HPXML Implementation Guide

Release 2.2

Julie Caracino (HPC)
Gavin Hastings (Tierra Resource Consultants)
Noel Merket (NREL)

Aug 08, 2017

Contents

The Home Performance Extensible Mark-Up Language (HPXML) Implementation Guide is designed to help program administrators, implementers, and software developers integrate HPXML into their operations and products. HPXML is an open data standard published by the Building Performance Institute (BPI) created to support a growing industry by facilitating communication and the exchange of information and data on residential buildings and energy performance.

HPXML is comprised of two standards:

- Data dictionary (BPI-2200) that defines terms and data formats related to the physical attributes and performance of buildings and measures, and
- Standard extensible mark-up language (XML) data transfer protocol (BPI-2100) that can be used to exchange data defined in the dictionary between different software systems.

If you are thinking about implementing HPXML, or just want to learn more about the data standard and how it may benefit your program or business, then this implementation guide will be a valuable resource. The guide is divided into two sections that are tailored to specific audiences.

- The *Program Administrator Guide* discusses the benefits of HPXML adoption and walks administrators through the process of goal setting, identifying data needs, data validation, testing, and quality management.
- The *Software Developer Guide* provides developers with information on versioning, document structure, XML element references, and use case validation. The section also provides links to sample HPXML files.

HPXML is periodically updated to reflect changes in the needs of the market and the stakeholders that are using the standard. Whenever a new version of HXPML is released, the Implementation Guide will be updated to reflect changes to the data standard and how it is being used in the market. If you have questions about HPXML or believe you can contribute to the overall success of its deployment, please email us at hpxml@homeperformance.org. For more information on HPXML, visit http://www.hpxmlonline.com.

CHAPTER 1

Overview of HPXML

What is HPXML?

In today's existing homes market, the electronic exchange of information on energy performance, energy conservation measures, and the physical and operational attributes of a home is often characterized by a lack of common terms, definitions, and two-way feedback systems. This limits the ability of decision-makers to access, aggregate, share and use data for the design and implementation of residential energy efficiency programs. This also makes it difficult for market actors to understand the drivers of variation in performance (e.g., of measures, contractors, programs, and homes), identify investment opportunities, analyze market trends, and project savings from energy efficiency investments.

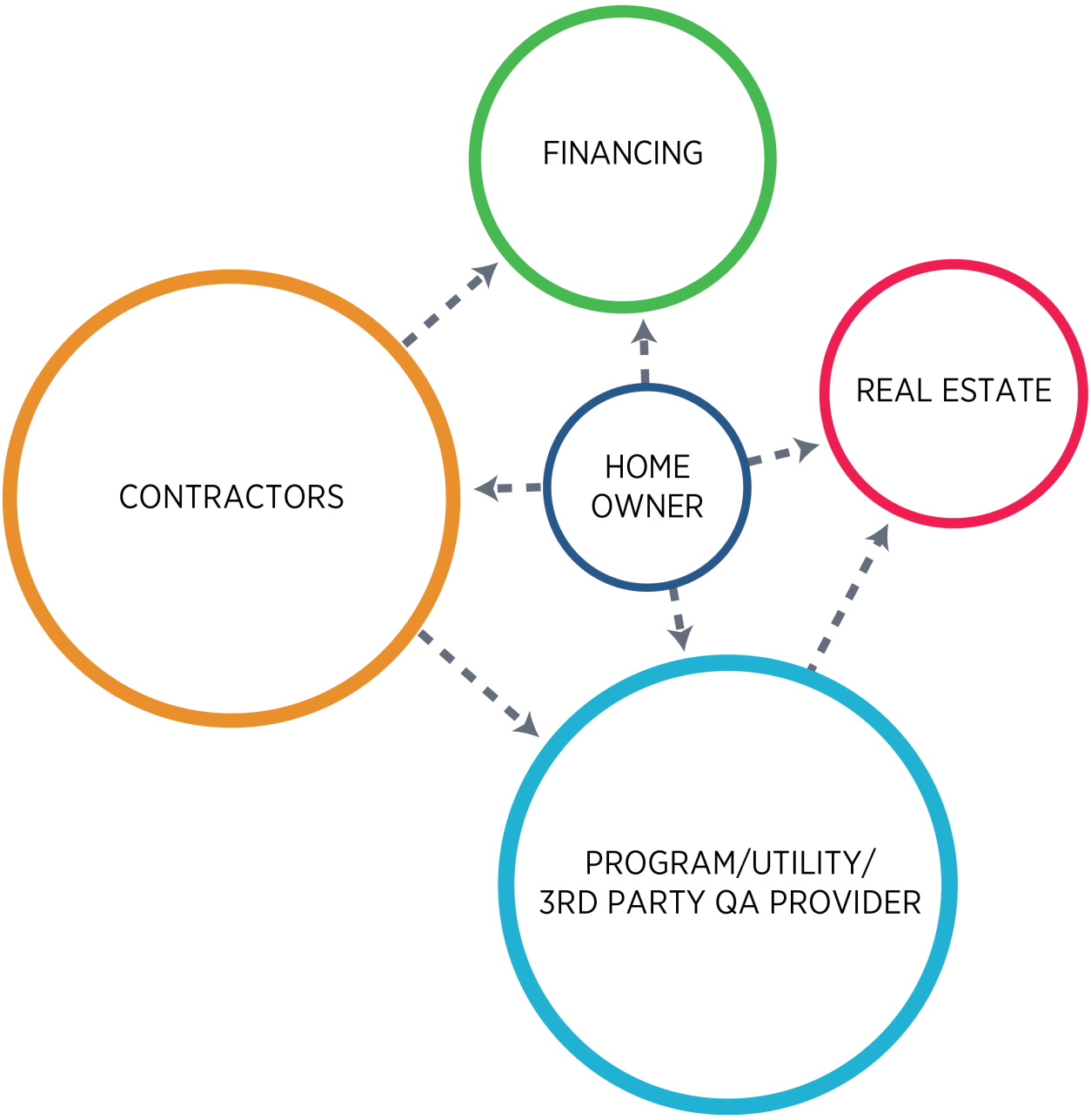


Fig. 1.1: Figure 1: Existing Market Conditions

Open data standards, like HPXML, are a powerful solution to some of the most intractable problems facing the residential energy efficiency industry. Data standards are crucial in enabling the quantification of energy savings that result from energy efficiency upgrades that in turn allow for savings guarantees, low-cost consumer finance, and the sale of energy efficiency into energy, capacity, and carbon markets. Data standards are crucial for ensuring that energy efficiency improvements in homes are properly valued in real estate transactions. Finally, data standards are crucial for supporting ongoing research into the best methods for making homes energy efficient because the aggregation of standardized data supports comparability and comprehensive analysis of program and contractor performance.

