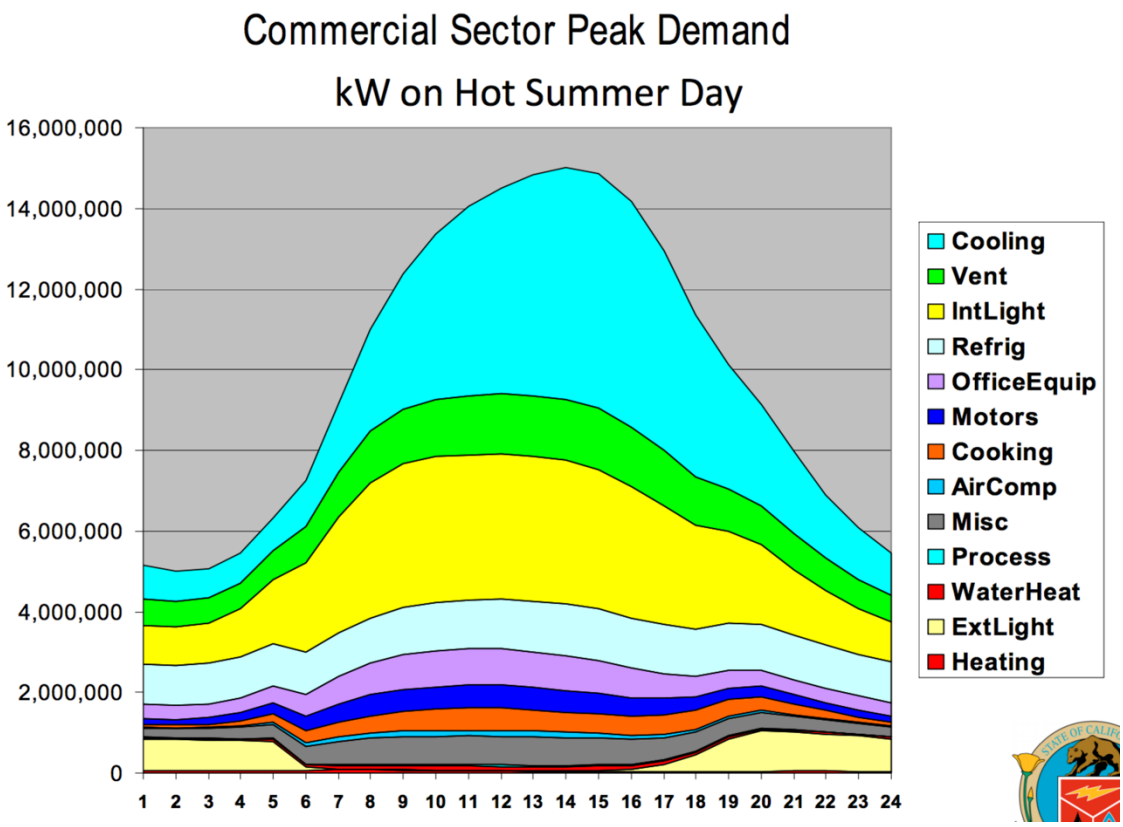


Lots of small buildings comprise California's small commercial buildings.

Credit: Brook 2016

In California in 2012, commercial buildings consumed the largest portion of the electricity consumption at 37%, and contributed 37% of the peak load (Brook, 2012). The figure below shows the end use profile of the peak demand on a hot summer day. Most of the peak load is due to cooling followed by interior lighting.

Figure 24: Load profile of commercial buildings in California

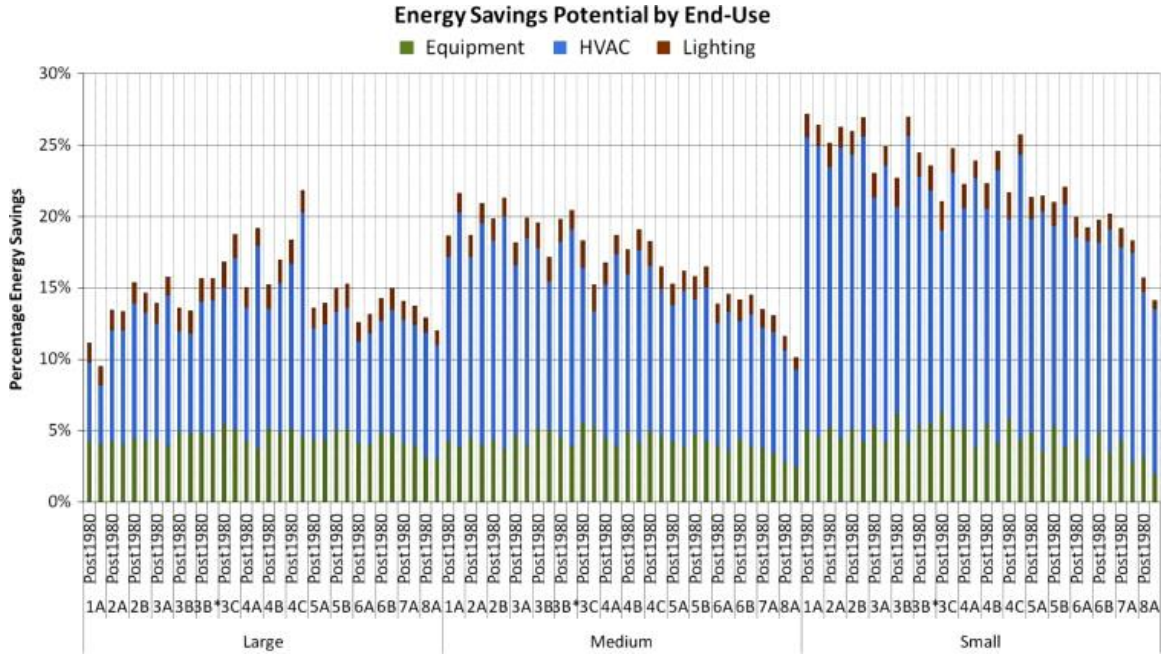


Most of the peak load is cooling and lighting on hot summer days.

Credit: Brook 2016

In a recent behavioral meta-study on commercial building energy efficiency potential, three types of buildings—office, retail, and education—showed the greatest energy efficiency potential (16-33%), followed by healthcare and lodging (Ehrhardt-Martinez, 2016). Lighting, HVAC, and computers showed the most promise as targets of energy efficiency measures (Ibid, 2016). One study (Figure 4) showed that HVAC-related savings potential was the highest (HVAC Savings by End Use: Large buildings: 5-15%, Medium 7.5-17%, Small 10-23%), followed by equipment (5-15%) and lighting (2-3%) (Azar and Menassa 2014). Hot water was another end use that showed potential for energy savings. In addition, the size of the building made a difference; the savings estimates were larger for smaller buildings.

Figure 25: Energy Savings Potential in Commercial Buildings



Mo Simulation results for energy savings potential by end use and commercial building size

Credit: Azar and Menassa, 2014

At minimum, for best effectiveness, XBOS-DR should control HVAC loads—especially cooling; lighting, computer and other equipment, and possibly hot water may also have potential for reducing peak load.

2.2.2 Energy Efficiency and Demand Response Use Cases

The goal of this section is to develop use cases for the XBOS platform that take into consideration:

- the objectives of the project to create a demand response management system for commercial buildings based on customer values
- the criteria for suitable devices and applications to be demonstrated on the platform.
- identifying the potential commercial buildings that the XBOS system could support such as university buildings, retail stores, and offices.

People carry out different tasks every day in commercial buildings. Achieving energy efficiency in these buildings requires the interaction between a number of actors and entities providing energy monitoring and consumption feedback, using automation systems, sensors and actuators, and carrying out economic strategies to save energy. In order to build a set of requirements, the team developed the use cases of the XBOS

system since it can lead to deeper insights of the interactions between the different actors and the XBOS system.

Physical functions

The ultimate goal of the project is to provide price signals to the XBOS-DR platform in various buildings, and have the XBOS-DR platform of each building (or the Negotiation Mediator for a campus of set of buildings) respond with a forecasted demand (e.g., not to exceed power demand per hour, 90% confidence interval for demand). The system will use machine learning for predicting and forecasting load. The team brainstormed potential functions of the XBOS-DR platform for both small commercial and larger commercial buildings. Categories of functions included HVAC, lighting, plugload, process load, storage, and human activities. The major difference between small and commercial buildings is in the HVAC system. The following tables describe potential functions for each category.

Table 1: Energy Efficiency and Demand Response strategies for commercial buildings.

Lighting	Energy Efficiency	Demand Response
Daylighting	Turn down lights when enough light through daylight (need sensors)	Turn off lights in daylit zones
General lighting	Use schedule or timers for general, especially to turn off during non-business hours. Use occupancy sensors if possible.	Turn off lighting if possible (e.g., overlighted spaces such as corridors that have both emergency and regular lighting.) If continuous dimming possible, dim lights progressively based on price.
Task lighting		Encourage use of task lighting instead of overhead general lighting where possible.
Discretionary signage		Turn off when possible
Plug loads (discretionary)	Turn off appliances when not in use (coffee/hot pots, standing desks). Turn off appliances during non-business hours (copiers, printers, computers, water cooler/heaters).	Shift load outside demand response period, unless needed (dishwashing) Coordinate (cycle) plugloads to reduce overall demand (e.g., laptops).
Process loads (bakery, data centers, small manufacturing)		Shift load to outside demand response period
Storage		Coordinate with demand (including EV charging/storage) and/or on-site generation (PV)
Human—change business activity		Shift hours/flex scheduling to avoid high demand work during demand response period.

Source: Therese Peffer

The following two tables describe strategies for HVAC in small commercial buildings, with multiple thermostats and packaged HVAC Roof Top Units (RTUs), and large commercial buildings, with Building Automation Systems (BAS).

Table 2: HVAC strategies for Small commercial buildings

	Energy Efficiency	Demand Response
HVAC via thermostat		
Zone temperature (deadband and schedule)	Implement deadband of 70°F-74°F (21.1-23.3C) during operating hours and 65°F (18.3C) (heat)/80°F (26.7C) (cool) for non-operating hours (nights, weekends, holidays)	Increase to 78F (25.6C), allow "cool blast" (immediate short-term occupant-control). Consider the method for setpoint change (stepped, exponential)
Coordinate multiple HVAC units	Prevent simultaneous heating and cooling in adjacent zones. Reduce demand charge by cycling HVAC units so minimize coincident use.	Progressively cycle different RTUs based on priority (e.g., cycle 25% for 30 minutes for regular period, then cycle 50% for 30 minutes for critical period). (QuEST)
Advanced controls for multi-staged systems, controllable dampers	Use outside air when possible. Use one HVAC RTU for ventilation, the others for cooling/heating when possible	Force RTU into first stage for DR event
Fault detection	Send alert when HVAC is running during non-business periods. Send alert when zone does not reach target temperature	(same)
Precooling/heating		Precool or preheat before DR event

Source: Therese Peffer

Table 3: HVAC strategies for Large commercial buildings.

	Energy Efficiency	Demand Response
HVAC via BAS		
Supply air temperature	Static increase to 58°F (14.4C); dynamically change with supply air pressure.	Increase to 60°F (15.5C); dynamically change with supply air pressure, increase the threshold. (Center for the Built Environment)
Zone temperature	Implement deadband of 70°F-74°F (21.1-23.3C) during day and 65°F (18.3C) (heat)/80°F (26.7C) (cool) at night (10p-5a)	Increase to 78F (25.6C), if possible, allow “cool blast” (immediate short-term occupant-control) as with Comfy or other
Reduce ventilation rate	Reduce minimum ventilation rate by 30-70%; reduce by 70-85% at night. Consider Time-Average Ventilation. Use demand controlled ventilation in conference rooms.	Reduce minimum ventilation rate by 70-85% for short periods with air monitoring. Consider Time Averaged Ventilation with longer cycles (Center for the Built Environment)
Increase condenser water temperature		(needs empirical testing—thought to be best for hot days)
*VFDs on chilled/ condenser water pumps	(automatically controlled)	(not pursued)
Fault detection	Send alert when HVAC is running during non-business periods. Send alert when zone does not reach target temperature	(same)
Precooling/heating		Precool or preheat before DR event

Source: Therese Peffer

2.3 Customer facing functions

The next step is to consider these functions from the user point of view. The most common use cases from the customer side are:

- Monitor and control energy use manually, remotely, conveniently
    - Turn off lights or remotely change schedule for business
    - Check current temperature and modify temporarily during the day, for holidays, or seasonally
- Engage in demand response
    - Choose level of engagement and priorities of curtailment
    - Receive alert and allow control for real-time demand response
    - Help to shape demand
        - Load following (e.g., PV)
        - Integrate EV or storage
- Automatically manage energy use to reduce energy cost
    - Reduce demand charge
    - Check acute alerts/alarm notification
        - E.g., Energy use during closed hours or Zone not reaching target temperature
- Check benchmarking in aggregate

These preliminary functions will drive the functionality of the user interface. The following table describes some of the potential functions:

Table 4: Prioritized list of functions for the user interface

Priority	Function	Description	Output to Customer
0	Demand Response Enabling	Through the user interface, Customer will receive ability to prioritize load curtailment (e.g., which thermostat, how much change in temperature setpoints) during 5-10 simulated demand response events throughout the test period.	Customer's computer
0	Energy Savings Estimate	Before the XBOS-DR platform is installed, the team will use previous utility bills and Green Button data to identify current demand charges and identify areas to reduce overall energy, and provide to the customer an estimate of average energy savings (likely 10-20%).	Estimate of energy savings
1	Diagnostics	Upon installation of the XBOS-DR platform, the commissioning period includes discovery of problems: -rogue zones (that do not reach set temperature), -systems on in off-hours (graphic notation) -respond to temperature overrides (occupancy info will help) -problem periods of the day (e.g., west facing zone in the afternoon, look at ramp rates of temperature), -malfunctioning dampers, fans -ducts not connected or damaged (How is this implemented? Work with Notification—watch out for false positives. Open door in hot climate, tstat in bad place)	Report every month that describes issues
2	Cost reduction due to Demand charge	After the commissioning period, algorithms will reduce demand charges by identifying and monitoring periods of peak demand (e.g., multiple RTUs on at the same time, other coincident loads (e.g., EV charging, refrigeration), and	Included in monthly energy report

	reduction and maintenance	coordinate the loads to ensure that the load does not exceed the demand charge threshold. (Request 12 month analysis from utility)	
3	Cost reduction due to Energy reduction	After the commissioning period, algorithms will reduce energy consumption while maintaining comfort (e.g., by incorporating schedules to reduce consumption during non-business hours, gently expanding temperature deadbands). Perhaps ask the customer how much savings do you want?	Included in monthly energy report
4	Notification service	The research team will work with the customer to provide an appropriate level of notification (the interval and urgency may be modified). Examples include: security breaches (detection of occupancy when none expected), excessive energy consumption (operation of building equipment during non business hours), temperature setpoints not met, etc	Frequency and urgency set by customer (some people want quarterl!!)
5	Convenience of control	Customer will receive and be instructed on how to use a web-browser user interface that will display all thermostats and other systems under control (lighting and plugloads). Turn off lights remotely. One touch changing thermostat setpoints for holiday (non-business).	Customer's computer
6	Energy report	The user interface will provide a means of viewing: Compare this building with benchmarks (e.g., EUI's for current best practice for this type/size building, other similar buildings type/size/age, other buildings in XBOS-DR pilot program) Current and historical whole building and system energy display (e.g., by identification, each RTU, lighting bank, plugload) Means of easily comparing (e.g., Monthly energy compared to this time last year normalized for weather) Estimate of current (e.g., estimate of cost to date this month from interval meter data) and historical cost (e.g., use GreenButton and utility data to	Summary energy report each month.

		provide actual historical billing info, able to summarize cost per year, per season) Energy savings so far, shown in cost savings, CO2 emissions averted, trees, happy polar bears Peak demand so far this period, with demand threshold Advice and tips: Increase temperature setpoint on hot summer days to save additional 5%! Is there a tool that we can build in to help choose programs? Understand the best solar strategy (sizing number of panels or battery, orientations of panels optimized for load)	