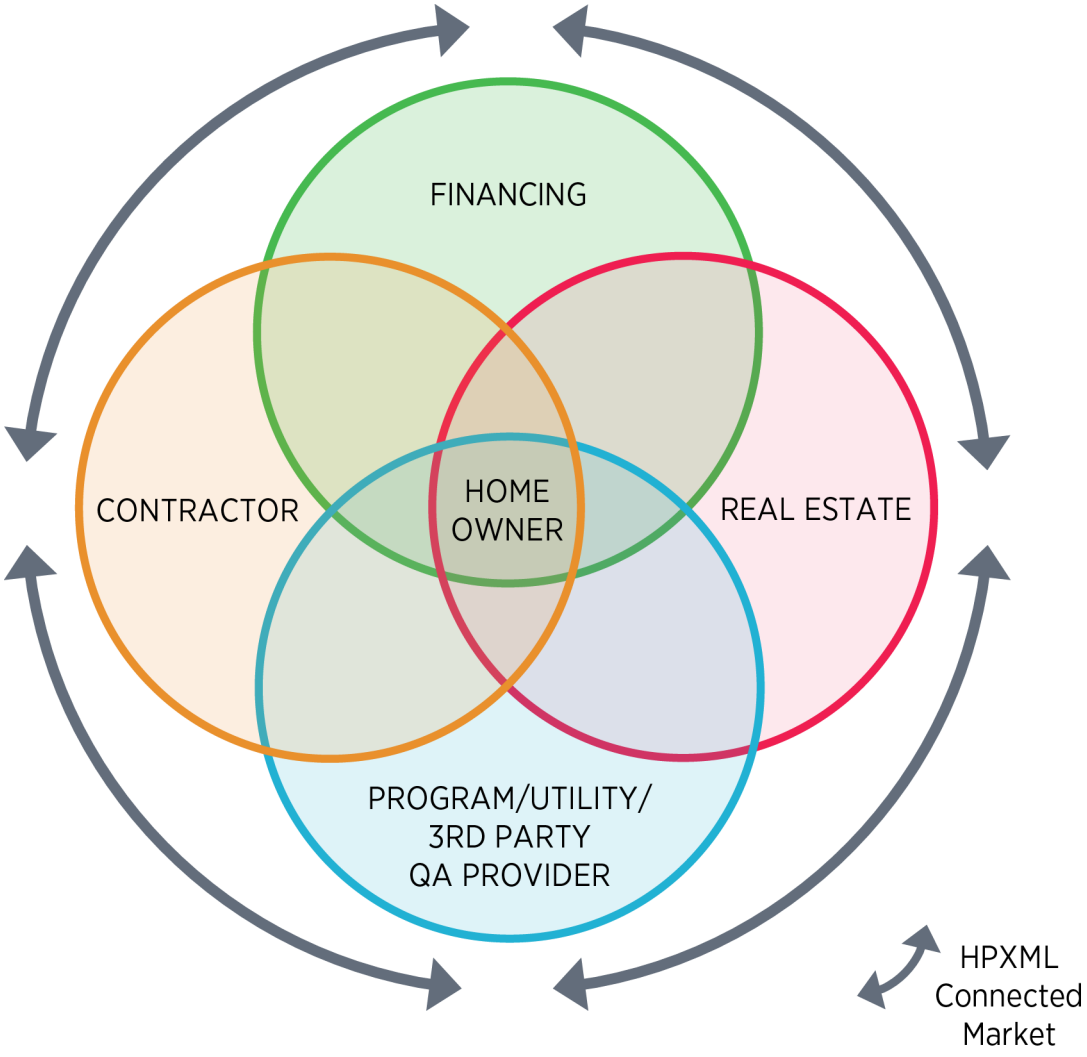


Fig. 1.2: Figure 2: Connected Market

HPXML is comprised of two BPI data standards that were published in 2013 to support a growing industry by facilitating communication and the exchange of information and data on residential building and energy performance. The purpose of HPXML is to:

- Standardize terminology and facilitate the collection of higher quality data as a means of tracking and quantifying work being completed across the residential energy efficiency industry;
- Create interoperability between software systems to allow the transfer of residential energy efficiency data across a diverse set of market actors, such as the real estate and financial sectors; and,
- Improve industry efficiency by reducing the costs of data collection and exchange between market actors.

The HPXML Data Dictionary (BPI-2200) standardizes terms and data formats related to the physical attributes and performance of buildings and measures. It defines the data elements necessary to provide a general description of a whole house or single measure energy efficiency upgrade for reporting, rebate and basic quality assurance (QA) purposes. The standard includes several smaller datasets that are being used to standardize data collection and reporting for specific use cases, for example, data required at the completion of a whole-house program (see section on Standard HPXML Datasets).

The HPXML Transfer Standard (BPI-2100) provides requirements for a standard extensible mark-up language (XML)

data transfer protocol that can be used to transfer information and data between different software systems. The transfer standard is the companion standard to the HPXML data dictionary. Each of the data elements defined in the data dictionary can be transferred using HPXML.

Data standards, like other standards, are more valuable the more broadly they are used. Every time a program implements a data standard, transactional costs fall for firms already using the standard. Software vendors can provide a rich set of data about a whole-house upgrade to programs using HPXML at very little additional cost because they have already made the initial investment to design their software to support the data standards. For the energy efficiency industry to realize the full benefit of the BPI data standards, it is crucial that more programs and other users adopt the standards.

Aligned with Industry Standards

HPXML is the most widely used implementation of the Department of Energy's (DOE) Building Energy Data Exchange Specification (BEDES). BEDES is a dictionary of terms, definitions, and data formats created to facilitate the exchange of information on building characteristics and energy use for the commercial, multifamily and residential industries.

HPXML is also aligned with the Real Estate Standards Organization's (RESO) Data Dictionary, which standardizes terms used in Multiple Listing Services (MLS) and other source providers nationwide so that information can be easily shared and understood in the real estate industry.

The benefit of having HPXML aligned with the RESO Data Dictionary and with BEDES is to increase interoperability across sectors and industries. Home energy efficiency information, including score and labels, can be transferred to MLS databases.

Management and Version Control

HPXML Working Group

HPXML has been developed, and is modified and maintained, by the stakeholders who use it, including organizations and individuals from residential energy efficiency programs, software development, government, home performance contracting, and the nonprofit sector.

The Home Performance Coalition (HPC) chairs a national working group of industry experts with oversight from the BPI Data and Modeling Standards Technical Committee using an open and consensus based decision-making process set by BPI. The working group meets quarterly to discuss updates and to vote on new versions of the standards.

Membership and participation in the HPXML working group is voluntary and open to all organizations and individuals interested in HPXML.

The HPXML Working Group can be a valuable resource for program administrators, implementers and software developers that are interested in adopting HPXML.

Versioning

The HPXML schema follows the Semantic Versioning 2.0 specification[1]. The version numbers follow a pattern of Major, Minor, and Patch (e.g., 2.1.0). The major version number is incremented when the schemas are changed in a manner that is incompatible with previous versions. Examples of changes that require a major version change include renaming elements, removing elements, moving elements, and removing enumerations.

[1] http://semver.org/

The minor version number is incremented when the schemas are changed in a manner that is backwards compatible with previous versions that share the same major version. In other words, a document created in a previous version of the schema will also validate against the new schema. Examples of changes that require a minor version change include adding elements, adding enumerations, and changing the annotation in the schema for an element.

Technical documentation can be found at http://www.hpxmlonline.com/.

HPXML Compliance

Stakeholders that wish to comply with the HPXML Data Dictionary and Transfer Standard shall use HPXML data in all cases in which an HPXML data element is sufficient to adequately represent the person, characteristic, concept or other home-related datum that is being described.

Data can be adequately represented by HPXML if HPXML data elements, singularly or in combination, can be used to describe the person or thing in a way that:

1. Could reasonably be understood by other energy efficiency professionals; and,
2. Does not result in significant loss of information or create significant risks of miscommunication.

Software developers may validate their HPXML files against the schema and its datasets by uploading files to the HPXML Validator[2]. Sample HPXML files are available on Github[3].

Continued Growth

When reviewing the datasets that are required for your implementation, it is possible to identify a data point you require that is not in one of the pre-defined datasets or in HPXML. If this is the case, the HPXML working group can assist in adding the new data element and in identifying how to incorporate it into the standard.

See *How to Add Data Elements* for details.

[2] https://hpxml.nrel.gov/validator/
[3] https://github.com/hpxmlwg/hpxml/tree/master/examples

CHAPTER 2

Standard HPXML Datasets

One of the areas in which data standardization holds the greatest promise is in data collection and reporting. If different energy efficiency programs all agree to collect the same (or very similar) data for the same use cases, such as the data reported by the contractor when the initial audit is conducted or the final job and test-out results are reported, information technology (IT) costs could be reduced significantly across the country.

Software vendors would be able to use the same reporting template, with only minor modifications, for multiple programs. Standard datasets would also greatly facilitate cross-program comparability as well as support research efforts and accurate quantification of savings for other uses.

HPXML includes four standard datasets that program administrators can adopt to standardize data collection and reporting for various use cases, including:

- Transfer of project data from third-party energy modeling or data collection tools to a program management database; and
- Export energy efficiency project data, including the DOE's Home Energy Score, to local MLS.

Each of the datasets specifies a number of required data elements that can be collected at specific points during the implementation of a residential energy efficiency program. Below is a summary of the datasets, followed by a discussion in the next section of the benefits of adopting data standards.

Audit

The audit dataset is designed for use by Home Performance with ENERGY STAR® or other whole-house programs that require auditors to submit energy audit results and a proposed workscope to the program for review. The dataset describes the baseline building with a proposed work scope. Required fields include data on the home's existing characteristics, health and safety needs, recommended improvements, and savings predictions.

The audit dataset was established through a consensus process of three geographically diverse existing whole-house programs, and is intended to meet the needs of most programs.

Programs that wish to adopt the audit dataset may download the Data Selection Tool[4] for guidance on the required data fields. Software developers may visit Github for an example of an HPXML audit file.