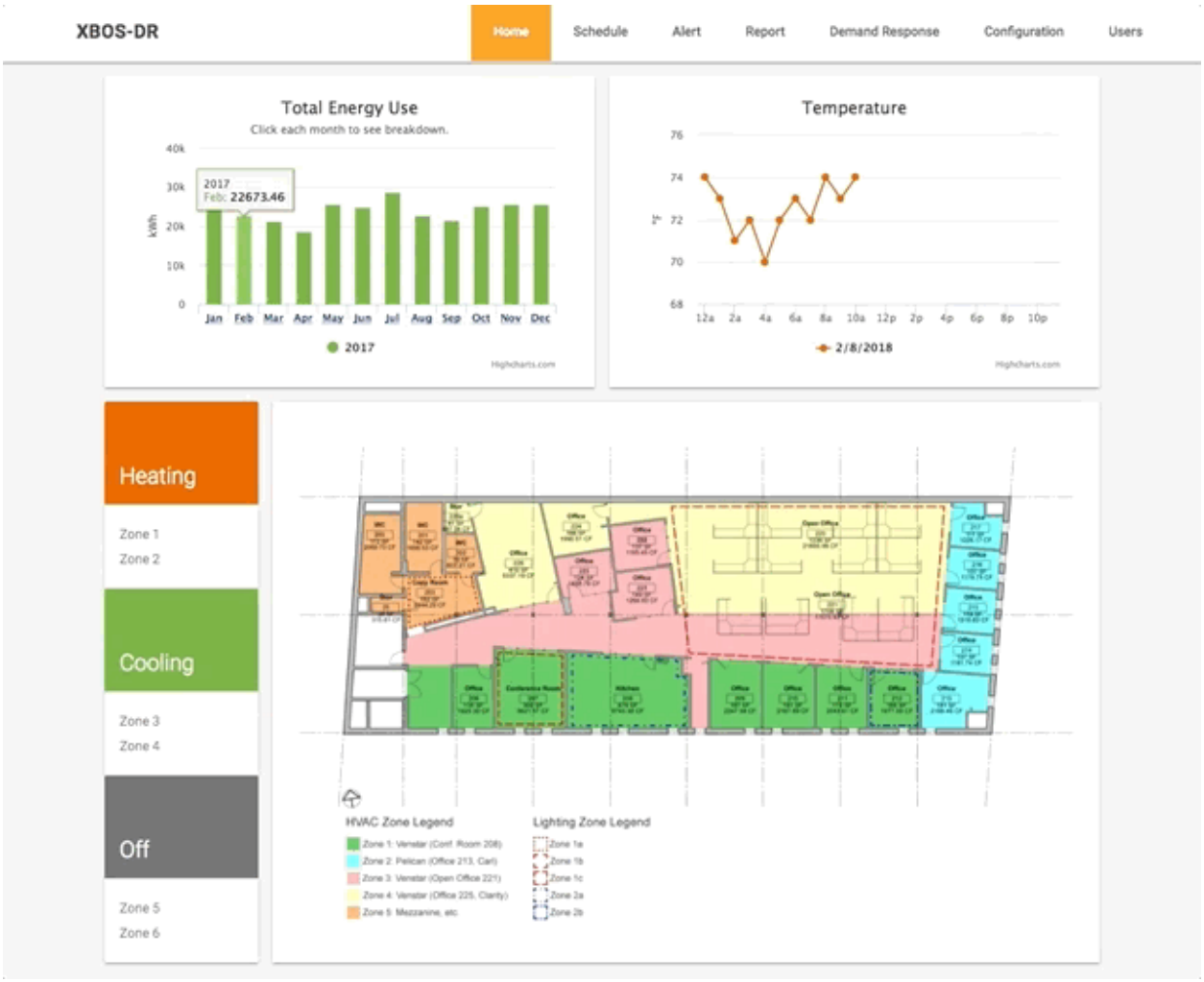
Source: Therese Peffer

2.4 User Interface

The main sections of the user interface are the Home screen, Schedule screen, DR screen, and Settings. See Appendix F for more details.

The home screen should provide the most utilized functions: energy use from real-time and utility provided data, and price for the day so far, indoor temperature in various zones, current state of Heating Ventilation and Air Conditioning (HVAC) and lighting systems displayed on a floor plan, and demand response notification. Figure xx shows an early design of the interface.

Figure 26: Early Version of the XBOS-DR User Interface



Source: Therese Peffer and Brandon Berookhim

APPENDIX B:
Front End Services: Price Messaging, Information Exchange, and Mediator

Siemens and Carnegie Mellon University—Silicon Valley Campus developed the EPIC platform, which includes the price messaging system, the information exchange module and the negotiation mediator. All code is on the XBOS-DR github site (https://github.com/SoftwareDefinedBuildings/xbos).

Instructions on administering the platform may be found https://drive.google.com/open?id=1xvfFQMRthFoncCDUm8_26BfetgN1VxWN.

Instructions on using the platform may be found here: https://drive.google.com/open?id=1M3-9JZKhpPWFDWizTRs9MoqRqFQ6dr1I.

The technical specification may be found https://drive.google.com/open?id=1FJi5kr59TnbUM5gsHYtW3Df0WnHmWE1e.

Develop the Price Messaging system

3.1.1 Outline applicable compliance regulations

Since our partners generating price or DR event signals (EPRI, PG&E and SCE) were not interested in receiving any responses from our systems (such as demand forecasts), our interactions with outside energy suppliers consisted solely of receiving (and not sending) price and demand response signals. Accordingly, the only compliance requirements were to support the communications, security and authentication specifications needed to poll for such signals. This was done using the OpenADR protocol version 2(b) and security keys provided by a third party.

3.1.2 Develop the price messaging software module

The Price Messaging module implements the default workflow of transactive signals from the utility to the building management system, and the forecasts back from the building management system to the utility. It also implements the workflow between buildings based on building data, which is a capability available for future use.

3.1.3 Integrate with other modules

The Price Messaging Module (i.e., the EPIC platform) integrates 3 other service modules: (1) the PullerPublisher on the utility side, (2) the Information Exchange Module, and (3) the Messaging Module. In Section 3.1.2 we discussed the models and workflow associated with conveying the transactive signal in OpenADR format between the utility and the building.

3.1.3.1 The PullerPublisher

The EPRI transactive signal is provided through an OpenADR VTN api using a polling implementation and TLS certificates for authentication. The EPIC platform uses a publish and subscribe mechanism with the design assumption that, at some point in the future, the CEC will want to round-trip price and forecast events, possibly at varying rates. As such, an intermediate layer, called the PullerPublisher, was created to handshake with the EPRI VTN, Pacific Gas And Electric (PG&E), and Southern California Edison (SCE). The PullerPublisher retrieves the OpenADR DistributeEvent in XML format and initiates an EPIC publish web service call. It should be noted that the PullerPublisher only pulls price events and does not push them. At present a separate web service is required to retrieve the return forecast events since none of the providers are set up to receive building forecasts.

3.1.3.2 The Rest Hooks messaging system

Once the content of a message is translated from XML to RDF, and converted from its native model to the SAIM information model, it is then published to the messaging system, which is implemented with the Rest Hooks. Rest Hooks is a topic-based Restful publish and subscribe system. Rest Hooks maintains a list of subscribers and credentials. Events delivered after a subscription request is made are delivered to subscribers directly from Rest Hooks. As a result, the EPIC platform captures the content as it is being published.

3.1.4 Work with Group 3 recipient (EPRI) to understand interfaces and document

As specified in the grant agreement, the system should be able to receive and process transactive signals coming from the Group 3 recipient (EPRI). EPRI chose to implement the transactive signal using the OpenADR 2.0b specification, which is able to communicate electricity prices for sets of intervals. EPRI sends an OpenADR DistributeEvent message each day, containing 24 1-hour intervals for the following day, with a price for each interval derived from the wholesale ISO price. We (using the PullerPublisher) subscribed to both the PG&E and SCE signals that EPRI generates, first with test signals and servers and later with their production servers where possible.

While our system successfully receives these signals and passes them on to the XBOS clients, we also implemented two additional sources of signals that are more relevant to the commercial buildings in the pilot test. The first is a signal generated from the commercial tariffs actually used by the buildings. The second is the Peak Day Pricing (PDP) event signal generated from either PG&E or SCE for their customers during heavy demand days. The tariffs implemented were:

- PG&E A-01
- PG&E A-06
- PG&E A-10
- PG&E E-19
- PG&E E-20
- SCE TOU-GS-3
- SCE TOU-8-RBU
- Federal Flat Rate (for LBNL)

It is worth noting that these tariffs were implemented according to a new standard under development by NIST, NREL and Mission Data, as an extension to the CIM (IEC 61970, 61968 and 62325) standards. This new model can capture much more complex tariffs than supported by the existing CIM standards. To our knowledge, ours is the first and only implementation of the new model.

3.1.5 Develop mechanisms to receive the transactive signal

As mentioned in Section 3.2.3.1, the PullerPublisher is a web service that integrates the transactive signal polling services with the EPIC publish and subscribe messaging system. The workflow whereby the transactive signal is delivered to XBOS clients

3.1.6 Incorporate the Transactive Signal Server and test in field and lab

3.1.7 Demonstrate the TSS within XBOS/DR and document in a Dispatch Demonstration

We had several integration tests with XBOS/DR, with increasing integration each time. The first was just a round trip communication workflow. The second was integration including live signals from EPRI, followed by PG&E and, later, SCE. The third was to receive pricing event forecasts. The fourth was to test the BRICK/data retrieval api for the Negotiation Mediator, and the last was to integrate the N.M. though we had to simulate the integration because the XBOS server stopped sending forecasts.

3.2 Develop the Information Exchange Module

The Information Exchange/Management module is the core component of the EPIC platform. It was designed to provide a system-agnostic information model that could enable interactions between the grid and building management systems, or between heterogeneous building management systems, which traditionally use different models.

3.2.1 Develop the criteria for suitable information models

There were three criteria for selecting information models for this project:

- Support the use cases associated with the project
- Use standards where possible
- Minimize complexity

The project was already committed to using OpenADR at the utility, to convey the pricing payload, and to use BRICK at XBOS, to convey building information such as meter and other sensor information values. It was determined early on that the FSGIM model would serve as an integration point for OpenADR and that OpenADR would be used to convey the pricing model to XBOS. It was also determined that, since XBOS communications didn't have to comply with the OpenADR message stack, the project would reuse the OpenADR oadrDistributeEvent structure to convey the forecast back from the building to the utility. In this way, the parsing of structure would remain the same in both directions, at least with respect to the pricing and demand information.

3.2.2 Locate, modify, and integrate suitable models

The models chosen for integration between FSGIM/OpenADR and BRICK are shown in the figure below:

3.2.2.1 Selected information models

- FSGIM (ASHRAE 201) - Energy models (from the standard, in OWL)
- OpenADR 2.0b - Price messaging model (from the standard, but converted from XML Schema to OWL)
- QUDT - Standards-compliant quantities, units, and dimensions models (from the standard, version 1.2)
- IFC - Building models (from the standard, in OWL)
- SAREF - Building models (from the standard, in OWL)
- SSN/SOSA - Sensor/Actuator models based on the SSO pattern (from the standard, in OWL)
- SSF - Sensor and Actuator models (adhoc, in OWL)
- EPIC Bldg - A building model that includes services (e.g., electricity, plumbing, etc.) and tariffs (created from a combination of sources, including a pre-standard tariff model from NIST, and Schema.org)

3.2.2.2 Model integrations (mappings and adapters)

Model integrations are done in two ways: (1) between ontologies that are part of the SAIM, and (2) between the models in the SAIM and legacy models such as BRICK. The mechanism for ontology mapping is the same in both cases.

- FSGIM <-> OpenADR
- FSGIM <-> QUDT
- IFC <-> QUDT, SSN/SOSA, SSF, EPIC Bldg
- SAREF <-> QUDT, SSN/SOSA, SSF, EPIC Bldg
- BRICK <-> QUDT, SSN/SOSA, SSF, EPIC Bldg

The bi-directional arrows represent integration mappings or adapters.

Model integrations serve the purpose of allowing the use of system-agnostic standards and other models to appear as a single database structure. As such, and if the right models are selected, and entire information domain can be represented and support interactions across different systems each having its own information model.

The process of integration is described in the technical specification in more detail, but essentially amounts to finding semantic intersections between models and ‘gluing’ them at these intersection points. Since standards are ‘curated’, they cannot be modified; only extended through inheritance. So the integration approach is to subclass the two models being integrated, if possible, and resort to other approaches if necessary. If either of the models being integrated isn’t curated, then direct integration can be performed, though this would be an undesirable scenario since we intentionally seek to use standards in all cases.

3.2.3 Integrate with existing XBOS data model

There is a section in the Technical Specification that describes in some detail how the BRICK ontology adapter was developed. Please refer to that document for details.

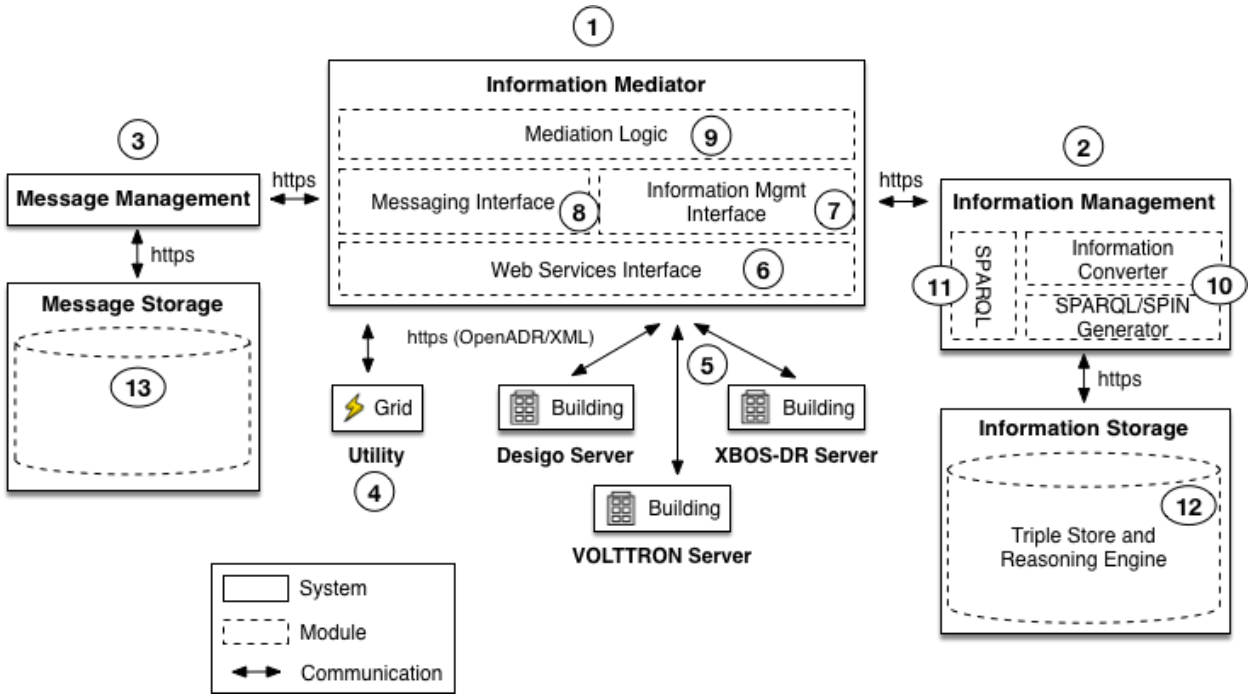
3.2.4 Prepare an Information Exchange Memo which summarizes the above

<To be described>

3.3 Negotiation mediator

The Negotiation Mediator was originally proposed as an autonomous computational unit that managed the interactions between the utility and buildings, but it was later changed to be a service running on the EPIC platform which, in the architecture diagram below, is now referred to as the Information Mediator. The Negotiation Mediator becomes active when a negotiation use case is identified. The Price Messaging System is the interaction manager. A high-level view of the architecture is shown in the figure below:

Figure 27: Components of Negotiation Mediator



The components of the Negotiation Mediator include the Information Mediator that looks at predicted energy consumption data from multiple buildings, Information Management and Storage, and Message Management and Storage.

Credit: Jack Hodges

A brief walkthrough of this architecture follows. The primary component of the platform is the Information Mediator (noted at 1 in the figure above), which serves as a common entry point and provides web service access points to the Information Exchange/Management services (at 2, through an interface at 7), and the Message Management (at 3, through an interface at 8). Interactions between the utility (at 4) or buildings (at 5) are mediated by the platform (at 6). Examples of different possible building management systems are shown in the figure, though in

this project only XBOS-DR servers are interacting with the platform. The mediation logic component (at 9) is part of the platform. Other aspects of the architecture, namely security, storage/retrieval (shown at 10-13) will be discussed elsewhere in this document. For a full discussion of the architecture please see the EPIC Technical Specification document.

3.3.1 Develop negotiation mediator (EPIC Platform)

The Siemens/CMU-SV team decided that a functional round-trip workflow was more important to design and develop before the Negotiation Mediator, so the design and development order was changed to reflect this change of priorities. The platform is a set of interacting web services that: (1) interact with the utilities and EPRI to obtain pricing events, (2) interact with buildings to receive pricing forecasts and building data, (3) manage messages, (4) convert, store, and retrieve pricing and building information, and (5) perform mediation. As a result, the team had to make choices about various approaches early in the program:

- Which web services architecture to use
- Which messaging system to use
- Which triple store to use

The Jersey/Grizzly java-based web services architecture was chosen to be consistent with the implementation of the Open Semantic Framework (OSF) which was going to be the interface layer to the the triple store.

Several messaging systems were looked at, including RabbitMQ, ZeroMQ, and Rest Hooks. Each had strengths and weaknesses, but it was desired to have the messaging system be an autonomous, restful system, and the only approach that satisfied this requirement at the time was Rest Hooks so it was chosen. All of the other approaches used web sockets, which was considered at the time a limitation in that the other services were to be restful.

3.3.1.1 Run data store performance tests

As performance of the triple store was going to determine success or failure of the approach, the choice of which triple store to use was addressed early in the program. Preliminary performance tests were run to compare triple-store insertion times for three different systems. The average time for an insertion is shown below for a trial of 1000 insertions, producing roughly 49,000 triples

Table 1: Average time for an insertion into the database.

Triple Store	Average insertion time (seconds)
Fuseki	0.084
AllegroGraph	0.175
Native TopBraid Live (TBL)	0.119

Source: Jack Hodges

3.3.1.2 Choose platform

For the sake of simplicity of the implementation, it was decided to use the native TBL triple store as it did not incur a significant performance penalty and does not require an additional interface to be built. During early implementation the TBL instance running on the native Topbraid Composer tool was used. Later this implementation was moved to an Amazon instance along with the rest of the platform, and ultimately a separate TBL license was used. The final AWS implementation platform was a t2.xlarge instance with 4 vCPUs, 16 GiB of memory, and 200 GiB of SSD storage.

3.3.1.3 Define mediator (EPIC Platform) architecture

Initially, we designed the architecture with the Negotiation Mediator as the principal component, with the other components supporting it. After developing the system, it became clear that the EPIC platform should be the master and the mediator a subordinate module. The Negotiation Mediator now operates as a module that responds to requests from the Price Messaging system.

3.3.2 Integrate XBOS/Information Exchange Module

The first portion of the Information Exchange/Management Module (IEM) is to ensure that transactive signals from Group 3, or utility, which are conveyed using OpenADR, Version 2, can be inserted into the Information Storage system (at 12 in Figure 1), which is based on the System Agnostic Information Model (SAIM). There are two aspects to this problem: (1) the information model (OpenADR and Building) aspects, and (2) the operational (platform and web services) aspects. Details of the implementation are described in the accompanying Technical Specification document.

3.3.2.1 Information (OpenADR and Building models) aspects

As described in Task 3.2.4, the Group 3 grantee, EPRI, played the role of a utility issuing transactive signals using the OpenADR 2.0b messaging protocol standard. In this standard, all such signals are carried via a DistributeEvent message that contains one or more event descriptions, each of which contains any number of discrete intervals. Each interval is associated with an electricity price[7]. To support the OpenADR protocol in our system, we integrated its defining information model into the SAIM.

3.3.2.1.1 Acquire OpenADR, Version 2B, in OWL

The information model for OpenADR 2.0b is available in XML schema form as a collection of .xsd files. We transformed these files into an OWL file encoded in Turtle format (.ttl) using the Semantic XML capability of TopBraid Composer. This transformation preserves the xsd-specific aspects so that conforming XML data can be mapped to the OpenADR OWL model, and vice-versa.

---

[7] On the price event side the value is price, in $/KW, but on the forecast side the intervals contain usage forecasts, in KW.

3.3.2.1.2 Integrate the FSGIM energy model with the OpenADR model

The next step was to integrate the OpenADR OWL model with the SAIM, a large part of which is built on the Facility Smart Grid Information Model (FSGIM, or ASHRAE Standard 201). The FSGIM already contains many of the OpenADR concepts such as intervals, interval payloads, and events. By defining many of our EPIC classes as subclasses of both the OpenADR classes and the FSGIM classes, it was relatively straightforward to relate the required OpenADR concepts to our native ones. Additional "helper classes" were defined that were needed solely to support the OpenADR message structure, but do not describe the energy usage attributes.

3.3.2.1.3 Develop conversion from XML to OWL for OpenADR

Using the SPINMap[8] capability in TopBraid Composer[9], we were able to dynamically map incoming OpenADR XML messages into RDF triples consistent with the SAIM and store them in our triple store. This SPINMap mapping process is extremely powerful, and uses declarative rules in conjunction with a general reasoning engine to accomplish the transformations.

3.3.2.1.4 Acquire sample pricing message in OpenADR

With the OpenADR model integrated with the SAIM and the mappings from OpenADR to EPIC defined, we were then ready to accept incoming messages. This was done by developing a web service (insertOadrMessagesDupCheck:XMLInsert) to be called by our EPIC executive system. When EPIC receives an OpenADR message (how it does this is described in the Price Messaging System section), it calls this service, which is implemented in a scripting language called SPARQLMotion[10].

3.3.2.1.5 Develop insertion of OpenADR model data to the SAIM

The web service introduced above transforms the XML data to OpenADR triples, maps them to triples consistent with our SAIM, stores those triples in our triple store[11], and returns key pieces of information to the calling routine so that the data, or even a complete reconstruction of the original XML message, can be retrieved as needed. The power of this approach is that incoming data in any of a variety of formats can be mapped to our neutral SAIM model, and later that data can be regenerated in any other desired format, including of course the original (incoming) form.

3.3.2.1.6 Retrieve OWL OpenADR from SAIM

---

[8] SPIN Map is an ontology mapping tool which is part of the TopBraid Composer, Maestro Edition, semantic development tool suite by TopQuadrant.

[9] TopBraid Composer is a semantic development tool by TopQuadrant.

[10] SPARQL Motion is a semantic application workflow tool which is part of the TopBraid Composer, Maestro Edition, semantic development suite by TopQuadrant.

[11] A semantic database based on RDF triples and supporting SPARQL and SPARQL reasoners.

Retrieval of the price message data basically performs the above sequence in reverse. A web service was developed (st:selectEventWithTag) that takes a unique retrieval tag as an input argument, executes a query against our triple store, then maps the results into the desired format, in this case a new XML message. Typically, XML encoding was used for messages to and from the XBOS buildings, while JSON was generated by our web services for use internally among the EPIC modules.

3.3.2.2 Platform aspects

The EPIC platform provides the networking and messaging components to convey the transactive signal from the utility to the semantic information store, and then on to the messaging system and the building management system. Please see the Price Messaging System section for details.

3.3.3 Negotiation Mediator use cases

Two primary use cases are implemented by the EPIC platform:

- Round trip interaction of pricing from utilities and forecasts from buildings. This allows for the utility to modify prices and see what the building forecasts will be, and thus to begin a new cycle until the forecasts meet the available supply.
- Mediation of prices for 'campus' buildings. This allows management of buildings that collectively act as a single VEN and replicates the round trip seen in the first use case but at the EPIC platform/building level.

A set of sub use cases based on mediation are as follows:

- Mediation based solely on forecast information and the collective tariff model
- Mediation based on actual use (building meter data) and the collective tariff model
- Mediation based on environmental sensor (e.g., temperature sensors) data, actual use (building meter data), and the collective tariff model

Although many other use cases can be imagined these were deemed a representative set for the purposes of demonstrating mediation capabilities of the EPIC platform.

3.4 Develop applications

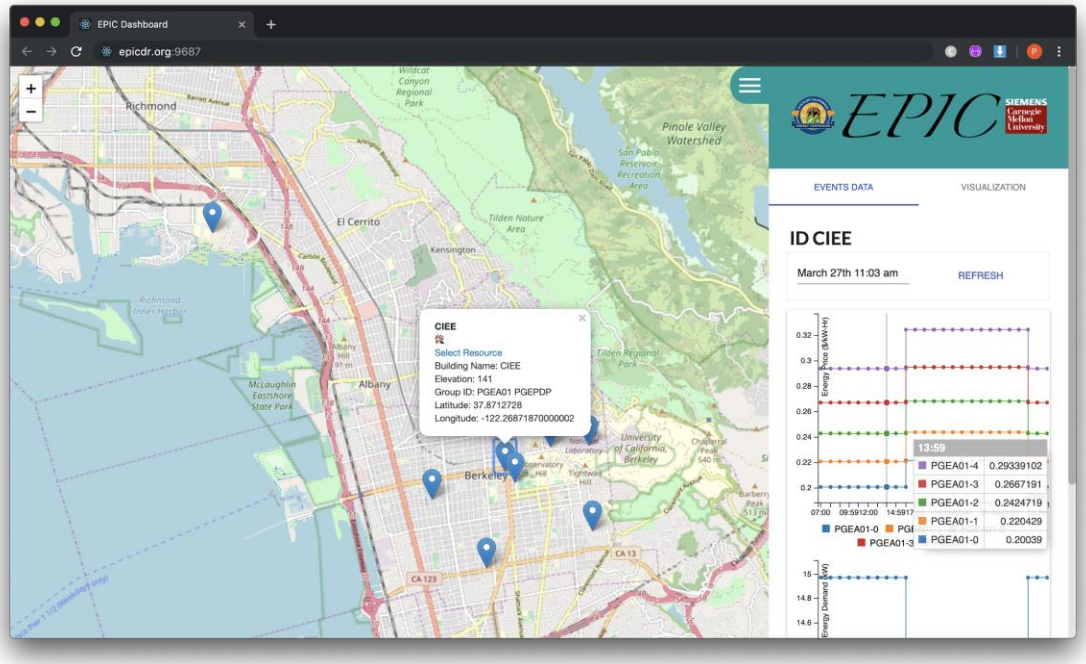
The EPIC platform brings together interactions between the utility and buildings. As a platform, EPIC manages many services that can be deployed to different servers. It is important, for debugging and general monitoring, to know what is going on in the platform at any given time. At the same time, the EPIC platform provides a view of all of the buildings in the pilot study and how they are reacting to the pricing events. As such, having a map-based dashboard that supports all of the buildings and their interactions with the utility can be a useful tool.

Two applications were developed to aid in debugging the EPIC platform and visualizing both workflow and the pilot study. Both are integrated into the same dashboard:

- Messaging visualizer
- Pilot buildings on map / dashboard

3.4.1 Overall DR dashboard

Figure 28: Demand Response dashboard.



The Demand Response dashboard show the price and energy demand from multiple buildings in the DR program shown on the map on the left.

Credit: Jack Hodges and Steve Ray

The DR dashboard is developed to assist in debugging the EPIC platform and visualizing the workflow. It is a stand alone single page web app that communicates with the EPIC platform using various REST API calls.

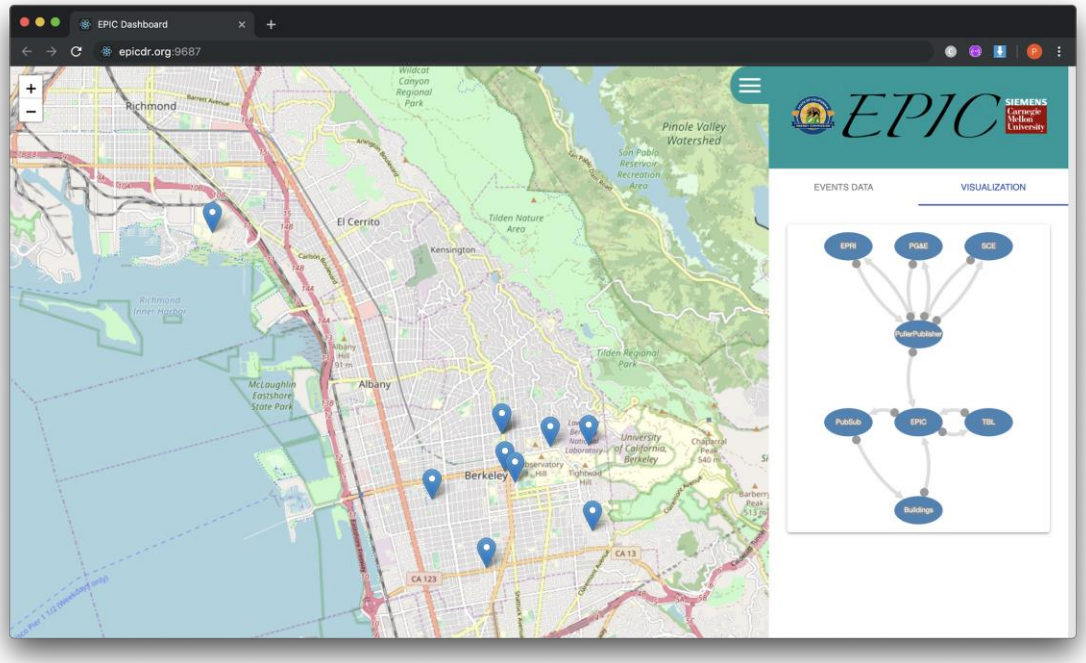
The DR Dashboard has a client application as well as a server application. The client application is built on top of ReactJs, along with a group of sister libraries such as LeafletJs for Map visualizations, cytoscape.js for graphs and chart visualizations. The application is an offline first Progressive Web App, designed to be Reliable, Fast and Engaging. The offline/cache first behavior is provided by using service workers and local storage at the client side.

The server application is written in python on top of the flask framework. It serves mostly as a middle layer between the client application and the EPIC Server api. The reason we have a middle layer is for security reasons. The EPIC platform authenticates incoming requests using SSL Certificates, and it is not safe to send SSL certificates and key pairs directly from the single page web application. Hence we use a python server that acts as an authenticating middle layer between our client and EPIC API Server.

Task 3.4.1.1 The messaging visualizer

The messaging visualizer is responsible for displaying the communication between different modules of the EPIC platform, and was designed to provide visual evidence that the messaging workflow is working properly. Every time a message is sent from one module to another (e.g., from EPRI to the Puller Publisher), a visualization signal containing the sender, receiver, event, and the associated building or group resourceID will be published onto the Rest Hooks module using the “VISUALIZE” topic. The messaging visualizer, which is subscribed to that topic, will receive the messaging signal. These signals will drive the client application of the DR Dashboard to show the flow of information. The visualization appears in near real time. Using the messaging visualizer, one can troubleshoot the system at the high level and visualize where the message may have been stuck.

Figure 29: Flow of price signal.



The diagram shows the flow of data.

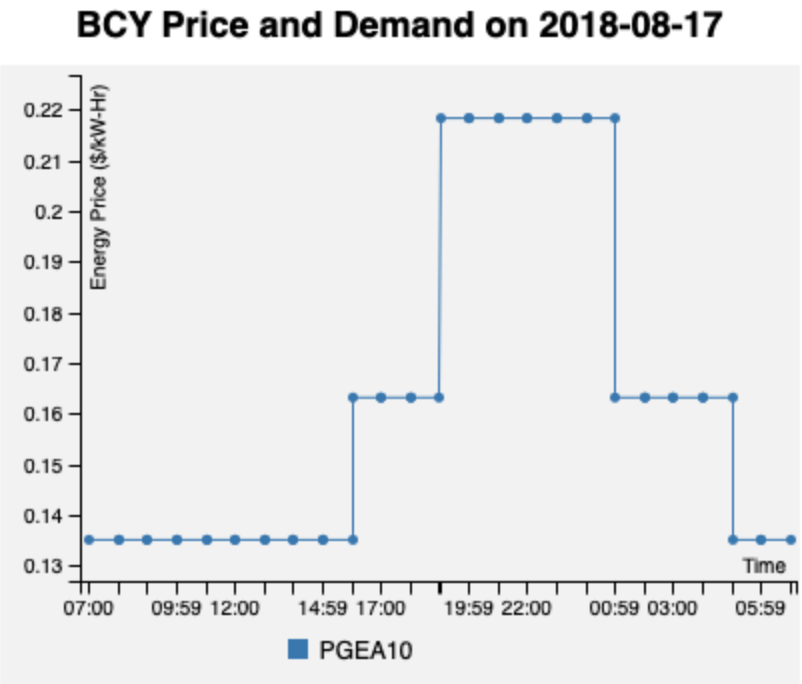
Credit: Jack Hodges

Task 3.4.1.2 The pilot building simulator and dashboard

The building dashboard is responsible for visualizing the price and energy demand in the EPIC system. The price and demand graphs are generated for particular dates and times that are

selected via the date/time selector in the epic dashboard. The visualization provides data about energy price and demand over a time range.

Figure 30: Price data from a PG&E tariff.



The diagram shows the change in price over a non-event day..

Credit: Jack Hodges

3.5 Making the software tools robust and secure

3.5.1 Conduct an overall security audit and verification of SSL management

The EPIC platform has a single endpoint to communicate with the utility companies and the XBOS servers. This endpoint is protected by the SSL encryption. In order to successfully establish a connection, the client will have to provide a valid signed certificate and the corresponding RSA key.

3.5.2 Develop tools for integration with XBOS

The integration with the XBOS involves two-way communications: the EPIC platform is able to send the Price Signal to XBOS, and XBOS is able to return with the corresponding forecast of the Energy Demand. Conventional POST/GET methods are not scalable for two reasons: (1) because the roundtrip can take a long time, and (2) the platform will have to individually manage connections to each of the buildings. Therefore, a Publish-Subscribe pattern was adopted to support this mechanism.