[4] http://www.hpxmlonline.com/tools-resources/data-selection-tool/

Upgrade

The upgrade dataset is designed to facilitate the transfer of information on completed whole house retrofits from contractors to Home Performance with ENERGY STAR ® or other whole house programs.

The upgrade dataset describes the baseline building (e.g., the pre-upgrade condition of the home, with proposed measures) with a completed work scope (e.g., description of installed measures, with modeled or predicted energy savings). The upgrade dataset was established through a consensus process of three geographically diverse existing whole-house programs, and is intended to meet the needs of most programs. Programs that offer more diverse incentives may need to add HPXML data elements to meet program needs.

Programs that wish to adopt the upgrade dataset may download the Data Selection Tool for guidance on the required data fields. Software developers may visit Github for an example of an HPXML upgrade file.

Home Energy Score

The Home Energy Score (HEScore) is similar to a miles-per-gallon rating that helps homeowners and homebuyers understand how much energy the home is expected to use. The HEScore also provides information on how to make the home more energy efficient.

The Home Energy Score Translator[5] was developed by the National Renewable Energy Laboratory (NREL) to generate HEScore inputs from HPXML files. The translator is available open source as a stand-alone script. It is also incorporated into the HEScore API. By using this translator, software developers can leverage their investment in HPXML to provide HEScore functionality with minimum development cost.

Home Performance Certificate

The Home Performance Certificate (BPI-2101-S-2013 Standard Requirements for a Certificate of Completion for Residential Energy Efficiency Upgrades) is a nationally recognized protocol that creates a bridge between the energy efficiency, real estate, and appraisal industries. The standard identifies a standard set of data elements for certificates that document the completion of a whole-house energy upgrade (HEU) and individual energy conservation measures (ECMs). A certificate that complies with the requirements of the standard can be branded and issued by home energy upgrade programs or by entities implementing nationally recognized third-party quality assurance (QA) programs.

The set of data elements required for inclusion in the certificate provides a clear, easy-to-understand description of the HEU or ECMs, including information about major energy-related improvements implemented, and, if relevant, predicted energy savings or other performance indicators. The certificate can be designed to be used as a reference document by real estate agents, appraisers, and other professionals during the home sale process. The certificate may be attached to an MLS listing sheet and shown to the buyer and buyer's agent as a demonstration of the home's relative energy efficiency, which may increase the home's market value. The certificate may also be used by appraisers and underwriters as a source of information about characteristics of a home related to energy consumption and energy savings.

The information in a BPI-2101-compliant certificate is aligned with the Real Estate Standards Organization's Data Dictionary, which specifies a standard vocabulary for MLS systems. This alignment allows information from a BPI-2101-compliant certificate to be more easily transferred to MLS systems. The information is also designed to align with data fields included on the Appraisal Institute's Residential Green and Energy Efficiency Addendum (AI Addendum). The AI Addendum is an attachment that appraisers can voluntarily elect to include in their residential appraisal to comply with Uniform Standards of Professional Appraisal Practice (USPAP) when completing a high-performance home assignment.

[5] https://github.com/NREL/hescore-hpxml

The scope of the standard does not include the appearance of a certificate. The certificate's sponsor may choose a certificate design that best meets its programmatic needs.

Arizona Public Service and Salt River Project currently are issuing the certificate to homeowners that complete the Home Performance with ENERGY STAR ® program.

Future HPXML Datasets

The development of standard datasets associated with other use cases, particularly evaluation, measurement, and verification (EM&V), is an important additional priority because it will generate additional cost savings and promote program adoption of the standards. New datasets will be published as they are identified.

CHAPTER 3

Program Administrator Guide

Steps to HPXML Implementation

A well-designed implementation plan is necessary to successfully integrate HPXML into program operations. This section will walk you through the process of implementing HPXML and provide guidance that can serve as a starting point for an implementation plan.

Each jurisdiction will have different goals, market needs, and regulatory requirements that will influence the scope of the HPXML project. However, leveraging the experiences of programs that have already implemented HPXML can lower individual implementation costs and help drive alignment between utility jurisdictions.

Step 1: Set Implementation Goals

HPXML can drive value into a program. Administrative costs associated with data entry, project review, incentive processing, reporting, and quality assurance can be reduced by adopting national data standards. Program administrators implementing HPXML may also elect to open the market to multiple data collection and energy modeling tools that comply with HPXML, allowing contractors to use the tool that best meets their business needs. The key to delivering this value is to determine the goals and scope of the project as early in the process as possible because these decisions will shape your implementation tactics. To start, identify the following information:

- **What is the baseline now and where do you want to be?** For example, if you want to "reduce the data collection burden for program partners," you will need to know how much it currently costs contractors in time or money to participate in the program. Then work with partners to identify a goal.
- **What are some of your program's current pain points?** Are there specific elements of your current program implementation that drive customer or contractor dissatisfaction? Are there any procedures in your process that create opportunities for errors or mistakes?
- **How will this help achieve overall program goals?** What is the current number of projects per year and what is the target number of projects you would like to complete? If you are focused on volume, you may want to automatically generate Home Energy Score or the Home Performance Certificate to promote a market-based strategy to increasing residential energy efficiency.

- **Are there local rules or regulations you need to consider?** Utility program regulators may require specific project details to be reported. You should coordinate with your local experts in these fields to help identify potential issues that will need to be addressed.
- **Which aspects of our overall goal should be prioritized?** Information technology (IT) solutions are often implemented in phases. Prioritizing your needs and implementation will help you save money and better manage resources. It is also important to engage stakeholders affected by the project (e.g., trade allies and contractors) to better manage expectations. This will also help reduce the pressure on the industry.
- **What is a realistic timing and rollout schedule?** The balance of the implementation document will help you identify the tasks you must complete to implement HPXML. Give yourself plenty of time to complete these tasks and, if possible, plan your testing and launch schedule in the low season in order to reduce contractor burden.

By thinking through the questions strategically and identifying your project goals, you should be prepared to design the balance of your implementation plan.

Step 2: Incorporate Stakeholder Feedback

HPXML can be a powerful tool to facilitate trade ally and contractor choice in software systems while delivering consistent and reliable data. It provides a great opportunity to improve contractor satisfaction and operational efficiency.

To maximize your potential for success, it is important to establish a method for collecting feedback from trade allies, contractors, and the software community throughout the HPXML implementation process. You may want to collect feedback and cultivate support for the HPXML goals you have established for your program in *Step 1: Set Implementation Goals*.

Trade Ally Feedback

Consider working closely with your local trade ally network, trade associations, or energy-efficiency contracting network to collect feedback important to your planning process. For example, some program administrators have hosted lunch and learning meetings prior to starting the design process. This has allowed the program administration team to identify the highest priorities for trade allies and cultivate support for the new program design. Ultimately, this has helped focus the project scope to deliver successful results.

When engaging your contracting network, consider a few key topic areas:

- Pain points for data collection and reporting
- How program goals will impact contractors' businesses
- Priorities for program design and subsequent rollout
- Early adopters who can assist in testing and give feedback on initial designs
- Functionality and user experience desired from the new program software environment

Identifying a representative group of trade allies and contractors that can assist throughout the process will be important in later steps of the implementation plan. Many successfully implemented programs have coordinated with contractors by setting up regular meetings, hosting dinners to collect feedback, or working closely with representatives from local trade associations.

Software Vendor Coordination

As the program administrator, you need to decide on the minimum qualifications for software to participate in your program. It is recommended to begin communication with software vendors as soon as your project plan is developed. The HPXML Working Group can help connect you to all of the software vendors that are currently coding to HPXML.

Since many of these vendors have implemented in other jurisdictions, they can offer valuable support for a program's planning process.

Programs that are considering opening the market to multiple modeling software tools may want to review the various features of HPXML-compliant energy modeling and data collection tools. Requesting a demonstration from each of the vendors is a good place to gain a general understanding of how the tools work. Each vendor takes a unique approach, which promotes innovation, and helps contractors deliver high-quality proposals and energy savings feedback to customers. Reviewing how each of the tools work will give your program an understanding of the current HPXML features software vendors are currently supporting. This will enable your program to be well prepared for the next step of determining data needs.

Step 3: Identify Data Needs

One of the areas in which data standardization holds the greatest promise is in data collection and reporting. If different programs all agree to collect the same (or very similar) data for the same use cases, such as the data reported by the contractor when the initial audit is conducted, or the final job and test-out results are reported, information technology costs could be reduced significantly across the country because software vendors would be able to use the same reporting template, with only minor modifications, for multiple programs. Adopting the HPXML standard datasets also will greatly facilitate cross-program comparability, and support research efforts and the accurate quantification of savings for other uses.

HPXML defines a long list of data elements, many of which may not be needed for your project. To determine which data points are needed, we recommend using the standard datasets described in *Standard HPXML Datasets* as a starting point, while considering the goals you identified in *Step 1: Set Implementation Goals*. Even though the Audit and Upgrade datasets were initially established for whole-house programs, many of the data points are relevant to other types of residential programs, for example, weatherization or single-measure programs. Also, many software vendors are already able to transfer the data required in these datasets. Any subset of data within these datasets will be easy to implement for existing HPXML-compliant tools.

The goal of this exercise is to identify the minimum data collection requirements needed to meet project goals. This means identifying what must be collected in the field, what must be transferred to your program management system, and what you are transferring and reporting to others, where applicable.

The BPI data standards support the collection and transfer of additional data points beyond these datasets. However, additional data may require software vendors to make substantive changes to their software. Program administrators should recognize the financial impacts of custom data collection requests and consider providing financial assistance to software vendors to meet any customization requests, as appropriate.

Data Selection Best Practices

When exporting or importing data from third-party data systems, it is important to develop a concise set of data requirements. As stated above, HPXML is capable of describing and transferring a large universe of data, but only a portion of data may be needed for a particular use case. Examples of use cases include:

- Reporting on the baseline conditions of a home and proposed improvements
- Reporting on the improvements to the home compared to baseline conditions
- Reporting on health and safety testing
- Reviewing a contractor's work as part of a QA process
- Reporting a home's energy efficiency assets to the parties in real estate transactions
- Reporting data on program activity to DOE and other agencies

Before reviewing the standard datasets, there are several best practices to consider.

Be Transparent with Stakeholders

Transparency with your contractors and software vendors throughout the process will help guide programmatic decisions and prevent challenges down the road. This is particularly true for the data selection process as stakeholders can bring attention to potential challenges that need resolution. For example, a program may request a specific data point that is not typically collected by most software products. If vendors can identify this need early on, a resolution can be reached in the planning phase and not delay the project later on.

Be Sensitive to Data Collection Burden

When selecting data requirements, programs should collect just enough data for program compliance, and measurement and verification of results. As a guiding principle, Home Performance with ENERGY STAR® recommends that programs employ administrative procedures that minimize the burden of participation for contractors and homeowners. When choosing mandatory data points for your program, it is important to recognize that every data point collected has a collection cost to contractors. Collaborate with evaluators, contractors, and software vendors to explore solutions that meet the need of the program as cost efficiently as possible.

Leverage Data Choices Made by Other Program Administrators

HPXML is easily extensible and therefore can be customized to the specific needs of a program. However, program administrators should look at the standard datasets being implemented by other programs around the country as many software vendors are already exporting and importing these standard datasets. When a program is using unique data requirements, the associated customization can push significant development costs for that specific program onto other parties, like software vendors and service providers. To streamline implementation and best leverage efforts from other programs, it is encouraged to coordinate with other program administrators to minimize the number of program-specific data fields required in your jurisdiction. This guide provides an HPXML data selection tool below to easily facilitate this coordination.

Schedule Updates to Data Requirements

Software developers tend to work in phases to control releases of their software. To avoid additional costs or confusion, try scheduling regular updates once or twice a year and communicate future changes as early as possible. This will help with version control and create a more manageable process for software vendors and your implementation team.

Adopting these best practices will assist in establishing a streamlined data selection process.

HPXML Data Selection Tool

To assist program administrators in reviewing the standard datasets that are being implemented by other programs, a data selection tool is available to help identify the required data points. The tool also helps programs select and communicate the requirements for the program to contractors, trade allies, and other stakeholders. The HPXML Data Selection Tool is a "living document" so if a program identifies data points that are not in the datasets, the HPXML working group can assist in adding them to the tool to meet the program's needs.

- Download: `HPXML Data Selection Tool`

The attached instructional video provides a walkthrough of the HPXML data selection tool. By using the tool, programs can quickly select the data required for HPXML program implementation. Programs can then forward the tool to the implementation team, trade allies and software providers, giving them clear guidance on the requirements for HPXML Implementation with your program.

YouTube: HPXML Data Selection Tool Tutorial

Additional details about the data structure and standard datasets are provided in the sections below.

Understanding the HPXML Data Structure

In the reviewing tool, notice the following descriptors for each data point:

- Data Category - A general description of the information at the building characteristic or contact information level. Note: Insulation has several data categories, depending on the insulating plane. For example, referencing insulation installed on the attic floor "Attic Floor Insulation" or on the bottom of the roof deck "Attic Roof Insulation."
- Data Element - A specific data point or descriptor within that data category. For example, insulation material type or R-value.
- Data Type - How the data should be provided, for example, as a number, text, enumeration, etc.
- Definition - A written description of the data point and what it means. Because the name of the data point is not always clear, this provides a narrative explanation of what each data point describes.

In most cases, there are several data points needed to describe any one building characteristic. For example, if you require blower door testing in your program, you will require "Air Infiltration" information. In this case, there are three data points that are required to describe an air leakage measurement, such as 2000 CFM_{50}:

"Building Air Leakage" = 2000

"Building Air Leakage Unit" = CFM

"House Pressure" = 50

This provides flexibility to receive the same data in multiple formats. For example, air leakage could be represented in CFM_{50}, ACH or ACH_{50}.

Setting the Program's Data Requirement Level

The program administrator's main task in this step is to determine the data element "requirement level." This sets the minimum requirements for software tools to participate in your program. In each of the standard datasets, there are two requirement levels:

- Required - All software must collect this data point and transfer it any time it exists in a home. This usually is driven by rebate qualifications or quality assurance requirements.
- Optional - Not required.

The *HPXML Data Selection Tool* will allow you to see the minimum required fields that have been agreed upon by the HPXML Working Group for the audit and upgrade datasets. In addition, you can use the Home Energy Score and Home Performance Certificate requirement toggles. By activating these toggles, you can see which fields would be required if you wanted to complete a Home Energy Score or to comply with the Home Performance Certificate standard.

The grayed-out fields are optional and represent fields that are relevant in many programs, but not required in the standard datasets. You can choose to make optional fields required in your program. However, not all software products on the market collect every possible data point. Making some of these data points "required" may restrict which products are eligible to participate in your program, or may require you to pay software vendors to code their software for this requirement. Communication with potential software vendors is important. Consider their feedback on required fields before determining your final data requirements. The HPXML Working Group can help facilitate that conversation in a constructive manner.

Some programs with a large number of measure-specific rebates are choosing to identify data points as "optional" to allow flexibility in implementation. Your program can use a minimum data collection standard that is required for

every home. However, if the contractor or software vendor wants to participate in the full spectrum of rebates, they can choose to send "optional" fields that trigger a rebate payment. This allows a diverse set of software products and contractor business models to participate, without mandating that every software and contractor support the full spectrum of rebates your program portfolio may offer. If you want to choose this path, it will be important to provide clear specifications on which "optional" fields will trigger which rebate payments.

How to Add Data Elements

When reviewing the datasets that are required for your implementation, it is possible to identify a data point you require that is not in one of the pre-defined datasets or in HPXML. If this is the case, the HPXML working group can assist in adding the new data element and in identifying how to incorporate it into the standard.

In some cases, this might include adding new elements to the standard to account for data points that could be applicable across many programs. To submit a new data element for consideration, you can use the working group's GitHub[6] account. This way, all members can see your recommendation and address it immediately. You can also email your request to hpxml@homeperformance.org.

Follow the steps below to submit requests:

1. Sign up for a user account on GitHub[7].
2. Go to the HPXML GitHub issues page[8].
3. Click "New Issue"
4. Fill out the form to ask a question or make a request. No need to assign a person, milestone, or label.
5. Click "Submit New Issue".

Please include the following information in your request:

- Name - Create a name for the data element or enumeration you feel best describes the term.
- Definition - Write a comprehensive definition of the data element and include references if necessary.
- Data Type - Include your recommended data type and maximum field length.
- Enumerations - If your recommended field is a pick-list, please include enumerations. Enumerations may also need a definition.
- Justification - Please provide a reason the data element is important to your energy efficiency program or market.
- Duplication - Review the dictionary thoroughly to ensure you are not duplicating an already existing data element or enumeration. Concepts can be expressed in a number of ways and rather than adding additional data elements or enumerations, we can use this recommendation to better define existing elements.