When the EPIC Platform receives the pricing payload from the VTN through the EPIC endpoint, it ingests the pricing information into the semantic knowledge graph and then publishes the event to the Rest Hooks module using the "PRICE-BAS" topic. The XBOS server, which has subscribed to the Rest Hooks module with the same topic of interest, will receive pricing information when they become available.

After receiving the pricing event, the XBOS server then publishes a forecast payload through the same EPIC endpoint. The forecast information will again be ingested into the knowledge graph and then published to the Rest Hooks module using the "DEMAND-UTILITY" topic. Any authorized party interested in (i.e., subscribes to) this topic will be able to receive the new forecast from XBOS.

For convenience, the subscription request was also built into the same EPIC endpoint. Authorized clients can subscribe to the Rest Hooks module with their receiving URL's and topics of their interest through the endpoint.

Another functionality of the EPIC platform is to get and store the building telemetry data, used for more effective and customized negotiations. So a pipeline is developed to poll the XBOS server for the building data. The received data is converted into an XML structure, which is then be ingested by the knowledge graph.

3.5.3 Develop deployment and runtime support tools

The EPIC platform comprises 2 separate but interacting components:

- Software modules, written in Java, Javascript, and Python
- Semantic models, written in OWL/RDFS and serialized in the Turtle format

Deployment tools for each of these were developed and are described below.

3.5.3.1 Platform deployment and runtime support tools

3.5.3.1.1 The staging, deployment, and runtime architecture

Details can be found in the Technical Specification document. A summary is given below.

3.5.3.1.1.1 Staging environment

The Staging environment contains all the individual modules of the EPIC platform, including the EPIC server, Rest Hooks service, and the PullerPublisher. Each of these modules are developed independently and thus has their own staging versions. This Staging environment keeps all versions of these modules so the EPIC platform could always be rolled back to a set of any combinations of these module versions. There are Bash scripts that automatically ship a set of modules to the Deployment environment.

3.5.3.1.1.2 Deployment environment

The Deployment environment keeps all the deployment versions of the EPIC platform. Each deployment is a set of any combination of the modules staging versions. This environment allows the administrator to roll back to any previous version of the EPIC platform if needed. There are Bash scripts to help deploy the entire EPIC platform of a certain version and also verify its runtime status on the cloud instance.

3.5.3.1.1.3 Runtime environment

The Runtime environment stores only the version of the EPIC platform that is running as well as any auxiliary files that the runtime generates. The service exists under the system level which is managed by the Systemd service in the Ubuntu instance. The Systemd service manager ensures the EPIC platform starts upon reboot of the instance and also keeps the platform alive at all times (i.e., restart after crash).

3.5.3.2 Semantic model deployment and runtime tools

The semantic models are deployed to the TopBraid Live environment running on the Amazon instance from versions running on local workstations in the TopBraid Composer development environment. As with all the EPIC code, the source code is maintained in a git repository. There are three project components to the semantic deployment:

- SemanticModels - where all the ontologies, declarative rules, SPARQLMotion scripts (web services), SPIN functions, and "static" building and tariff data are stored
- SemanticVocab - the triplestore where all the mapped event data exists, both pricing and demand forecasts
- SemanticArchive - a secondary triplestore that holds older data from SemanticVocab, for performance purposes

To deploy a new version of any semantic models or code, a local backup of the SemanticModels project is created that is then pushed to the Amazon instance using a TopBraid upload utility. This can be done "on the fly," without stopping and restarting the TBL executable.

3.5.4 Provide a User Manual and an Administration Manual

Three documents have been produced to support the EPIC platform:

- EPIC Technical Specification: Presents the project architecture, rationale, use cases, and requirements that have driven the platform implementation.
- System Administrators Manual: Presents the platform management
- Users Guide: Presents the mechanisms for interacting with the EPIC platform

3.5.5 Provide Tutorials

3.5.6 Package system for distribution

Code development was managed using git, therefore github has been used for distributing the code-set, along with the supporting documents. It should be noted that while the semantic model code is written in OWL, the TopBraid tool sold by TopQuadrant is used as the execution environment. It would take significant work to port the OWL source code to run in another

semantic environment with the same functionality (web services, etc.). However, the integrated ontologies themselves could be viewed/edited in any mainstream ontology editor.

3.6 Pilot Test in small/large commercial buildings

We have supported the pilot test by working with PG&E, Southern California Edison, and EPRI to provide test and live pricing information to the XBOS buildings, to respond to demand (PDP) events, and to receive and interpret demand forecasts from buildings. See our discussion of the Negotiation Mediator and Price Messaging System for details.

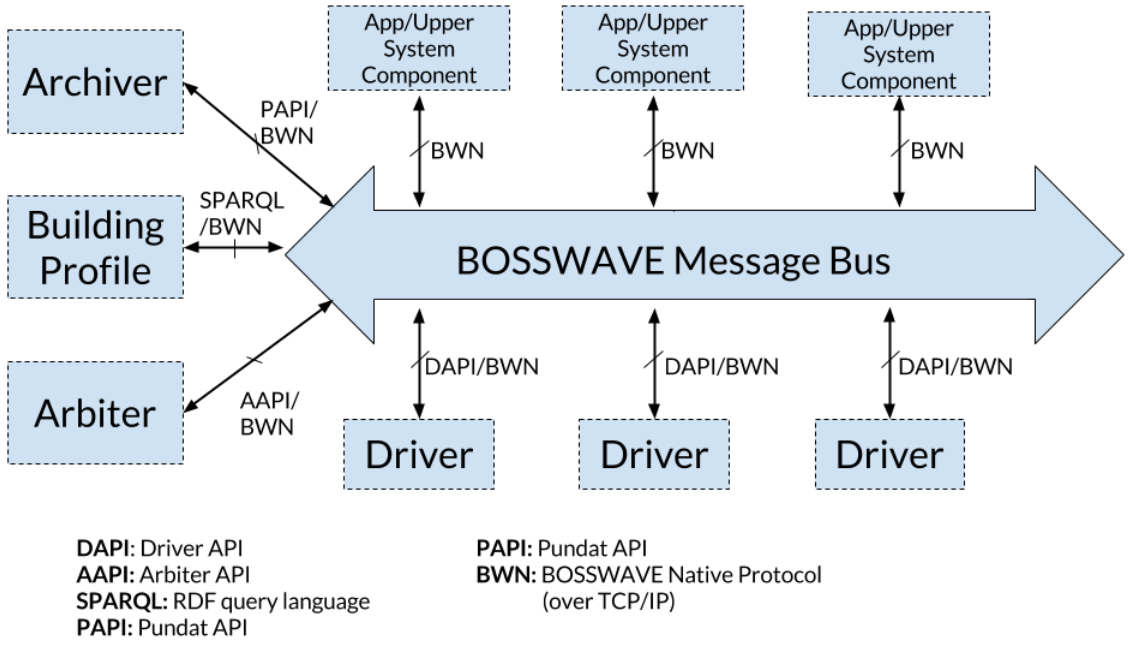
APPENDIX C:
XBOS-DR Platform Development

4.1 XBOS Overview

The eXtensible Building Operating System (XBOS) platform is an open-source, secure, distributed operating system realized on top of a family of technologies developed by the Software Defined Buildings[12] (SDB) group at UC Berkeley. XBOS-DR is an extension of the XBOS platform with a focus on integrating building management with demand response capability.

Because XBOS was designed to be expanded, these extensions can be implemented entirely within the XBOS architecture. As such, first is described the high-level architecture of XBOS before exploring each of its components, the interfaces between them, and finally how the required components of XBOS-DR can be implemented within this framework. Figure 5 provides a high level view of the overall XBOS system architecture that integrates building characteristics with individual control systems.

Figure 31: High level depiction of XBOS system architecture.



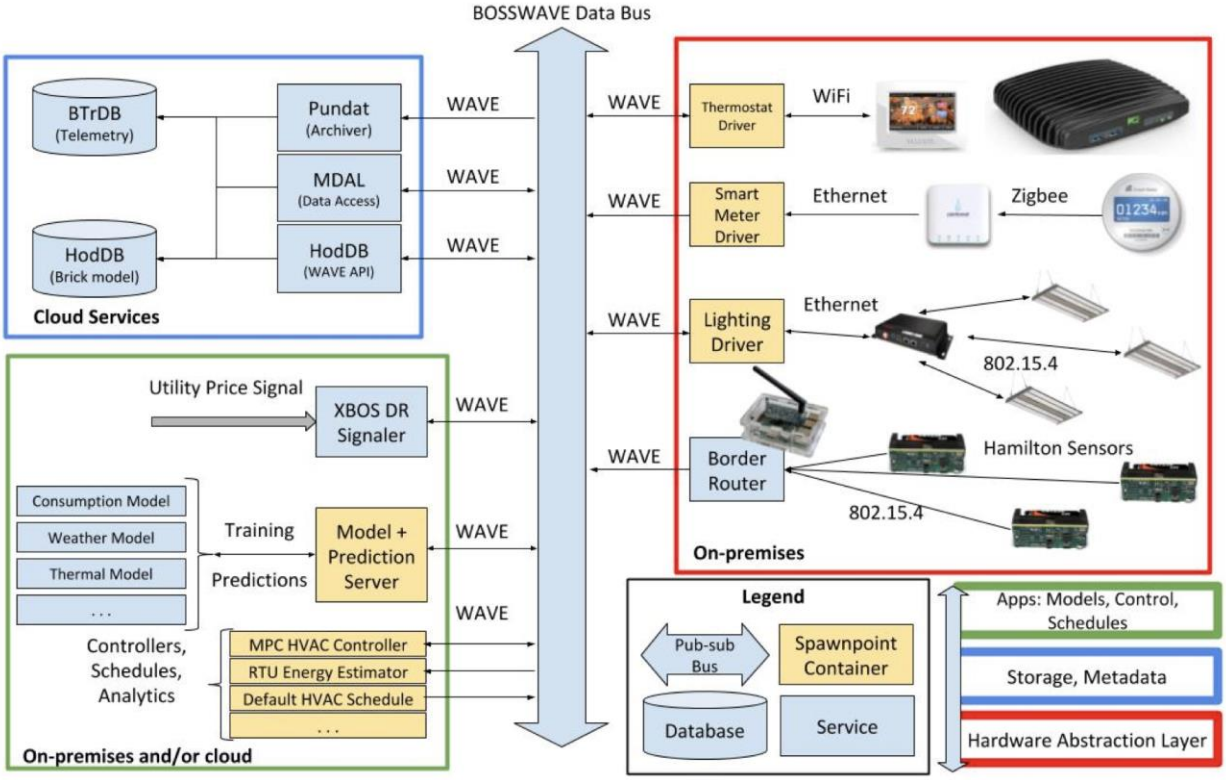
The diagram shows the components of XBOS.

Credit: Gabe Fierro.

[12] http://sdb.cs.berkeley.edu/

The figure below shows an overview of the software architecture: hardware abstraction, data management (storage/metadata), applications (price ingestion, models, controls, schedules). The architecture of the XBOS-DR platform: a secure data bus (Wave) is the underlying backbone of the platform. The upper left shows the databases for time series data as well as the metadata. The upper right shows the various hardware components, such as the thermostats, interval meter gateway, lighting controllers, and indoor environmental sensors. The lower left shows the transactive price signal, the model and prediction server, and various control schemes.

Figure 32: Schematic of XBOS Platform.



The diagram shows the components of XBOS.

Credit: Gabe Fierro.

At the core of XBOS is a secure, distributed *publish-subscribe* message bus called BOSSWAVE (Building Operating System Services Wide Area Verified Exchange)[13] [14]. BOSSWAVE offers a large number of features germane to the implementation of a secure, distributed system, but three are especially relevant for this high-level discussion:

[13] https://github.com/immesys/bw2

[14] http://www.sciencedirect.com/science/article/pii/S1084804516302521

- BOSSWAVE's pub-sub communication model decouples the producers and consumers of data, which permits the scaling of popular data sources and facilitates the discovery of distributed resources.
- BOSSWAVE integrates a strong notion of identity with a fine-grained permission model; data, services, applications and devices can only interact if allowed.
- BOSSWAVE allows distributed administration of these entities; rather than a single bottle-necked individual managing permissions for all entities, administration and auditing roles can be designated to other individuals in a hierarchical manner.

As opposed to a more traditional *point-to-point* pattern for interaction, XBOS uses the *publish-subscribe* (or *pub-sub*) pattern, which dictates that messages are not sent directly to specific receivers/subscribers (nor are they received from specific senders/publishers). Instead, messages are sent to an intermediary called a *broker*. Publishers describe each message with an identifying topic when sending to the broker. Subscribers tell the broker the topics they are interested in, and the broker forwards the relevant messages to the subscribers as they are received at the broker.

4.1.1 XBOS Components

There are seven primary components that make up the XBOS platform:

1. The **BOSSWAVE Message Bus** is a distributed message bus consisting at the core of a mesh of routers that forward traffic among interested parties, and at the edge of a set of application-level gateways called agents which provide a programmatic interface for interactions with resources exposed over BOSSWAVE.
2. **Drivers** interface directly with the set of devices and external APIs in an XBOS deployment and provide a uniform protocol for reading from and writing to these resources.
3. The **Archiver** service archives time series data published on BOSSWAVE and provides an API for retrieving historical time series data. Time series storage is provided by the BtrDB time series database, developed at UC Berkeley15 16.
4. The **Building Profile** service stores a query-able virtual representation of a building, its subsystems, and how these are interconnected (e.g. HVAC, lighting, electrical and spatial subsystems). This representation follows the recently developed Brick metadata schema for buildings[17], which enables the writing of portable building applications.
5. The **Arbiter** service performs conflict management and maintains invariants for the set of actuatable (writable) resources in XBOS on behalf of the applications, controllers and other processes running in an XBOS instance.

---

[15] http://btrdb.io/

[16] https://www.usenix.org/system/files/conference/fast16/fast16-papers-andersen.pdf

[17] http://brickschema.org/

6. **Applications** and upper system components are processes that leverage the set of distributed resources in an XBOS instance. These processes include—but are not limited to—visualizations, model training and development, analytics, dashboards and controllers.
7. **Spawnpoint** (not pictured above) is a tool for reproducibly deploying, running and monitoring distributed, containerized processes over BOSSWAVE.

For each of these components, we provide a more detailed figure of their construction and link to the code and APIs involved in Appendix B.

4.1.2 Communication

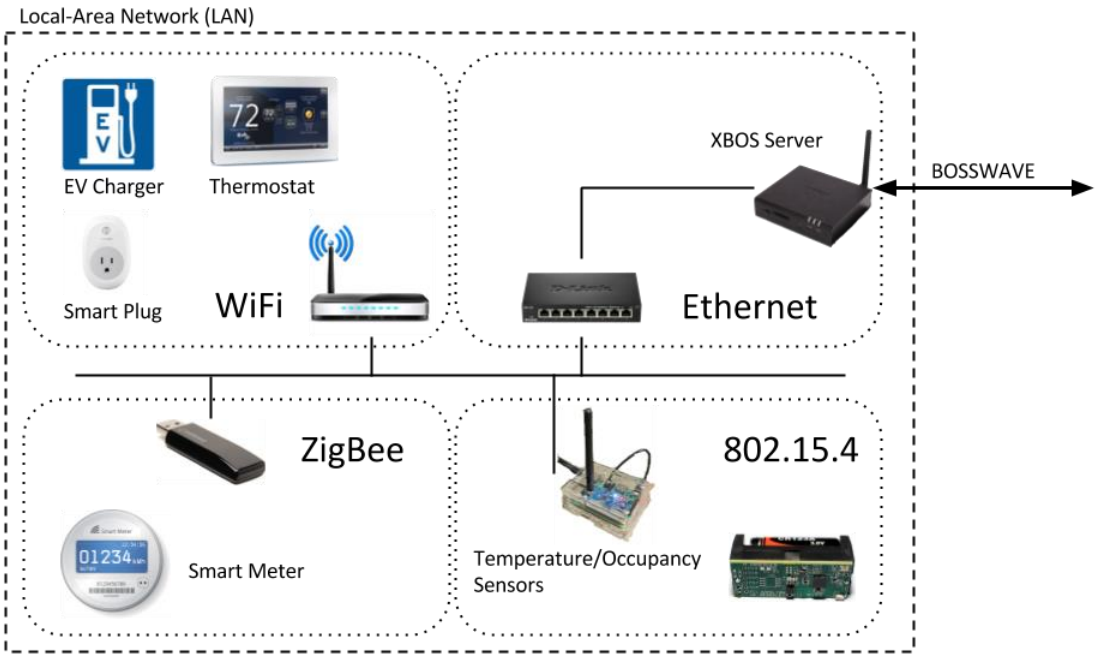
Implementing XBOS-DR with XBOS

The XBOS architecture is extensible in that it allows for the integration of additional services, controllers, drivers and other components that may provide functionality beyond what is already contained in XBOS. For XBOS-DR, these additional components involve integration with the utility grid, buildings, building automation systems, sensors, thermostats, and other devices such as electric vehicles.

Communication

There are two considerations for communication in the XBOS-DR system: the physical medium (how to talk to a device or service) and the application details (what the device/service says and how to talk to it).

Figure 33: Schematic of XBOS Communication.



The diagram shows the communication components of XBOS.

Credit: Gabe Fierro.

This diagram illustrates the high-level network architecture of an XBOS-DR deployment at a small commercial building. All devices in an XBOS-DR will exist on this network, which removes the restriction that all devices need to use the same physical layer (i.e. only ZigBee devices).

Various hardware will use WiFi/gateways to communicate to the XBOS-DR system. Data from the interval meter will come from the Rainforest Eagle gateway plugged into the miniature computer that communicates via Zigbee to the smart meter. The thermostat will use WiFi or a proprietary gateway. Deployed temperature, occupancy, illumination, CO2 and other sensors will communicate on an 802.15.4 network, using a Raspberry-Pi border router to integrate with the XBOS-DR LAN.

There are several modes of communication common to the types of devices and services XBOS-DR will integrate with. Integration will take place in an XBOS-DR driver. The construction of XBOS-DR drivers is general enough that all of these patterns can be implemented.

POSTing to API endpoints: to read from or write to a device, the driver must send HTTP GET or POST requests. This mode is common for consumer-grade "smart" devices such as thermostats and plug load controllers.

Polling interfaces: rather than provide an API for accessing information, some devices and services will push recent and historical data to a website where it can be periodically scraped by retrieving and parsing the HTML at a configurable interval. This is common for retrieving weather data, calendar data and energy usage data.

Client protocol: to interact over protocols such as OpenADR 2.0 and OCPP, the client needs to implement specialized parsing and operational logic.

For each mode of communication with the devices and services involved in an XBOS-DR deployment, the role of the XBOS-DR driver is to maintain correct operation with the underlying resource while providing a simpler API to the rest of XBOS-DR. The complete design of these APIs will be determined as the project advances, but the following design points are relevant:

- XBOS-DR driver APIs will consist of binary BOSSWAVE messages published on well-known URI topics
- The contents ("payload") of these messages will be encoded using MsgPack, an efficient binary serialization protocol similar to a typed JSON
- Drivers will report state by publishing BOSSWAVE messages either when state changes or at a configurable interval
- To receive actuation requests, drivers will listen on a set of well-known URIs for messages containing the action to be taken and the set of parameters for that action.

4.1.3 Security

Security considerations in XBOS-DR take the form of three concerns: authentication, authorization, and minimizing the attack surface.

Authentication is the identification of a person or entity. Where possible, all applications, services, drivers and users will be identified by a unique BOSSWAVE public/private key pair.

This allows XBOS-DR to potentially use BOSSWAVE's authorization capabilities across all entities involved, including the building owner, the device manufacturer, the building occupants, maintenance crew, and the grid operator.

Authorization is the process of verifying and enforcing what actions an authenticated entity is allowed to perform. BOSSWAVE's authorization model enables the granting, revocation and auditing of these permissions in terms of which entities can perform an action ("who") and in what segments of time that permission is valid ("when"). This fine-grained administration of permissions covers the authorization of access to XBOS-DR services, applications, drivers and data.

The final concern is how to minimize the attack surface of XBOS-DR. BOSSWAVE's approaches towards authentication and authorization cover security within an XBOS-DR deployment and provide some measures (such as DOS protection) against external threats. However, in a system characterized by the integration of many "black box" 3rd-party services and devices, a real challenge is how to prevent these from becoming backdoors into an XBOS-DR deployment. For example, a UDP hole punching webcam can let an attacker gain access to the local-area network, bypassing any firewalls or BOSSWAVE security measures deployed at the network's gateway. Each device and service integrated into XBOS-DR should be audited to see what information it sends and receives.

4.2 Hardware Abstraction Layer

This Subsection describes various devices and applications considered for incorporation into the XBOS-DR platform. Define hardware abstraction.

In 2013, the UC Berkeley research team implemented an earlier prototype of the platform in a small commercial building. The team learned a few lessons from this experience. Not all connected/networked devices have an easy-to-use API. Some devices did not have the functionality we wanted (e.g., ability to define heating and cooling setpoints at the same time). Some devices tended to disconnect from the network and not reconnect very easily (RTA). One Ethernet connected device (Prolyphix) seemed to lose the current time/day.

Regarding HVAC functionality, networked thermostats are a fairly inexpensive way to integrate multiple roof top units common to small commercial buildings. The research team has successfully used the Pelican and Venstar connected thermostats. QuEST has been using Pelican, so the team will use the Pelican thermostats. For large commercial buildings, the team has successfully used python coded scripts to actuate Building Automation Systems using BACnet through the pybacnet library with the Siemens Apogee System. The team will continue to explore other BAS types. The team has explored alternatives to pybacnet, such as BACpypes (http://bacpypes.sourceforge.net/), and determined that BACpypes is the best interface to use.

Controlling lighting systems is less straightforward. Ideally, the research team would work with state of the art LED lighting and controls, but these retrofits are expensive and not within the budget of the project. The team will opportunistically recruit a few buildings with recent

retrofits with control systems that have open APIs or will let the team access to the APIs. In the past Enlighted has given the team access, so the team will pursue that vendor. In addition, Daintree presumably has a fairly open API. Conventional light switches may be replaced by several vendors; EnOcean has several light switch products made by various manufacturers (WattStopper, Leviton) that use the EnOcean proprietary 902 MHz protocol.

The research team has experience and contacts with a few plug load controllers, namely Enmetric smart plugstrips. Past experience showed the team that many common plug load controllers were flakey, often disconnecting from the network. The team will test several products.

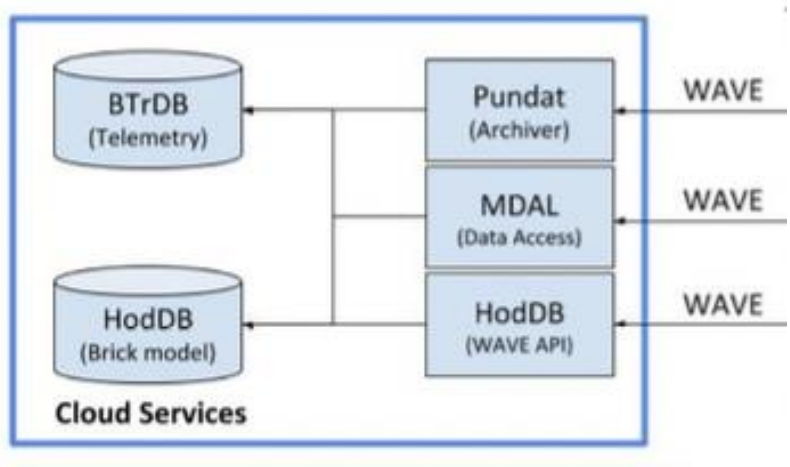
The research team has used the Rainforest Eagle as a gateway to the smart utility meter; the less expensive Rainforest RAVEn may be an alternative.

Regarding auxiliary wireless sensors, such as temperature, occupancy, humidity, light, and carbon dioxide, the team will explore several options, including one platform (Hamilton) built xby current graduate student.

4.3 Data Management Services

Various papers describe the data management services in detail, see Figure below. The main storage database, BTrDB (http://btrdb.io/) is a high performance database developed at UC Berkeley. Metadata management—the tagging of time-series data—is handled using the Brick metadata schema (https://brickschema.org/), which uses several tools. HodDB (hoddb.org) is a specialized RDF database and SPARQL query processor for Brick, developed at UC Berkeley. It stores models of buildings and serves queries on those models. MDAL Mesh Data Abstraction Library (MDAL) is a C++ library used for data access. Pundat is a data archiver and metadata query processor for BOSSWAVE, the authentication/authorization broker, both developed at UC Berkeley.

Figure 34: Schematic of XBOS Data Management.



The diagram shows the data management services of XBOS.

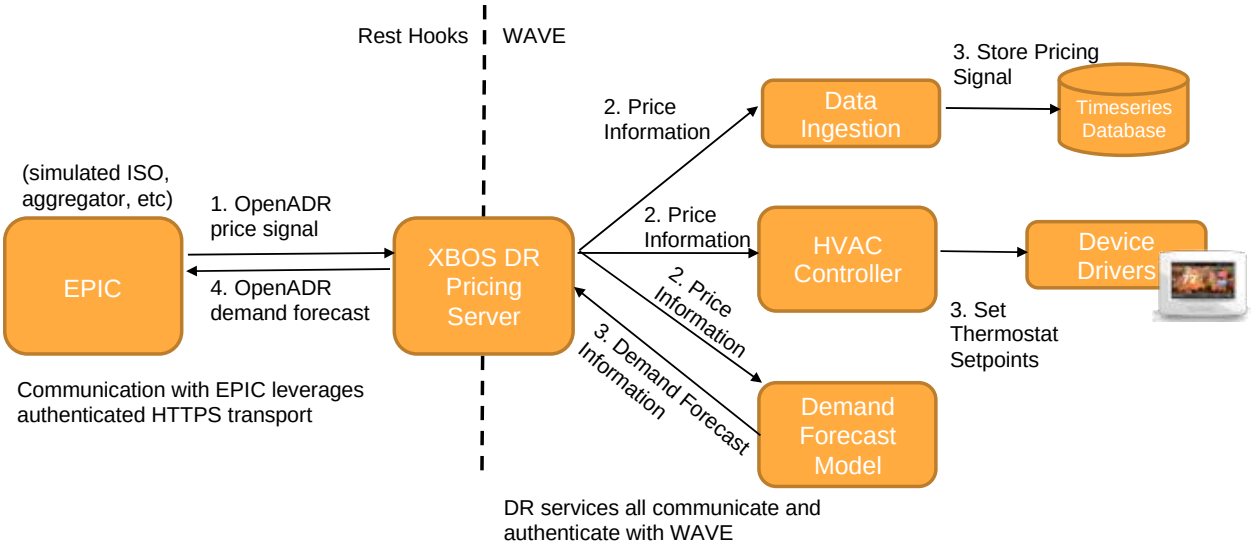
Credit: Gabe Fierro.

Over the course of the project, the team found issues with the BOSSWAVE security implementation, especially in cases where the Internet connection was intermittent. The team modified BOSSWAVE to WAVE and WAVEMQ. WAVEMQ provides a secure authenticated publish-subscribe data bus with fine-grained control over permissions.

4.4 Applications

4.4.1 XBOS Pricing

Figure 35: XBOS Pricing Architecture



As outlined in Chapter 3.1, the EPIC platform and services generates three different pricing signals, which are then passed to the XBOS to control the building HVAC.

Figure 35 depicts the architecture of the XBOS pricing services and the flow of operation. The EPIC platform publishes a pricing signal for each tariff to the XBOS-DR Pricing Server using the openADR format. The XBOS-DR server extracts the pricing information from these tariffs and publishes this information using WAVE to the corresponding topics for each tariff. The data ingestion service, which is subscribed to all of the pricing topics, then stores the pricing information in the timeseries database for each of the tariffs. Similarly, the HVAC controller for each building, which is subscribed only to the tariff topic associated with that building, uses these prices to generate the schedule and control the HVAC systems. The schedule setpoints are then further relayed to device drivers in the building. The drivers are responsible for enforcing the scheduled setpoints. Finally, a demand forecast service, which models the total building power consumption, uses the pricing information to predict the overall power consumption of each building. The predicted consumption is then published to the XBOS-DR Pricing Server. The Pricing Server uses the demand forecast to generate an openADR compatible signal with the predicted forecast, which is then published to the EPIC platform.

4.5 Hardening

XBOS hardening was an additional task to "test" the existing documentation for XBOS, identify gaps and fill in these gaps. Additionally, the software was also tested and any issues or bugs and improvements were raised as issues and pull requests on the Github repository of XBOS. XBOS hardening spanned across the two versions of XBOS - v1[1], [2] that used WAVE/BOSSWAVE[3] and v2[4] that is currently being supported and that uses WAVE/WAVEMQ[5]. As XBOS v2 is what would be used moving forward, the hardening effort was realigned to test the software and create the documentation for installation and deployment of the software.

XBOS v1

As part of testing out the documentation in XBOS v1, a machine that had XBOS installed was provided to a team, who did not have any previous hands-on experience with XBOS, along with a TP-Link smart plug. The task for this team was to follow the documentation and set up a driver for the smart plug and write a piece of software to get information from the plug and also to turn the plug on/off. Additional tasks involved using the python libraries developed as a part of XBOS v1 (mdal[6], dataclient[7] and pymortar[8]) to query data using BRICK queries and also to create analytical applications that used the stored data. An output of these tasks were improvements to the driver deployment script and also minor updates to the documentation and also various new applications that queried meter and thermostat data to build energy baselines and demand response event baselines.

XBOS v2

One of the main improvements in XBOS v2 was the ability of a site to continue operation even during periods without Internet connectivity. It also used gRPCs[9] for message encoding and serialization and this enabled device drivers and services to be written in multiple languages. XBOS v1 only used BTrDB as the timeseries data store, where as in XBOS v2, a user had the option to use either BTrDB[10] or InfluxDB[11]. During the hardening tasks, all these new features were tested:

1. Installation and deployment of XBOS v2 in a site
2. Continued operation of XBOS drivers and services in the site during network outages
3. Use of protobufs to define new messages to be sent across the WAVEMQ message bus
4. Driver development for new devices and services
5. Plugin development for new drivers so that the messages published on the WAVEMQ message bus can be ingested and stored in a InfluxDB database
6. Use of python libraries (pyxbos[12] and pymortar[8]) to query and retrieve data from the datastore

References

[1] XBOS v1: https://github.com/SoftwareDefinedBuildings/XBOS

[2] XBOS v1 docs: https://docs.xbos.io/

[3] BOSSWAVE: https://docs.xbos.io/bosswave.html

[4] XBOS v2: https://github.com/gtfierro/xboswave

[5] WAVEMQ: https://github.com/immesys/wavemq

[6] mdal: https://github.com/gtfierro/mdal/

[7] dataclient:https://github.com/gtfierro/mdal/tree/mdal-0.0.7

[8] pymortar: https://mortardata.org/

[9] gRPC: https://grpc.io/

[10] BTrDB: http://btrdb.io/

[11] InfluxDB: https://www.influxdata.com/

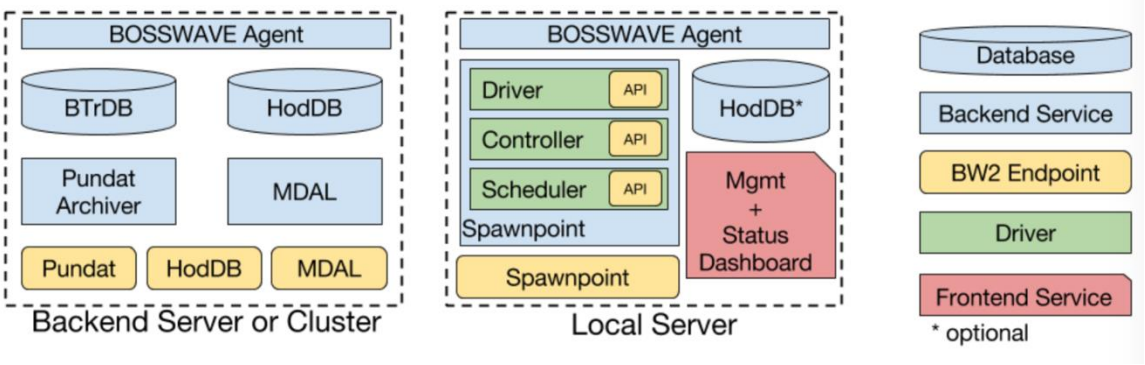
[12] pyxbos: https://pypi.org/project/pyxbos/

4.6 XBOS-DR Installation

XBOS follows a "microservice" architecture; that is, an instance of XBOS is composed of a set of distributed services connected by a secure bus (BOSSWAVE). As such, there are several ways to arrange these services. In this guide (https://docs.xbos.io/install_overview.html), we will cover one common architectural configuration.

In this configuration, there is a local on-premises server and a set of remote services.

Figure 36: Schematic of servers needed to host XBOS-DR.



The diagram shows the servers of XBOS.

Credit: Gabe Fierro.