Once you have defined the dataset needed for your program and have identified all required fields, you are ready to proceed to the next step. Remember, this can be an iterative process. It is good to do due diligence in the planning process. However, even the best implementation plans may need to be modified as the program goes to market and a large number of homes start running through it.

Note: Schedule opportunities later in your implementation to check in on data requirements and adjust as needed.

---

[6] https://github.com
[7] https://github.com
[8] https://github.com/hpxmlwg/hpxml/issues/

Step 4: Procure or Modify Program Management Systems

Procurement of an HPXML-compliant program management system or modification of an existing system to be HPXML compliant is a critical juncture for every HPXML implementation. The cost and scope of this endeavor will be largely dictated by the business objective(s) established in steps one to three.

The following section will provide some best practice guidance for acquiring HPXML-compliant program management software and how to use the tools provided in this guide to assist in that process.

Build a Clear Scope

Clearly outlining your business objectives and defining the dataset you want to use are two large steps toward building an accurate scope of work to inform your procurement process. Within your business objectives, you will want to define how you want the data to work for your program. Here are some examples of items to discuss:

- Are you using the system to qualify projects for incentives or financing?
- What kind of quality assurance/quality management do you want to do? How much QA automation do you want to include in the system? What data does your implementation team need to review on the project and at what stages? Programs may want to reference guidance from the Home Performance with ENERGY STAR® Sponsor Guide (v1.5). Section 6 and Appendix F provide a description of quality management system approach, how it uses data to determine compliance and measure performance, and directions for developing a QMS plan.
- What is the workflow of your program?
- What feedback systems do you want for both contractors and participating customers?
- What type of program reporting are you doing?
- Are you exporting data to third parties? If so, what are the specifications of their systems?

If your program needs to issue a Request for Proposals (RFP), there are some additional best practices to consider:

- Include all of your program's incentive or financing rules. This will help the developers understand what pieces of information are most important to you and what you will be reviewing.
- Consider including narrative examples of what you are trying to accomplish with this system's functionality. Technical specifications are important, but adding commentary on how you want to use it is equally helpful.
- Ask other programs to share their RFP scopes of work. With each new program adoption of HPXML, RFPs are getting more consistent and comprehensive. The HPXML Working Group can help your procurement team identify recent RFPs and connect you with a program contact.

Leverage HPXML Experience

In procuring a software provider, it is recommended that you consider their HPXML experience, including prior implementations of HPXML and participation in the working group. Many software vendors have already built in HPXML compliance or have been an active part of the HPXML development process. This experience should help speed up implementation and may reduce costs.

There are several program administrators that participate in the HPXML working group. If you have questions, or would like to learn about their experiences, email your request to hpxml@homeperformance.org or visit http://www.hpxmlonline.com.

Plan for Testing and HPXML Coordination Support

In your scope and timeline, leave time for testing and provide resources for technical support in setup and integration. The amount of coordination and testing will vary depending on your data selection and use cases. Many of the

programs that were the first to implement HPXML did not leave enough time for testing, which can be a couple of months depending on the scope of the project. If not accounted for in planning, this can put the project behind schedule or place a lot of pressure on the implementation team.

See the testing section for further guidance.

Step 5: Design a Data Validation Process

Designing a good system for data validation that automatically checks all submitted data is critical for ensuring high-quality data, maintaining contractor satisfaction, and streamlining quality assurance activities. The good news is that most of this work will be completed by your software vendor. However, you will want to undergo a thorough scoping exercise with your vendor to identify what types of validation checks you want your program software to complete.

For example, a basic validation check ensures that all data is present and in the right format. More advanced validation checks can ensure that data falls within an acceptable range for program compliance purposes or to guide quality assurance. You may be able to automate quality assurance review on health and safety results to ensure all standards are being met. If not met, a user can be warned and intervene on the project.

If done correctly, a good validation system can speed up your process and significantly reduce cost. If done too hastily, it can increase frustration within your contractor base or yield lower-quality data.

Your data validation systems should align with your business objectives. For example, if you are using HPXML for a rebate program, you will want to validate that:

- All files are in the correct HPXML format
- Data points required for rebate eligibility screening are collected
- All rebate eligibility rules are met
- Health and safety standards have been followed

Your program will probably have specific requirements that you are trying to validate against. As you develop your validation system share the requirements with prospective third-party software vendors. These software vendors will likely include these same requirements in their software validation protocols to warn contractors when a requirement has not been met. This is an important benefit of using a national data standard.

Once validation rules have been set, it is equally important to ensure that the user's experience is optimized to minimize frustration and clearly communicate validation errors. If a contractor is not receiving clear validation error messaging and cannot resolve the issue during the upload process, this can lead to a large number of phone calls and a higher technical assistance requirement resulting in greater cost to the program. If the process is not managed properly, you can burn out your users and create contractor dissatisfaction. This, of course, is not an observation exclusive to HPXML, but a best practice in any software implementation in general.

HPXML Toolbox

To assist in validation, the National Renewable Energy Lanoratory (NREL) has created an HPXML Toolbox[9] that includes an HPXML schema validator, dataset validator, data dictionary, a collapsable tree view of HPXML, and a web API that can be incorportated into software workflows. The tool is useful to both software developers to test their implementation against the known datasets. It is also helpful for program administrators to see what data is in an HPXML file and what additional data would be required to meet any of the use cases.

[9] https://hpxml.nrel.gov/

Phasing of Validation

As you prepare your rollout schedule, consider implementing a phased validation system roll out. If you clearly define the scope of each phase and roll it out following a regimented schedule, you can greatly assist the market in adapting to the new system, while giving you the opportunity to improve the data quality and functionality of your system over time. For example, if you are running a whole home program with incentives or rebates, here are some phases to consider:

Phase 1 – File Validation and Minimum Data Requirements

In this phase, you will want to verify that the uploaded HPXML file is in the correct structure and minimum data points required are present and in the correct format. This will allow you to get to market quickly and begin to test your systems. You will still need a person to review the files to ensure that the data provided meets the technical requirements of your rebate program. For example, you will validate that all required insulation data points are present, but review the installed R-value to make sure that it meets your program requirements. You will also want to add a check to ensure that no health and safety problems exist.

Phase 2 – Advanced Data Validation and Automated QA

Over time, you can begin to layer in engineering assumptions that provide automated QA or guided QA. For example, you can add a validation check to see if installed measures are consistent with standard building practices. This way, if you receive a file that claims to have an attic with R-100, a QA advisor is triggered to review the project.

By using this type of validation system, you can significantly reduce your labor requirements for reviewing submitted files because you can focus your labor on probable errors or problem areas. In some cases, you may be able to get to auto-approval on a selected number of projects.

Note: Over time, you can add even more sophisticated systems. If you launch with a very complex validation system, there is a high likelihood that many of the initial project submissions will fail as the market is still adapting to the new program environment. If you take a phased approach, you can ease this tension and coordinate with the market to facilitate high-quality data transactions while reducing admin costs.

Step 6: Implement Testing Protocols and User Training

With every new HPXML program launch, there are a number of testing activities that need to occur to maximize the opportunity for success. You should not underestimate the need for testing, but assume at least two months of testing will be needed. Work with the software community to schedule each stage of testing and schedule time for your staff to focus on this activity. If you can set a clear plan for testing early on, and stick to your production schedule, this should allow you to get to market in a timely and organized manner.

There are four main testing roles, each of which requires a different group of testers:

HPXML File Testing

This is best completed between your program software provider and third-party software vendors that are importing data to your system. Ideally, your program software provider will supply third-party software vendors with a testing environment in which they can submit test files and receive direct validation error feedback. Activity will likely be between the IT experts from both your program software provider and the third-party software vendors.

Program Compliance Testing

Based on the validation protocols that you have set up, and in order to check program compliance, a group of testers that are intimately familiar with the program requirements will review files in a test environment. The goal is to see if files submitted by the third-party software vendor are being captured in your program management system and that the system is parsing the data out as expected. Usually, this is completed by someone from the implementation or QA team.

End-to-End Testing

Especially if you are using third-party modeling tools, it is important for testers from the program implementation team to run a series of test homes using each of the software systems as if they were a participating contractor. This is to check to see if data entered into each system is making it all the way to your program database as designed. In addition, you will be familiarizing yourself with the user experience and can identify potential problems before releasing to the contractor base.