Local Server

The local XBOS server has the following:

- **BOSSWAVE Agent:** a gateway to the global, secure pubsub communication plane

- **configuration:** a git repo containing driver and service configurations. This repo is not necessarily pushed to a remote server
- **Spawnpoint:** a local Spawnpoint daemon for running drivers requiring access to the local network
- **watchdogs:** a set of monitors for local processes to help determine the health of the deployment

Installation instructions for the local server are located at
https://docs.xbos.io/local_install.html

Core Services

There are several core services important for a full XBOS installation. These services can be run on a local node, but we recommend to run these on a more capable server or cluster.

These services are:

- BTrDB: a fast, scalable timeseries database for storing all telemetry
- Pundat: an archival/data ingester service for data published on BOSSWAVE
- HodDB: a database and query processor for accessing Brick models describing deployment sites
- MDAL: a data retrieval tool for fetching timeseries data using metadata stored in a Brick model

Thus the components involved in an XBOS installation at a site include the following steps:

Base BOSSWAVE

- install BOSSWAVE on a server
- create a local git repo for versioning configuration information
- configure namespace:
    - create a namespace key for the site
    - create an alias for the site
    - establish a designated router for the namespace

Process Management

- configure entity for `spawnd`
- adjust spawnd configuration and deploy spawnd:
    - entity
    - spawnpoint alias ("name")
    - base spawnpoint uri
    - router endpoint

Driver

- install wizard:
    - choose the device, then fill out the corresponding params file
    - create entity; give it the required permissions:
    - create the archive request; also associated with the device name.
- spawnpoint

APPENDIX D:
Pilot testing

5.1 Site Selection Criteria

Small commercial buildings include grocery stores, banks, retail, cafes/restaurants, offices, and others. Each business has different appliances, HVAC, or lighting loads, a different load profile over the course of the day, and different needs with respect to a business model. The site selection criteria helped choose the buildings/businesses that will best benefit from the XBOS-DR platform and guided the building recruitment.

General requirements:

- Must be a commercial building less than 50,000 sf.
- Must have electrical service provided by an investor-owned utility (SCE, PG&E, or SDG&E)
- Must have individual meter for the building or space under consideration
- Not multi-tenant
- Must pay for electricity bill
- No additional planned retrofits or renovations between now and December 2018.
- Must commit to participating in the 12-18 month project and agree to interaction with the building control systems.
- Must agree to building energy audit.
- Has at least 12 months of interval meter data available.

Desirable criteria:

- Existing electrical sub-metering in place (e.g., any additional zone or appliance power metering).

Table D-1: Site selection critieria.

Category	Minimum Requirement	Preferred Conditions	Source of data
Building Use	Building space is currently occupied, regularly used, and not expected to be vacant (except in the case of tenant turnover) from now until December 2018		Interview with building owner/tenant and utility
Change of Occupancy	No change in occupancy (tenant and use) in the last 12 months	No change in occupancy (tenant and use) in the last 24 months	Interview with building owner/tenant and utility
Planned energy retrofits	No additional energy efficiency retrofits planned from now until December 2018	No energy retrofit in the past 12 months (for baseline)	Interview with building owner/tenant and utility
Major renovations	No major renovations planned from now until December 2018	No major renovations in the past 12 months (for baseline)	Interview with building owner/tenant and utility
Internet	Must have reliable Internet service		Owner survey/email
Configuration	Must have accessible place to install XBOS server and peripherals (e.g., Ethernet port)	Single story, open plan areas, nice to have fairly central location for XBOS server	Owner survey/email
HVAC	Must have functioning air conditioning with thermostat with Common or C-wire (provides 24 vac power to thermostat from HVAC control board). Can have electric or heat pump heating (not necessary).	Air conditioning with more than one single zone HVAC units. Simple packaged rooftop units are ok. Would be nice to have at least one sophisticated system (one multi-stage, Variable Frequency Drive fans)	Review mechanical drawings OR owner survey/email. Building audit.

Category	Minimum Requirement	Preferred Conditions	Source of data
Lighting Equipment	Would be nice to control at least one bipole (simple on-off) switch	Building with recent retrofit of LED lights controlled by a controls system (e.g., Lutron, Enlighted, Daintree) with an open API[18]	Owner survey/email
Plugload Equipment	Would be nice to control at least one load (water cooler, vending machine, printer, etc)	Multiple plug loads controlled.	Owner survey/email or building audit.
Occupancy Sensors (from existing security system/ lighting system)		Would be nice to work with an established vendor that has occupancy sensors (e.g.: Lutron) with an open API.	Owner survey/email or building audit.

Source: Therese Peffer and Marco Pritoni

5.2 Recruit customers and buildings

Outreach was primarily conducted by QuEST through existing contacts and relationships with local government entities, regional energy efficiency bodies, and industry associations. These bodies participated in program presentations then broadcast the details to their staff, constituents, and membership. Interested customers received site visits, screening, and as appropriate, program services. Organizations included:

- **PG&E East Bay Energy Watch** (serving Alameda and Contra Costa County)
- **SJV Clean Energy Organization** (serving the entire San Joaquin Valley and High Desert Region of California)
- **California Green Business Network** (statewide membership of certified "green" small and large commercial entities)

[18] Application Programming Interface is "a set of routines, protocols, and tools for building software applications. An API specifies how software components should interact." www.webopedia.com/TERM/A/API.html

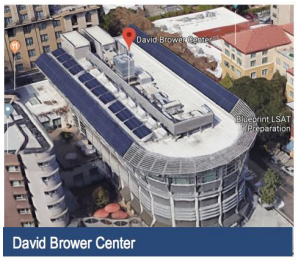
- **Municipal Implementation Team** (PG&E program providing energy efficiency technical support services to local government)
- **StopWaste.org** (Alameda County public agency)

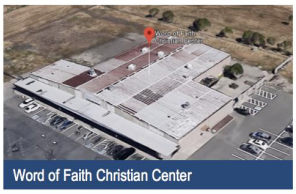
Recruiting materials can be found in Appendix D.

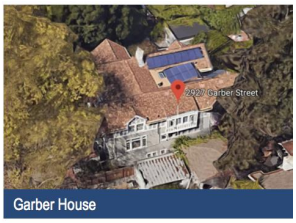
The recruitment process took place over several months, from April 2017 through Nov 2017. There were several stages, including a walk through and audit of the building. In many cases the building looked eligible at first, and after much back and forth discussion, it was determined that the meter was ineligible or some other issue prevented the eligibility.

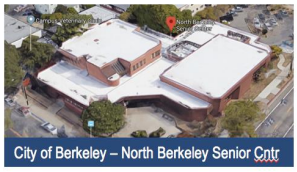
Figure below shows images of the buildings and Table 8 describes the buildings, mostly in the PG&E territory, but two from SCE.

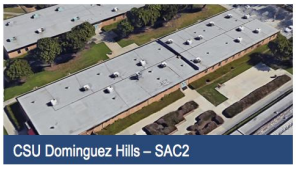
Figure 37: Images of the project buildings.











Photos of the buildings.

Credit: Greg Thomson and Google Earth

Table 7: Buildings selected for the project.

Site Name	Area	Controlled Sys.	Classification	Risk Cat.	Tariff	IOU	Climate Zone
CSU Dominguez Hills (SAC2)	15,548 SF	HVAC/LGHT	Business (Higher Ed., Offices/Classrooms)	III	Master Metered	SCE¥	8
Orinda Community Center	20,488 SF	HVAC	Multi-use assembly spaces (Theater, Meeting rooms)	III	HA10SX	PG&E¥	12
North Berkeley Senior Center	20,834 SF	HVAC	Senior center (banquet hall & kitchen)	III	HA10SX	PG&E	3
The Local Butcher Shop	2,850 SF	HVAC/REF	Mercantile (Commercial Mixed-Use)	II	HE19S	PG&E¥	3
Avenal: Animal Shelter	4,132 SF	HVAC	Animal Shelter (with storage)	II	HA1X	PG&E¥	13
Avenal: Movie Theatre	15,820 SF	HVAC	Assembly (Movie Theater, meeting rooms)	III	HE19S	PG&E¥	13
Avenal: Veterans Hall	8,683 SF	HVAC	Senior center (banquet hall & kitchen)	III	HA1X	PG&E¥	13
Avenal: Recreation Center	2,417 SF	HVAC	multi-use community center with IT training facility	II	HA1X	PG&E¥	13
Avenal: Public Works Department	12,700 SF	HVAC	Moderate Hazard Storage	I	HA1X	PG&E¥	13
Fire station 1, Hayward	8,700 SF	HVAC/EV	Business (with storage, kitchen, and sleeping areas)	IV	HA10SX	PG&E	3
Fire station 8, Hayward	6,500 SF	HVAC/PV	Business (with storage, kitchen, and sleeping areas)	IV	A6	PG&E	3
Berkeley Corporation Yard	9,600 SF	HVAC/PV	Business (offices)	II	A10SX	PG&E	3

Richmond Field Station, Bdg 190	1,850 SF	HVAC/EV	Business (Higher Ed., Offices/Classrooms)	II	Master Metered	PG&E¥	3
South Berkeley Senior Center	10,427 SF	HVAC	Senior center (banquet hall & kitchen)	III	HA1X	PG&E	3
Jesse Turner Fontana Community Center	43,193 SF	HVAC	Assembly (Banquet Hall, Indoor Gymnasium)	III	Master metered	SCE	10
CIEE	8,424 SF	HVAC/LGHT	Business (offices)	II	A1X	PG&E¥	3
LBNL building 90C	18,500	HVAC	Business (offices)	II	Master Metered	PG&E¥	3
Word of Faith Christian Center	19,733 SF	HVAC	House of Worship and Accessory School Spaces	III	HA1X	PG&E¥	12
Orinda Library	24,250 SF	HVAC	Library	III	HA10SX	PG&E	12

5.3 Obtain baseline data

In order to determine the effect of energy efficiency and demand response event measures, the team worked to obtain historical electrical energy consumption from each participant. Participating customers authorized QuEST to receive a daily interval data. The authorization was executed, electronically, through each customer's utility account management portal.

QuEST became a certified, third-party recipient of utility usage data and designed its system to communicate with utility Green Button or Share My Data API (see Appendix E).

On a daily basis, the utility sends QuEST notifications that new interval data is available for download. QuEST's system automatically responds with the appropriate access tokens and downloads the data.

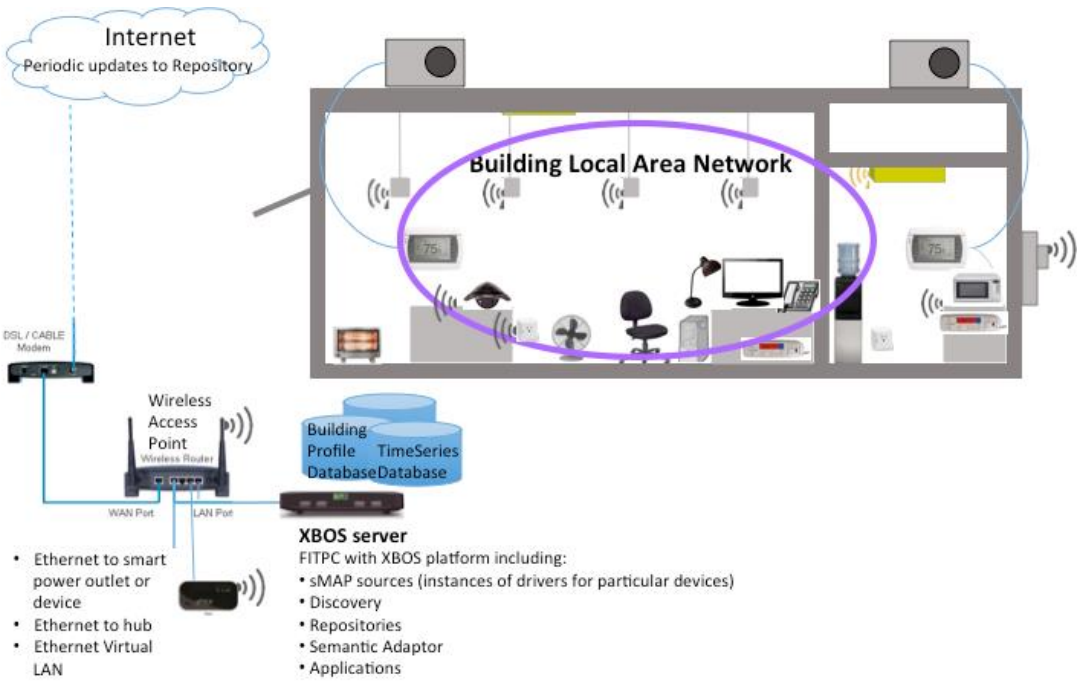
Downloaded data quality and consistency are not guaranteed, therefore QuEST runs data validation scripts to detect gaps and outliers. When corrective actions are implemented, the actions are logged and the data points are flagged.

QuEST delivered "validated" data to the XBOS server on a daily basis.

5.4 Install XBOS and components

This section describes the platform and devices, see Figure 16.

Figure 38: The XBOS-DR platform installed in a small commercial building.



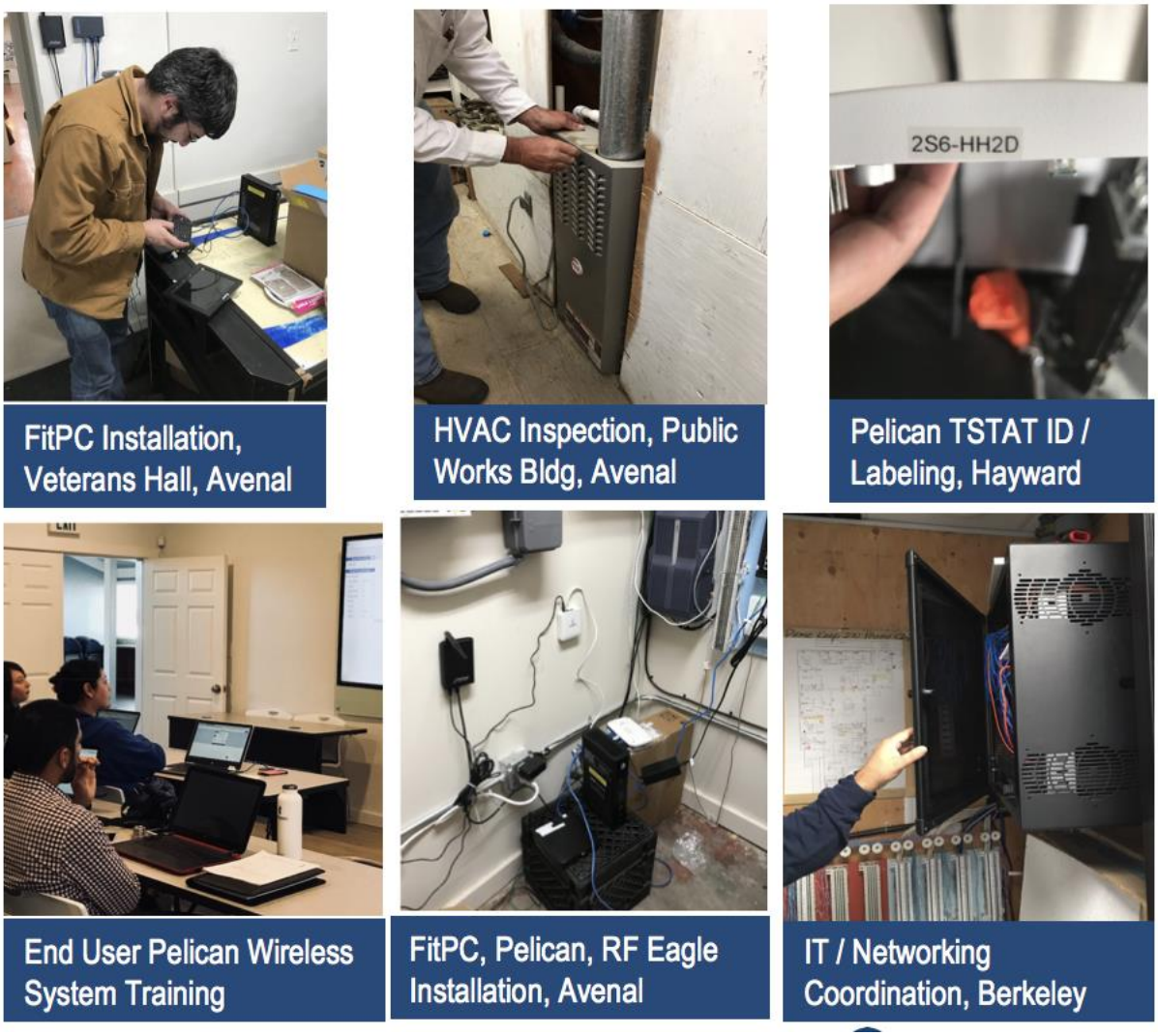
Schematic of XBOS in a small commercial building

Credit: Therese Peffer

The XBOS-DR platform consists of:

- A miniature **computer** (Fit PC) either connected to the building's Local Area Network or creating its own LAN via multiple gateways. Requires connection to the **Internet**. Likely will need an **Ethernet port**.
- A gateway to the whole building interval **meter**: either with the Rainforest Eagle gateway or TED current transducers on the main circuit breaker panel to the building.
- Connection (e.g., via WiFi) or gateway to **connected thermostats** (whether through the Pelican gateway or via WiFi to Venstar)
- Connection to **lighting**, either through a gateway (e.g., Enlighted systems for controlling LEDs) or replacing the manual bipole switch with a connected switch (e.g., EnOcean device) (See Appendix E for installation procedure)
- Connection to **plugload outlets** or plugstrip control, either through WiFi or a gateway
- Connection to sensors: some buildings will use Hamilton **sensors**, with temperature, occupancy (PIR) and some CO_2. The experiments run in these buildings may require more sensors than a commercial installation. These additional sensors can be utilized to "ground-truth" some of the measurements and predictions (e.g. occupancy) or to assess what sensors are required.

Figure 39: Photos of installation at different sites.



Installation and training images from the sites.

Credit: Irina Krishpinovich

5.5 Collect and monitor data

All data went to BTrDB and was accessible via a web-based plotting tool (Mr.Plotter) or through writing Python queries to access the data through HodDB.

Quality controlled data.

5.6 Commission devices and platform

The researchers encountered a number of issues (e.g., bad Internet in Avenal, Bad TED energy data from Jessie Turner, negative readings at RFS).

5.7 Conduct DR tests

Two main strategies: expand setpoints, force first stage, and MPC

The problem and resolution with price notification, and notice 5 days ahead

5.8 Conduct EE interventions

EE from installing Pelican thermostats

MPC in CIEE

Appendix E: Instructions for installing XBOS v2 in an Ubuntu 18.04 machine

Assumptions:

- Git is installed
- User name on the machine is user and the home directory is /home/user

To be noted:

- Please contact Anand Prakash <akprakash@lbl.gov> or Gabe Fierro <gtfierro@cs.berkeley.edu> if you run into issues during the installation.
- As XBOS is still in its infancy, the software is being updated. Hence, this installation guide can quickly become outdated.
- We will be releasing a docker image with this setup, and that would take care of a large portion of the setup and also would handle any updates to the software

Step 1: Set up the python environment using Anaconda package manager (make sure the right version is chosen)

```
$ mkdir Downloads
$ cd Downloads/
$ wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86_64.sh
$ bash Anaconda3-2019.03-Linux-x86_64.sh
$ vim ~/.bashrc
>> export PATH="/home/user/anaconda3/bin:$PATH"
Save and exit
$ source ~/.
```

Step 2: Install go (make sure the right version is chosen)

```
$ cd ~/Downloads/
$ wget https://dl.google.com/go/go1.12.4.linux-amd64.tar.gz
$ sudo tar -C /usr/local -xzf go1.12.4.linux-amd64.tar.gz
$ vim ~/.bashrc
>> export PATH=$PATH:/usr/local/go/bin
>> export GOPATH="/home/user/go"
```

This folder doesn’t exist below

```
>> export PATH=$PATH:/usr/user/go/bin
Save and Exit
$ source ~/.bashrc
```

Step 3: Install protobuf packages and compilers

Install protoc and protoc-gen-go

```
$ sudo apt install golang-goprotobuf-dev
$ wget
https://github.com/protocolbuffers/protobuf/releases/download/v3.7.1/protoc-
3.7.1-linux-x86_64.zip
$ sudo apt install unzip
$ unzip protoc-3.7.1-linux-x86_64.zip -d protoc3
$ sudo mv protoc3/bin/* /usr/local/bin/
$ sudo mv protoc3/include/* /usr/local/include/
$ go get -d -u github.com/golang/protobuf/protoc-gen-go
```

Getting an error when trying to do git checkout

Also, the directory has changed in /home/user/go

```
$ cd $GOPATH/src/github.com/golang/protobuf/
$ GIT_TAG="v1.2.0"
$ git checkout $GIT_TAG
$ go install github.com/golang/protobuf/protoc-gen-go
```

Step 4: Install other required packages

Install bzr

```
$ sudo apt install bzr
```

Install other basic packages

```
$ sudo apt-get install autoconf automake libtool curl make g++ unzip
```

Install raptor2

```
$ cd ~/Downloads
$ wget http://download.librdf.org/source/raptor2-2.0.15.tar.gz
$ tar -xvf raptor2-2.0.15.tar.gz
$ cd raptor2-2.0.15
$ ./configure --prefix=/usr --disable-static && make
$ sudo make install
```

Step 5: Install the required python packages

```
$ pip install pymortar
$ pip install pyxbos
$ pip install googleapis-common-protos
$ pip install pyarrow
$ pip install grpcio
$ pip install grpcio-tools
```

Step 6: Set up wave

Either download the latest version of the code ($ git clone https://github.com/immesys/wave) and build it or contact Gabe Fierro <gtfierro@cs.berkeley.edu> for the compiled binaries of wave and waved, configuration file wave.toml and systemd file wave.service or check the latest releases. Assuming that you have files: waved, wv, wave.toml, wave.service, follow these instructions:

```
$ chmod +x waved
$ chmod +x wv
$ sudo cp wave.toml /etc/wave/
$ sudo cp wv /usr/bin/
$ sudo cp waved /usr/bin/
$ sudo cp wave.service /etc/systemd/system/
$ sudo systemctl daemon-reload
$ sudo systemctl start wave.service
```

Check if the service has started successfully:

```
$ journalctl -u wave -f
-- Logs begin at Tue 2019-04-23 19:13:12 UTC. --
Apr 24 16:47:47 <machine> systemd[1]: Started "WAVE agent".
Apr 24 16:47:47 <machine> waved[18079]: server started on 127.0.0.1:777
^C
$ vim ~/.bashrc
```

```
>> export WAVE_AGENT="127.0.0.1:777"
>> source ~/.bashrc
```

Step 7: Create wave admin entity for your machine.

This will be stored in /home/user/entities

```
Create a WAVE_DEFAULT_ENTITY for your machine (creating entities)

Create admin entity:
$ mkdir ~/entities
$ cd ~/entities
$ wv mke -o admin.ent -e 50y --nopassphrase
$ vim ~/.bashrc
>> export WAVE_DEFAULT_ENTITY="/home/user/entities/admin.ent"
$ source ~/.bashrc
```

Contact Gabe Fierro <gtfierro@cs.berkeley.edu> to grant permissions to publish and subscribed to your admin entity on the XBOS namespace along with the permission to grant permissions to other entities.

```
Once you've been granted permissions, do the following to make sure you have
permissions:

$ wv rtprove --subject $WAVE_DEFAULT_ENTITY -o adminproof.pem
wavemq:publish,subscribe@<namespace hash>/*
passphrase for entity secret:
Synchronized 8/11 entities
Synchronized 8/11 entities
Synchronized 12/12 entities
Perspective graph sync complete
wrote proof: adminproof.pem

$ wv verify adminproof.pem
```

Step 8: Setup wavemq

```
$ mkdir ~/wavemq
$ cd ~/wavemq
```

Contact Gabe Fierro <gtfierro@cs.berkeley.edu> for the compiled binary wavemq, configuration file wavemq.toml and systemd file wavemq.service and save these to ~/wavemq (or you can download the latest release or the code from https://github.com/immesys/wavemq and build the binaries). Then follow these instructions:

```
$ chmod +x wavemq
$ sudo cp wavemq /usr/local/bin/
$ sudo mkdir /etc/wavemq
$ sudo vim /etc/wavemq/wavemq.toml
>>
[WaveConfig]
  database = "/var/lib/wavemq/wave"
  # this is optional, but required if you want your site to operate with no
internet
  defaultToUnrevoked = true

  [WaveConfig.storage]
      # This is the default HTTPS server
      [WaveConfig.storage.default]
      provider = "http_v1"
      url = "https://standalone.storage.bwave.io/v1"
      version = "1"

[QueueConfig]
  queueDataStore = "/var/lib/wavemq/queue"
  # This is one day in seconds
  queueExpiry = 86400
  # 10k items (it will hit 100MB first)
  subscriptionQueueMaxLength = 10000
  # 100MB
  subscriptionQueueMaxSize = 100
  # 100k items (it will hit 1GB first)
  trunkingQueueMaxLength = 100000
  # 1GB
  trunkingQueueMaxSize = 1000
  # 30 seconds
  flushInterval = 30

[LocalConfig]
  # the address to connect to as an agent
  listenAddr = "127.0.0.1:4516"
```

```
[PeerConfig]
  # the address to connect to as a peer (not used for site router)
  listenAddr = "127.0.0.1:4515"

[RoutingConfig]
  PersistDataStore = "/var/lib/wavemq/persist"
  # This will be created for you
  RouterEntityFile = "/etc/wavemq/router.ent"
  [[RoutingConfig.Router]]
      Namespace = "<Contact gtfierro@cs.berkeley.edu> for the namespace hash"
      Address = "<Contact gtfierro@cs.berkeley.edu> for the designated router
machine"

$ sudo vim /etc/systemd/system/wavemq.service
>>
[Unit]
Description="WAVEMQ"

[Service]
Restart=always
RestartSec=30
ExecStart=/usr/local/bin/wavemq /etc/wavemq/wavemq.toml

[Install]
WantedBy=multi-user.target

$ sudo systemctl daemon-reload
$ sudo systemctl start wavemq
Check if it's started successfully:
$ journalctl -u wavemq -f
```

Step 8: Setup xboswave

```
$ cd
$ git clone https://github.com/gtfierro/xboswave.git
$ cd xboswave
$ git submodule init
$ git submodule update
Edit Makefile as below:
$ vim Makefile
>>
```

```
GOPATH = /home/user/go
PLUGINS=$(wildcard plugins/*.go)

.PHONY: proto
proto: proto/xbos.proto
	protoc -Iproto/ -Iproto/googleapis --go_out=plugins=grpc:proto
proto/*.proto

.PHONY: proto-py
proto-py: wavemq/mqpb/wavemq.proto
	mkdir -p python/pyxbos/pyxbos/wavemq
	mkdir -p python/pyxbos/pyxbos/wave
	cp wavemq/mqpb/*.proto python/pyxbos/pyxbos/wavemq
	cp wave/eapi/pb/*.proto python/pyxbos/pyxbos/wave
	cd python/pyxbos; \
	python -m grpc_tools.protoc -Ipyxbos/wavemq -I../../proto/googleapis --
python_out=pyxbos --grpc_python_out=pyxbos pyxbos/wavemq/*.proto; \
	python -m grpc_tools.protoc -Ipyxbos/wave -I../../proto/googleapis --
python_out=pyxbos/wave --grpc_python_out=pyxbos/wave pyxbos/wave/*.proto; \
	python -m grpc_tools.protoc -I../../proto -I../../proto/googleapis --
python_out=pyxbos ../../proto/*.proto; \
	sed -i -e 's/^import \(.*_pb2\)/from . import \1/g' pyxbos/*pb2*.py; \
	sed -i -e 's/^import \(.*_pb2\)/from . import \1/g'
pyxbos/wave/*pb2*.py


$ make proto

$ make proto-py

$ make ingester


Set up ingester
$ cd ~/xboswave/ingester

$ make sshhostkeygen

Create ingester entity:

$ wv mke -o wavemqingester.ent --expiry 50y
$ wv name --public --attester wavemqingester.ent <namespace hash> xbos
$ wv rtgrant --attester $WAVE_DEFAULT_ENTITY --subject wavemqingester.ent --
expiry 3y --indirections 0 "wavemq:subscribe@xbos/*"
$ wv rtprove --subject wavemqingester.ent -o ingesterproof.pem
wavemq:subscribe@xbos/*
```

```
$ vim ingester.yml
Change configurations here (password for the ingester shell)


Maybe do the next step using screen or systemd service:

$ ./ingester


In another terminal, check if Ingester shell is running:
$ ssh -p 2222 localhost

$ list

user@localhost's password:

XBOS/WAVE ingester shell

>>list

+----+----------+-----------+----------+--------+--------+---------+-------+-
-----------+

| ID | ENABLED? | NAMESPACE | RESOURCE | SCHEMA | PLUGIN | CREATED | ERROR |
ERROR TIME |

+----+----------+-----------+----------+--------+--------+---------+-------+-
-----------+

>>^D
```

Step 9: Example driver: dark_sky (weather)

```
$ cd xboswave/python/pyxbos/pyxbos/drivers/dark_sky
$ wv mke -e 50y --nopassphrase -o dark_sky.ent
$ wv name --public --attester dark_sky.ent <namespace hash> xbos
$ wv inspect dark_sky.ent
$ wv rtgrant --attester $WAVE_DEFAULT_ENTITY --subject dark_sky.ent --expiry
3y --indirections 0 "wavemq:publish@xbos/dark_sky/*"
$ wv rtprove --subject dark_sky.ent -o dark_sky_proof.pem
wavemq:publish@xbos/dark_sky/*

$ wv verify dark_sky_proof.pem


Create config file
```

```
$ vim dark_sky.yaml
dark_sky:
      api: <enter api key for the app>
      url: https://api.darksky.net/forecast/
      lat: 40.5301
      lng: -124.0000
wavemq:
      namespace: <namespace-hash>
```

Start the process:

```
$ python dark_sky.py
```

Add to ingester:

```
$ ssh -p 2222 localhost
```

```
$ add xbosproto/XBOS plugins/weather_current_plugin.so <namespace-hash>
dark_sky/*
```

Appendix F: User Interface Testing

Customer facing functions

The Goal of the user interface is to enable a commercial customer to respond to demand response or price signals appropriate to his/her specific business needs. The researchers developed use cases in functionality of the user interface and control platform. The most common use cases are: 1) monitor and control energy use manually, remotely, conveniently (such as check current temperature and modify temporarily during the day, for holidays, or seasonally), 2) engage in demand response (receive alert, choose level of engagement at the zone level), 3) manage energy use to reduce energy cost, and 4) check diagnostics.

The Objectives of the user interface design are to provide:

- a means of receiving demand response or price signals from the utility, including notification of future events
- a means of prioritizing and managing demand response strategies (e.g., increasing thermostat setpoint on hot days to reduce air conditioning) according to the specific needs and loads of the customer. This can include HVAC, lighting, plug loads, EVs.
- feedback to the customer of the effectiveness of that demand response strategy (e.g., did the strategy save money? Did it negatively impact the business (e.g., productivity, sales))
- single place to manage (group and schedule) multiple thermostats for open business periods, closed times, vacations/holidays, including the potential of remote control
- temperatures in the space by zone
- whole building energy data
- usability
- provide fault detection and diagnostics (e.g., are systems using energy during closed periods?)

User Interface Development

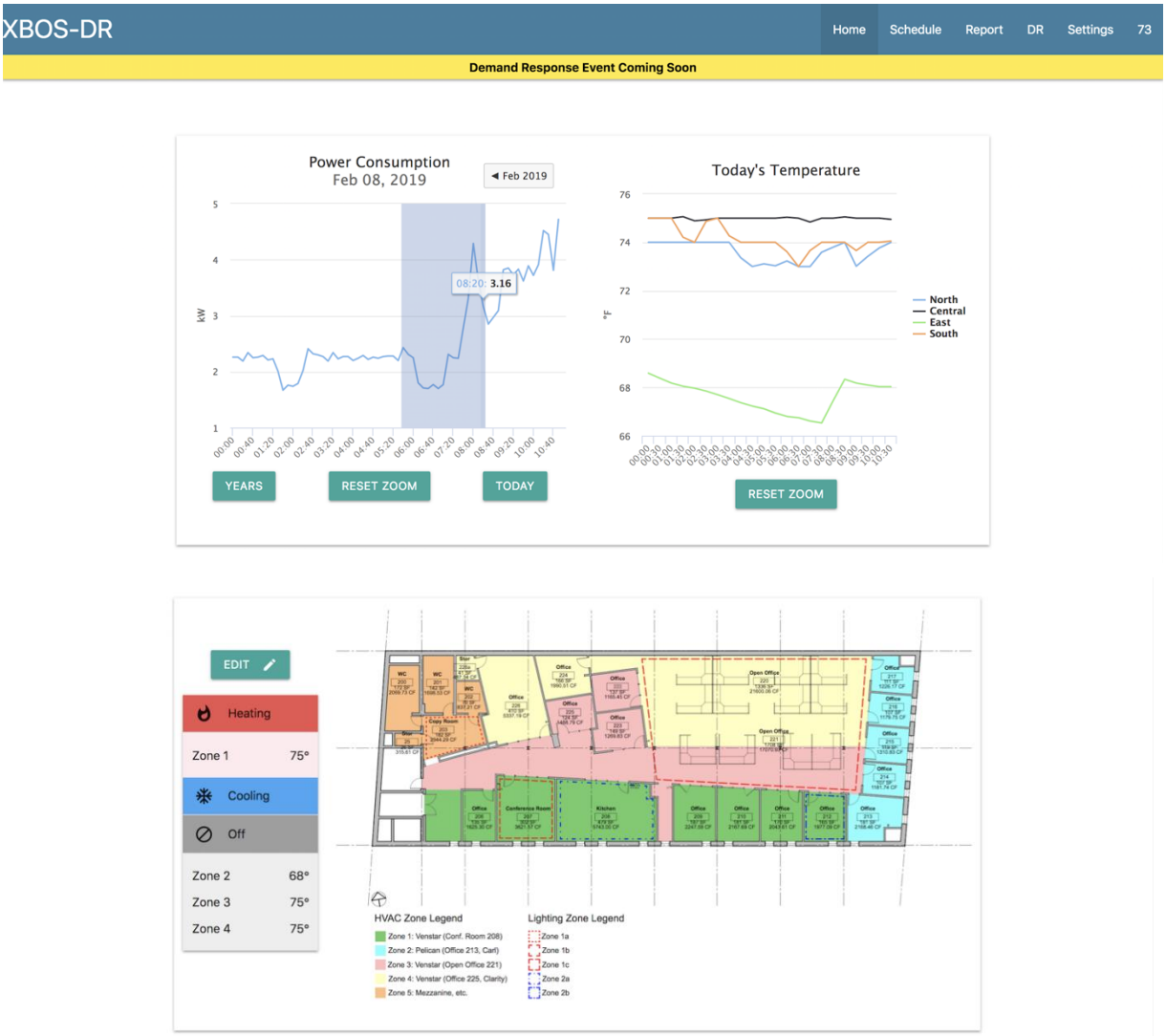
The research team developed a user interface for the project. The main sections of the user interface are the Home screen, Schedule screen, DR screen, Report screen and Settings screen.