User Testing

Once you have completed end-to-end testing, it is always good to identify a group of users (contractors or trade allies) who can try the system in a test environment or as a soft program launch. Using a limited number of contractors, you can identify any potential problems before full-scale release.

It is good to designate a program representative responsible for overseeing the testing process. This person should be equipped to make decisions regarding software requirements and functionality on behalf of the program. As things are discovered during the testing process, it will be important to determine what may need to be fixed before launch verses what can be addressed over time. This role will help ensure that the program requirements are being met, while keeping the project on track.

User Training and Contractor Roll Out

After testing is complete, you can launch your new HPXML-based systems. As in most implementation steps up to this point, it is extremely important to engage your contractor and trade ally networks to achieve success. When rolling out a new software environment, especially if multiple software products are being introduced into the market at the same time, programs should coordinate software training along with the software vendors. Each software vendor may have their own training practices, webinars, or other resources they prefer to use.

Fig. 3.1: HPXML Contractor Training for Arizona Public Service (Source: EnergySavvy)

The trainings should include a clear understanding of how to download and upload HPXML files. This includes making transparent to all users the data collection requirements that you have determined as a part of your HPXML implementation.

Being very transparent about this in training can significantly reduce the tech support required for your program implementation team and software vendors, while also increasing contractor satisfaction.

Step 7: Develop a Quality Management Plan

Once your HPXML program is launched, you can take advantage of the standardized data you have been collecting. This could be enhanced reporting, integration with real estate databases, or any of the other activities identified in the Overview section of this document.

To ensure success with these activities, the collected data must accurately reflect the characteristics of the homes participating in the program. It is important to develop a data quality management plan that allows for continuous evaluation of the program to maximize accuracy.

In addition to standard quality assurance activities, the following practices can be integrated into a program plan:

Complete a Regular Data Review

Consider scheduling a regular data review at least once a year, or more frequently, if possible. This is a basic audit of your data to verify that you are capturing all of the required data, to screen for data anomalies, and to ensure your validation systems are working as expected. After these meetings, consider communicating findings to all trade allies and software vendors to facilitate continuous improvement.

QA Trend Analysis

Consider working with your program software vendor to add trend analysis capability into your reporting systems. Even with an automated validation system, it is possible to game the data and report only what will pass the validation protocol. Your project-level QA strategy should provide some verification activities to reduce the potential of this. However, it is often not cost-efficient to schedule third-party verification of every home. Looking for data trends can assist in identifying potential issues. For example, if a large portion of jobs are reporting an installed condition that is almost identical to manufacturer specifications, you could focus QA on these jobs to further verify data accuracy.

Integrated Measurement and Evaluation

With a rise in automated billing analysis software, automated metering and connected thermostats, there is an emerging ability to verify performance data in a more dynamic fashion. This can help quickly identify potential issues with data quality and help focus on continuous process improvements.

Regardless of the strategy you choose for your program, it is important to complete a comprehensive evaluation of your data before you begin to export data to market, especially if you intend to use the data to guide financial investments.

Summary

This concludes the Program Administrator section of the Implementation Guide. The next section provides similar guidance for software developers.

CHAPTER 4

Software Developer Guide

Introduction

This section of the Implementation Guide is designed to outline the technical details of implementing HPXML. HPXML is an expansive standard data format based on extensible markup language (XML) and maintained by the Building Performance Institute's HPXML working group. For more information on HPXML, visit http://www.hpxmlonline.com.

The HPXML format is defined by a set of XML Schema (XSD) documents that outline all the acceptable data elements, their structure, and relation to one another. The schema itself is very flexible. Almost none of the elements are required, which allows for any level of detail in home performance data to be transmitted in the format. This level of flexibility is very useful but it also can be dangerous for the software developer. Two developers could each create an implementation that would represent their data in valid HPXML, but would be divergent. The purpose of this guide is to document the assumptions that are not codified in the schema that are necessary to write and interpret HPXML documents across platforms.

Standard Datasets

In order to better utilize the standard and have consistency across the HPXML workging group has created *Standard HPXML Datasets* that define what data elements are required for a pre-defined set of typical use cases. While most programs will have their own specific data needs to be supported by HPXML these can serve a useful starting point to enable interoperability between existing systems already in the marketplace.

The majority of the assumptions and recommendations in this Guide come from the set of pilot implementations in the *Audit-Upgrade Standard Dataset*. This guide primarily documents lessons learned from those pilots. As other datasets are created, this guide will continue to be updated.

HPXML Working Group

One of the best resources you will have available to you as you develop to HPXML is the HPXML Working Group. Many of the current developers are working group members. They have a lot of experience in getting HPXML working and can help you avoid costly pitfalls.

To join the working group, send an email to hpxml@homeperformance.org.

GitHub

The HPXML schemas, example files, and this guide are all maintained on GitHub[10]. You will always be able to find the latest version of the schemas there. Additionally as the schemas evolve over time you can be connected to and influence that process.

Versioning

The HPXML schemas follow the Semantic Versioning v2.0[11] specification. The version numbers follow a pattern of *Major*. *Minor*. *Patch* (i.e. 2.2.0).

The first element of `HPXML.xsd` will indicate the version of the schema via the `version` attribute. Note that when referencing a version, the patch number is assumed to be zero if it is omitted.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://hpxmlonline.com/
↪2014/6"
 targetNamespace="http://hpxmlonline.com/2014/6" elementFormDefault="qualified"␣
↪version="2.2">
```

Additionally, starting with version 2.0, the version of the schema used is required in the `schemaVersion` attribute on the root element of every document.

```
<HPXML xmlns="http://hpxmlonline.com/2014/6" schemaVersion="2.2">
```

Major

The major version number is incremented when the schemas are changed in a manner which is incompatible with previous versions. Examples of changes which necessitate a major version change include:

- Renaming elements
- Removing elements
- Moving elements
- Removing enumerations

A different xml namespace is used for each major revision. Starting with version 2.0, the namespaces follow the pattern where the year and month are when the major version number was changed.

```
http://hpxmlonline.com/[Year]/[Month]
```

Minor

The minor version number is incremented when the schemas are changed in a manner that is backwards compatible with previous versions that share the same *Major* version. Backwards compatible in the context of HPXML means that given the schema changes, a document created in a previous version of the schema will also validate against the new schema. Example of changes which necessitate a minor version change include:

[10] https://github.com/hpxmlwg/hpxml
[11] http://semver.org/

- Adding elements
- Adding enumerations
- Changing the annotation in the schema for an element

Warning: Based on the definition of backwards compatibility, adding enumerations is a non-breaking change. However, it can be breaking for receiving systems if they are not expecting the change. The working group will provide warning when new enumerations are added so that receivers have an opportunity to respond by updating support.

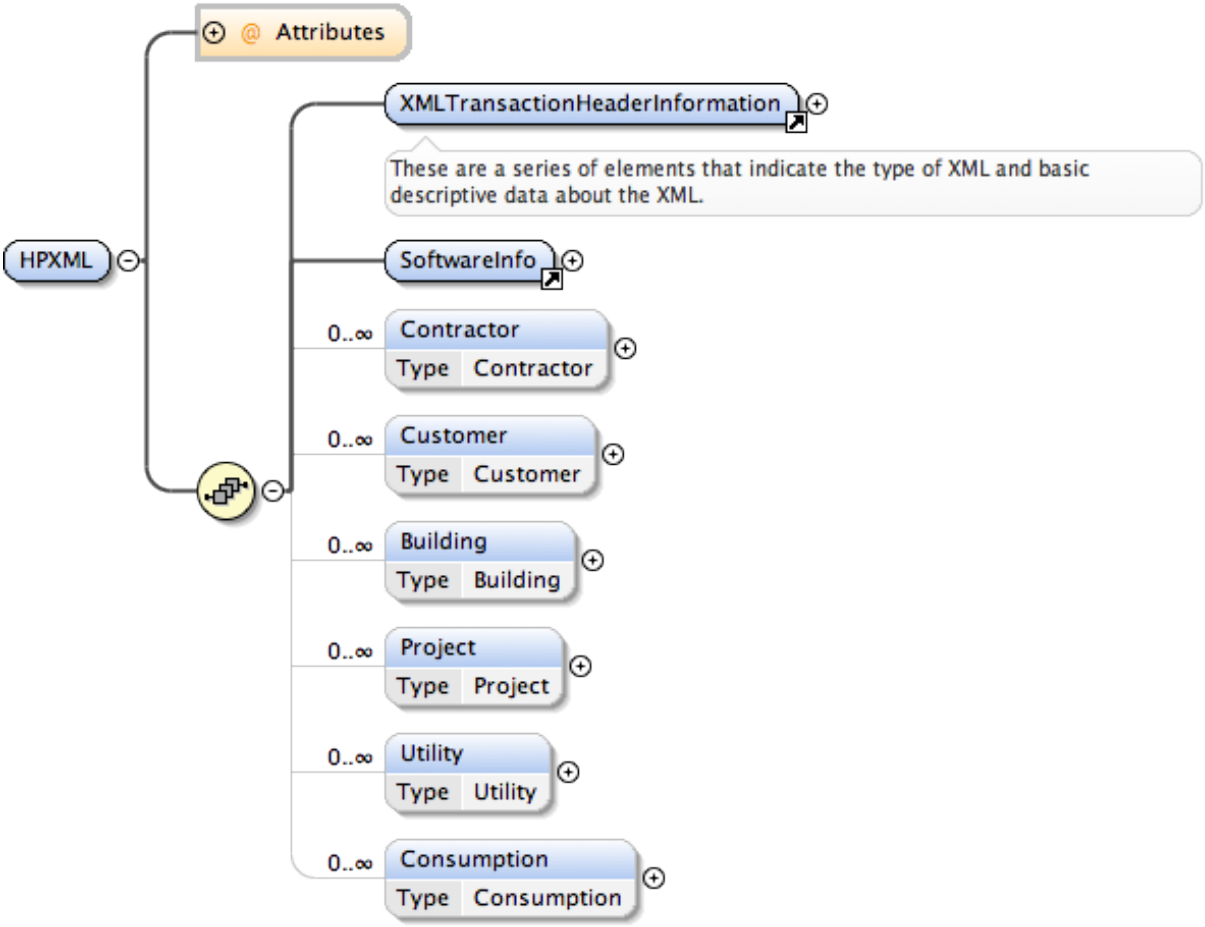
Patch

A patch version number is incremented when a backwards compatible change (as described in *Minor*) is made to address a bug.

Document Structure

Table of Contents

Top Level Nodes



Each of the top level nodes described below with the exception of *XMLTransactionHeaderInformation* and *SoftwareInfo* represent a high-level block of information about a building or project that can be related to other nodes to describe useful information about a building, the people and businesses who interact with the building, and actions taken on the building. The relationships between the top level nodes are defined with *XML Element References* that can be used, for example, to associate a *Building* with a *Project*, *Consumption* with a *Building*, or a *Contractor* with a *Project*.

The schema itself does not enforce the particular constraints for *Standard HPXML Datasets*, but rather provides a container for all the relevant components and a referencing scheme to relate them.

XMLTransactionHeaderInformation

The `XMLTransactionHeaderInformation` element meta data about the HPXML file.

```
<XMLTransactionHeaderInformation>
  <XMLType>audit</XMLType>
  <XMLGeneratedBy>Housesoft 1.0</XMLGeneratedBy>
  <CreatedDateAndTime>2014-09-02T17:32:12Z</CreatedDateAndTime>
  <Transaction>create</Transaction>
</XMLTransactionHeaderInformation>
```

`XMLType` is generally unused and may be deprecated in the future.

`XMLGeneratedBy` is often used to transmit the name of the software that generated the HPXML file. It may also be deprecated in the future due to its redundancy with *SoftwareInfo*.

`CreatedDateAndTime` is the date and time the file was generated in the ISO 8601 format.

`Transaction` describes whether this is a new document or an update to a previous one.

SoftwareInfo

`SoftwareInfo` provides a place to transmit information about the software used to generate the HPXML.

```
<SoftwareInfo>
  <SoftwareProgramUsed>WOPR</SoftwareProgramUsed>
  <SoftwareProgramVersion>1.0</SoftwareProgramVersion>
</SoftwareInfo>
```

Contractor

The `Contractor` node describes a business that the customer works with to do an audit or upgrade to their building.

Customer

A customer is the owner, tenant, or some other person who has a vested interest in the house being described and worked on. This node is a place to describe that person, their contact information, and their relation to the building.

Building

The `Building` node describes the physical characteristics of a building at a point in time - past, present, or future.

Project

The `Project` node describes work that has been completed or is to be completed to a *Building*. The measures described can have references pointing to specific components on the building and what was changed between the pre- and post-upgrade states and associated costs.

Utility

The `Utility` node represents a utility company.

Consumption

The `Consumption` node stores and represents the energy and/or water use of a building. It can contain high resolution electric smart meter data, information on fuel oil, or more typically, monthly gas or electric bills.

Extension Elements

Because it is impossible to foresee every possible data point that will ever need to be collected and transmitted about a house or upgrade, most elements in HPXML contain an `extension` element containing an `<xs:any>` designation. That allows any element from any namespace to be inserted. This is to facilitate transfer of data elements not available in the standard.

```
<extension>
   <QuantityWoodChucked>as much wood as a wood chuck could chuck</QuantityWoodChucked>
</extension>
```

Warning: Please exercise extreme caution and discretion when you consider implementing `extension` elements. Typically, the temptation to use them happens when a difference arises between the way your software and/or home performance program represents a certain data field and the way HPXML represents it. It is crucial in these cases to either map your data into HPXML or change the way you represent it internally to conform to the HPXML standard. **If each software vendor and home performance program extends HPXML in non-standard ways, the value proposition of the standard is nullified.**

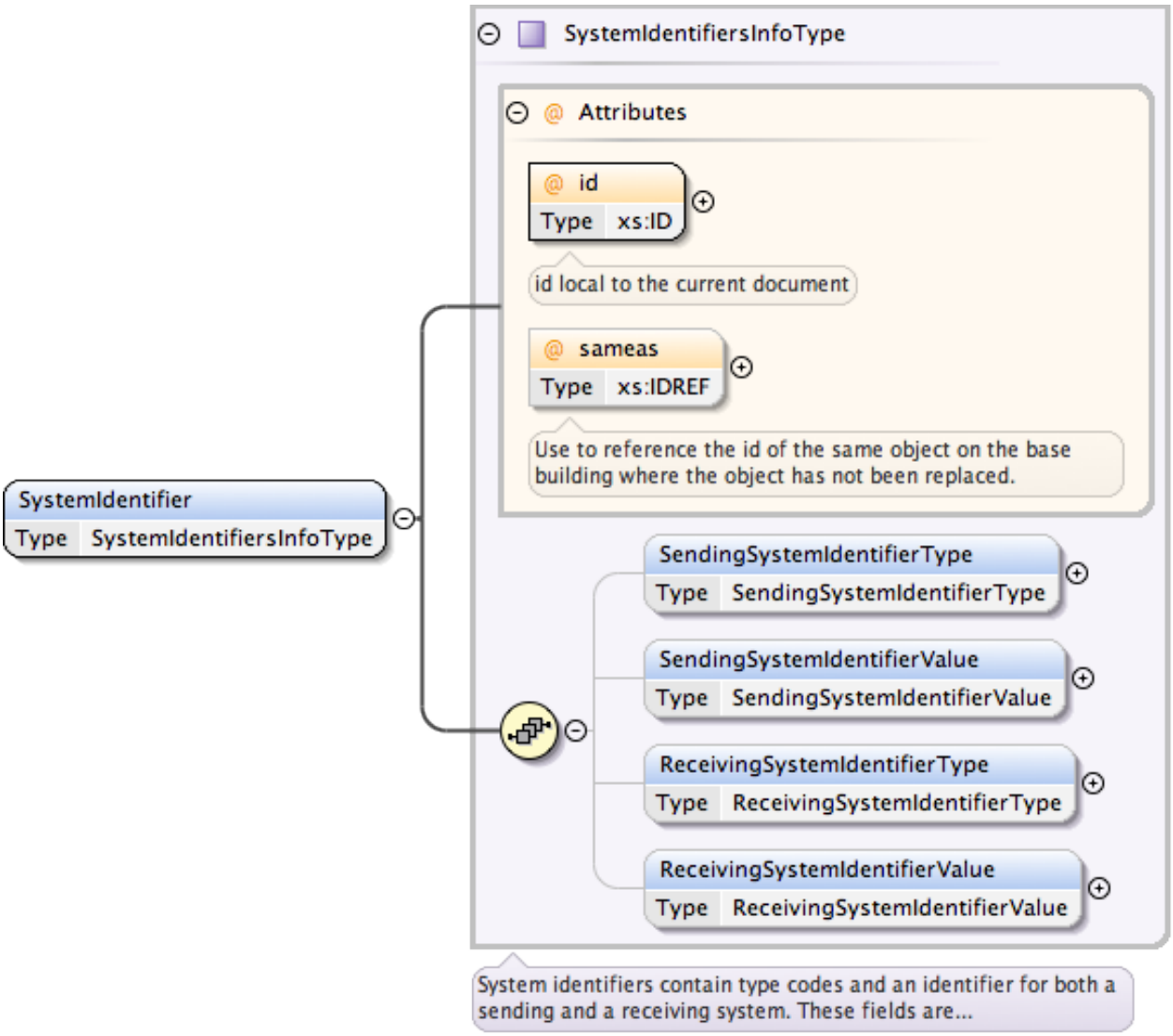
If there is no possible way to map your data into existing HPXML data fields, please contact the HPXML working group before implementing an extension. The working group would prefer to extend the standard for the benefit of everyone and avoid the use of extensions wherever possible.

Usually, you will not be the only one with the need for a particular element that was overlooked in the standard. By participating in the working group and lobbying for the elements you need you can enhance the value of HPXML for all parties.

XML Element References

XML documents are inherently hierarchical. This works out well for describing many things regarding houses, and home performance in general. For instance, an AFUE is a property of a furnace, which is part of an HVAC system on a building. Many other relationships are not as strictly hierarchical, however. An example of this would be the relationship between two furnaces in different snapshots of the building where one furnace replaced the other. In that case, there is no clear parent-child relationship.

In HPXML we have the ability and requirement to identify many elements with unique identifiers. This facilitates referential relationships between elements throughout the document. This is done with the `SystemIdentifier` element.



Intra-Document References

The most common reference you will make is a reference that is internal to the document.

ID and IDREF

Each element in the HPXML document that could need to be referenced has a required `SystemIdentifier` sub-element which in turn has a required `id` attribute. That attribute has the xs:ID data type[12] in XML Schema. This provides a unique identifier for that element within the document. This is similar to a primary key in a relational database.

When an element needs to reference another element it uses an xs:IDREF datatype[13], which ensures that the id referenced exists somewhere within the document. One simple example of this is how a window can reference the wall to which it is attached.

[12] http://www.w3.org/TR/2012/REC-xmlschema11-2-20120405/datatypes.html#ID
[13] http://www.w3.org/TR/2012/REC-xmlschema11-2-20120405/datatypes.html#IDREF