The home screen shows the overall energy usage of the subject property, the temperatures in each zone, the floor plan of the area, and the ability to change the zone temperature setpoint temporarily.

Home Screen

The home screen should provide the most utilized functions: energy use from real-time and utility provided data, and price for the day so far, indoor temperature in various zones, current state of Heating Ventilation and Air Conditioning (HVAC) and lighting systems displayed on a floor plan, and demand response notification. Figure below shows a design of the interface.

Figure 40: Home screen of User Interface



The home screen shows notification of events as well as basic energy and temperature information.

Credit: Therese Peffer and Brandon Berookhim

Data displayed:

Notification of event (e.g., today, confirmed for tomorrow, likely in 2 days)

Whole building energy: so far today, last 12 months

Outside air temperature

Current price of energy

Temperature per zone so far today

Floor plan of building showing zones, state of zones (heating, cooling, or off) and zone temperature

Input from user:

Per zone, change setpoint temporarily (e.g., until the next scheduled period or for 15 minutes and learn if changed repeatedly)

Schedule screen

The schedule screen should provide a means of setting the heating and cooling setpoints for various epochs of time. Defaults are closed business hours and open hours; more periods could be added, such as prep time/maintenance time. The schedule should provide a means of grouping the zones (e.g., all six zones of the Basketball courts) to use the same heating and cooling setpoints.

Data displayed:

Names of zones for the building

Read from the current setpoints and open/closed period for each thermostat/zone

Input from user:

Changes in time period or epoch of setpoint duration (e.g., OpenHours from 8am to 6 pm): includes name of epoch, additions or deletions of epochs

Changes in setpoints per zone

Changes in start/end time of epoch

Ability to add Holiday setpoints

Ability to add Demand Response day setpoints

Demand Response

The Demand Response screen allows the user to simulate the loads of the building during a DR event, and select a level of disruption using the Cost-Comfort Index slider, which balances the temperature setpoint selection (within a safety range) for cost versus comfort.

The demand response screen should:

- Show simulation of future demand response day (predicted energy, indoor temperature, HVAC state, estimated cost during DR event, estimated cost over a baseline day) given a cost-comfort index. This can be shown by whole building and zone level.
- Allow the user to change the cost-comfort index (e.g., let the optimizer know whether one prefers to maximize saving money or providing comfort) and see the simulated effect of this index. This could be shown for the whole building and by zone.

Data displayed:

Past DR event days

Whole building (or zone) energy use saved (number, actual event vs. Baseline, kWh)
Cost savings compared to baseline (number)
Cooling Degree-Hour

Simulated DR event graphic

Whole building (or zone) energy use, (event hours shaded), HVAC state, price vs. time, given cost-comfort index (lambda)

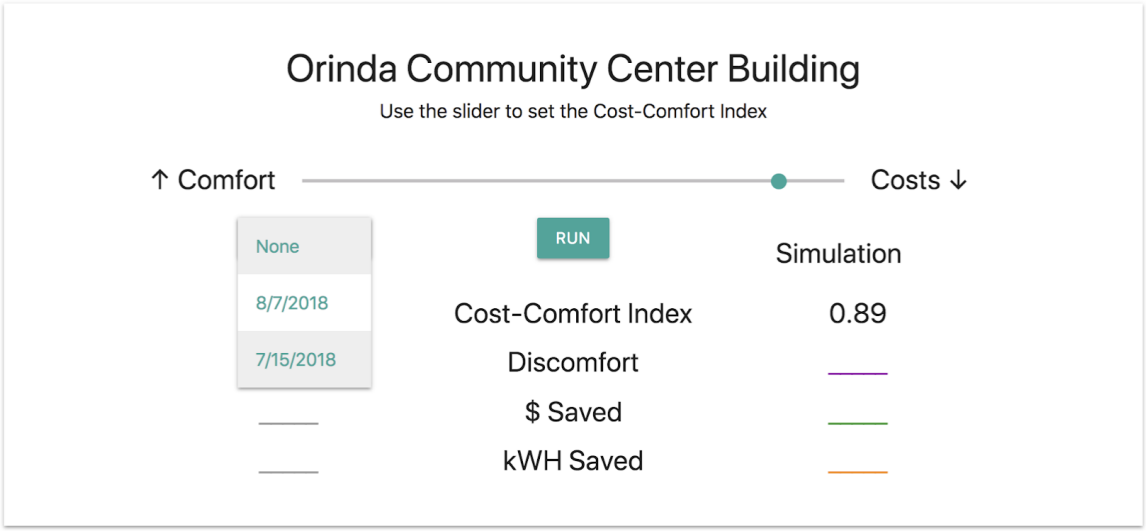
By zone numbers

Input from user:

Select building or zone view, select zone

Select cost-comfort index (lambda) **or** allow user to choose max allowable temperature during historical DR event per building or per zone

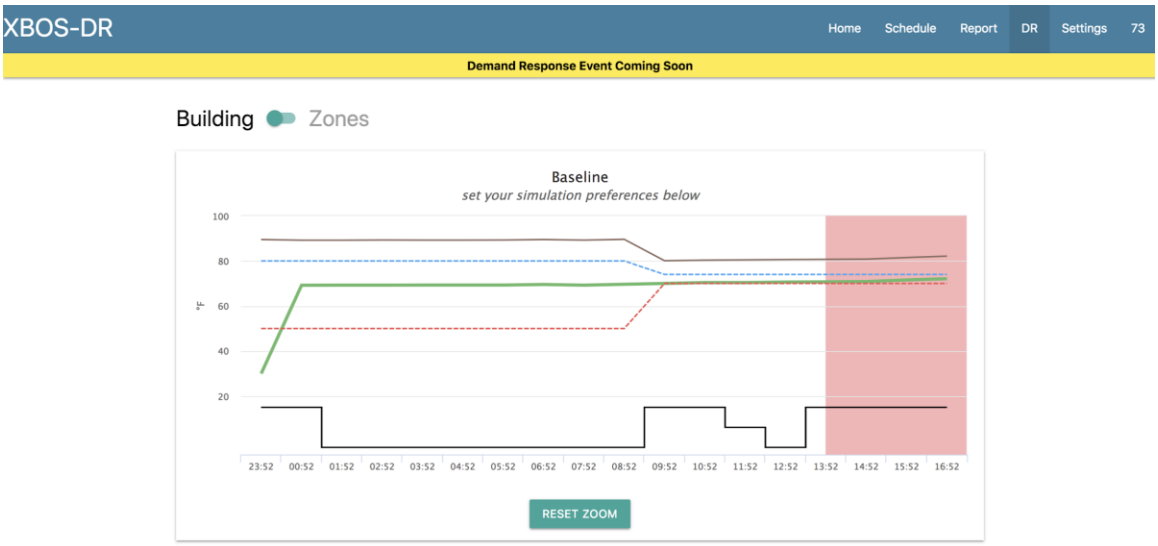
Figure 41: Cost-Comfort Index on DR screen of User Interface



The cost-comfort index allows one to select and simulate the desired level of balance between saving money or remaining comfortable during a demand response event.

Credit: Therese Peffer and Brandon Berookhim

Figure 42: Display of Simulated Data on DR screen of User Interface



The cost-comfort index allows one to simulate the effect on indoor temperatures, setpoints, and estimated energy consumption.

Credit: Therese Peffer and Brandon Berookhim

Report

The Report screen allows the user to diagnose, detect issues, and otherwise see the overall energy consumption of the building and zones. For the user interface test, this screen showed the zone analysis information.

Settings

The setting screen should allow one to:

Manage the notification of the demand response event

Change the temperature from F to C

Lock or unlock thermostat from interface

User Interface Testing Protocol

This section describes the method developed to test the user interface and the results.

Testing Protocol

The research team developed a protocol for the testing. The test location was a small conference room with a table and chairs. Test equipment was a laptop that held the user interface with live data from one of the building sites. Testing included at least two facilitators with a single subject for approximately 1-1.5 hours.

One facilitator read from a script describing the goal of the user interface: ("The Goal of the user interface is to enable, say a retail business or office-based organization who are commercial electricity customers to respond to demand response or price signals appropriate to his/her specific business needs."

The facilitator asked the subject to "Imagine you are the [office or facilities] manager of [the CIEE office or the movie theater]. You have this interface which allows you to view and control various things."

The facilitator told the subject that she would ask several questions, to conduct several tasks using the user interface. The facilitator asked the subject to "talk out loud" [well, first I'd look to see…then I'll try this button to see what it does], to "try to work it out yourself as best you can first, then we can answer questions and discuss afterwards." It is important that the subject knows that the user interface test is a test of how well the user interface works, "it is NOT to see how well you perform. It is common to feel dumb or frustrated, but that just means the user interface failed—not you."

The facilitators took notes, roughly measured time to complete task and if task was successfully completed, and noted any errors (choosing the wrong page tab) and points of confusion.

The facilitators had seven tasks, and added a few more as the test proceeded.

Tasks on the User Interface Test

In general the tasks progressed from the easy to more difficult.

Task 1: Please look at the interface and answer whether you think there is a demand response event happening soon, and if so what day.

Task 2: Please look at the interface and answer what is the building's power consumption at 9:30a.

Task 3: Please look at the interface and answer what is the current temperature of [name a room] right now.

Task 4: How might you change the temperature setpoint for [name a room] right now.

Task 5: Please look at the interface and answer how would you manage your building's response to a demand response event.

Task 6: How would you estimate the effectiveness of that demand response strategy—did it save money?

Task 7: How is the building's energy performance? Can you tell if the systems are using energy during closed periods?

General questions

Any questions? What do you think overall? What worked and what didn't, e.g., what was clear and what were the areas of confusion?

Testing Results

The research team tested three subjects: a mid-20s male (office administrator), a middle-aged female (director of operations), and a middle-aged female (energy services provider who worked with facilities managers).

Task 1: The home screen had a banner across the top that indicated that a demand response event was imminent. All subjects quickly responded that an event was occurring.

Discussion: the terminology is not clear: perhaps the event could be labeled more specifically according to the utility (e.g., PG&E has PDP events and SCE has CPP events) and the time of the event (e.g., event beginning at 2pm today, or lasting from 4-9pm today), rather than "happening soon."

Task 2: On the top left of the home screen is a graphic that shows the current power consumption of the building over the course of a day, with buttons that allow one to see previous months, and ability to zoom in on a certain time. All subjects found the graphic immediately; one subject quickly clicked on the graphic to see the exact number displayed in the pop up window. The other subjects found the exact number after a little help. The team asked follow up questions about navigating through this graphic: zooming in, finding historical data, and returning to the first screen. One subject easily found historical data and figured how to zoom and reset back; the other subjects easily found the other months, but needed more time to figure out the navigation.

Discussion: the font size for the y axis is likely too small. Suggest spelling out abbreviations (kilowatt instead of kw), revising the method to see other months or years, add text to indicate ability to zoom (with "Reset to zoom" button greyed out until this feature is used).

Task 3: On the top right of the home screen is a graphic that displays the indoor temperature of each zone in the building and the mean of all the temperatures. Subjects could quickly tell the temperature of a zone if it were selected. One subject answered the question by looking at the zone plan at the bottom of the home page.

Discussion: it wasn't obvious that one could select or deselect the visibility of the temperature by zone through the legend. A short piece of text could indicate this functionality.

Task 4: The bottom of the home screen has a floor plan with zones of the buildings in different colors; to the left of this floor plan are three colored bars which indicate which zones are

currenting heating, cooling or are off. Two subjects were able to find a way to change the temperature by typing text; one subject took slightly longer but was still able to finish the task.

Discussion: The colors of the zones are not chosen to reflect lack of hierarchy; subjects were confused if there was any meaning in color choice of zones. The term “override” was not immediately understood. Subject felt it was faster to type numbers than to use the arrow buttons (which are quite small) to change the temperature set point.

The research team added a question here regarding changing the schedule. This question required moving off the Home screen to the Schedule screen, first grouping the zones and setting up the setpoints, then choosing the epochs or time periods. The subjects required prompting to go to another screen, and had some difficulty navigating the various options. Once they got to the screen to schedule the time periods, some understood how to change time periods; less obvious how to remove time periods, save, and assign the setpoint modes to a time period.

Discussion: this question is notoriously hard for any thermostat, and in hindsight, the research team could have set it up differently. The question mixed up initial setup tasks (e.g., grouping like-zones together, and setting modes of setpoints) with everyday routine tasks (e.g., changing the temperature setpoint of a mode or changing the scheduled time to start or end differently). The term “epoch” was not understood. This task might be easier if there were default settings, and the subject had to change them. Once prompted to read the instructions, subjects were able to add and delete time period starting/ending nodes, and one subject grasped the concept of assigning setpoints to time periods. This screen has two “save” icons: one for the time period node placement and one for the overall schedule; these were confusing. The terms Open and Closed were not clear; might be better to use “business operating hours” or “closed hours”.

Task 5: This question required finding the Demand Response tab. Once subjects found it, they explored the cost-comfort index. Two understood that one could move the pointer across to choose different settings.

Discussion: The labeling for the cost-comfort index was confusing. The team discussed that the baseline mode is full comfort (0 or all the way to the left), so the pointer’s default should be all the way to the left. Then the user could advance the pointer to the right (e.g., this would represent full cost-savings using the do not exceed setpoints for the thermostats). Subjects were confused as to what was simulation versus control.

Task 6: The top graph on the Demand Response screen showed a simulation of indoor temperatures, setpoints, HVAC state, and energy. The subjects found the graphic and two correctly interpreted the dotted lines as simulation.

Discussion: Currently the simulation engine only shows four hours, which is confusing.

Task 7: This task required going to the Reports screen. Once subjects found it they were presented with a dense table of numbers.

Discussion: The subjects felt that the table could be sortable, or highlighted with color the cells that showed problems areas.

In general: the subjects liked the user interface and felt it was intuitive and did not overwhelm with information. The user interface provides a lot of information in one place in an easy to navigate way that is not currently available.

Conclusions/Recommendations

The research team developed and tested a user interface for the XBOS-DR platform which provided several functions. The Objectives of the user interface design were to provide:

- a means of receiving demand response or price signals from the utility, including notification of future events
- a means of prioritizing and managing demand response strategies (e.g., increasing thermostat setpoint on hot days to reduce air conditioning) according to the specific needs and loads of the customer.
- feedback to the customer of the effectiveness of that demand response strategy (e.g., did the strategy save money? Did it negatively impact the business (e.g., productivity, sales))
- single place to manage (group and schedule) multiple thermostats for open business periods, closed times, vacations/holidays, including the potential of remote control
- visualizing temperatures in the space by zone
- visualizing whole building energy data
- usability
- provide fault detection and diagnostics (e.g., are systems using energy during closed periods?)

This task took much longer than expected, partly due to student matriculation and partly due to its complex nature, as it involved front end and back end development to create the live interface. The user interface testing showed that many of these goals were achieved, namely: receiving and understanding the demand response signal, managing DR events, managing multiple thermostats, visualizing temperatures in the space by zone, and visualizing whole building energy. Scheduling and reporting could be made easier through simple changes to the interface.

The research team plans to continue to develop and test the user interface for other projects.