```
<?xml version="1.0" encoding="UTF-8"?>
<HPXML xmlns="http://hpxmlonline.com/2014/6" schemaVersion="2.2">
    <XMLTransactionHeaderInformation>
        <XMLType></XMLType>
        <XMLGeneratedBy></XMLGeneratedBy>
        <CreatedDateAndTime>2014-09-03T16:06:24Z</CreatedDateAndTime>
        <Transaction>create</Transaction>
    </XMLTransactionHeaderInformation>
    <SoftwareInfo/>
    <Building>
        <BuildingID id="bldg1"/>
        <ProjectStatus>
            <EventType>audit</EventType>
        </ProjectStatus>
        <BuildingDetails>
            <Enclosure>
                <Walls>
                    <Wall>
                        <SystemIdentifier id="wall1"/>
                    </Wall>
                </Walls>
                <Windows>
                    <Window>
                        <SystemIdentifier id="window1"/>
                        <AttachedToWall idref="wall1"/>
                    </Window>
                </Windows>
            </Enclosure>
        </BuildingDetails>
    </Building>
</HPXML>
```

sameas

The `sameas` attribute is a special IDREF that is used most predominantly in the case of the *Audit-Upgrade*. It serves to link components of buildings between the pre- and post-upgrade `Building` nodes. Each `Building` node is a full description of the building and measures only affect some of the components. For components that do not change it is useful to have a way to indicate that they are the same item.

```
<?xml version="1.0" encoding="UTF-8"?>
<HPXML xmlns="http://hpxmlonline.com/2014/6" schemaVersion="2.2">
    <XMLTransactionHeaderInformation>
        <XMLType></XMLType>
        <XMLGeneratedBy></XMLGeneratedBy>
        <CreatedDateAndTime>2014-09-03T16:06:24Z</CreatedDateAndTime>
        <Transaction>create</Transaction>
    </XMLTransactionHeaderInformation>
    <SoftwareInfo/>
    <Building>
        <BuildingID id="bldg1"/>
        <ProjectStatus>
            <EventType>audit</EventType>
        </ProjectStatus>
        <BuildingDetails>
            <Enclosure>
                <Walls>
```

```
                    <Wall>
                        <SystemIdentifier id="wall1"/>
                    </Wall>
                </Walls>
            </Enclosure>
        </BuildingDetails>
    </Building>
    <Building>
        <BuildingID id="bldg1post"/>
        <ProjectStatus>
            <EventType>proposed workscope</EventType>
        </ProjectStatus>
        <BuildingDetails>
            <Enclosure>
                <Walls>
                    <Wall>
                        <SystemIdentifier id="wall1post" sameas="wall1"/>
                    </Wall>
                </Walls>
            </Enclosure>
        </BuildingDetails>
    </Building>
</HPXML>
```

Inter-Document References

The `SystemIdentifier` element also has sub-elements that facilitate specifying identifiers for both a sending and receiving system. This way a document could identify components based on where it is coming from and going to. This feature currently isn't used much in lieu of the much simplified *Intra-Document References*.

Development Tools

HPXML Toolbox

To facilitate the adoption of the HPXML standard use cases, NREL has developed the HPXML Toolbox[14]. Features of the toolbox include a data dictionary/schema explorer tool as well as a validator.

Data Dictionary and Schema Explorer

One of the challenges of HPXML is understanding and communicating the structure of the schema, how data elements are organized, and how they are to be interpreted. Up until recently the best way to learn those details was to inspect the schema files[15]. The online data dictionary[16] is a searchable website that documents HPXML structure, element interpretations, the standard datasets to which an element belongs, and how any element in HPXML relates to the other data standards such as BEDES and RESO.

[14] https://hpxml.nrel.gov
[15] https://github.com/hpxmlwg/hpxml/tree/master/schemas
[16] https://hpxml.nrel.gov/datadictionary

Validator

The toolbox provides a schema validator plus standard dataset validator. Submitting an HPXML file will perform the following steps:

1. Validation of the file against HPXML schema and determination of schema version.
2. Validation against each of the standard use cases including output describing any missing elements required to meet a particular use case.
3. Display of HPXML file in collapsable tree form.

The validation functionality is available as a website[17] for testing and viewing individual files, or as a web service API[18] that allows the validation to be called as part of a software workflow.

Data Selection Spreadsheet

A *HPXML Data Selection Tool* has been developed to help program administrators select the data points they are requiring to collect for specific use cases. The data selection tool is a spreadsheet with the required and optional data points selected by working group members for specific standard datasets, including Audit, Upgrade, Home Energy Score, and Home Performance Certificate. A detailed desccription of these use cases and the corresponding standard HPXML datasets can be found in section *Standard HPXML Datasets*.

In the spreadsheeet, data points required by a dataset will be in black color instead of light grey. Elements that are optional are highlighted in red.

The standard datasets provide a complete list of data elements required to either provide or process.

Standard HPXML Datasets

The standard datasets each specify a specific set of elements required for some typical use cases of HPXML. These have been developed by the working group as a consensus. Each standard dataset is described in more detail at *Standard HPXML Datasets*, and the enumerated list of elements for each dataset can be found at the online data dictionary[19] or *HPXML Data Selection Tool*. This section of the documentation is meant to describe the structure, organization, and technical details of each dataset.

Audit-Upgrade

Contents

[17] https://hpxml.nrel.gov/validator/
[18] https://hpxml.nrel.gov/api/
[19] https://hpxml.nrel.gov/datadictionary

- *Project*
  - *Energy Savings*
  - *Measures*

The audit and upgrade datasets cover two scenarios:

1. A baseline building with a proposed work scope (see *Audit*)
2. A baseline building with a completed work scope (see *Upgrade*)

Both datasets describe a pre- and post-upgrade building and actions (or upgrades) that occur during both states. To accurately describe these scenarios, the HPXML document needs to have the *top level nodes* as described below.

All XML examples are taken from the `examples/audit.xml` document. It contains all of the required fields defined in the Audit dataset.

Contractor

The `Contractor` elements should identify the name of the contractor (the person who is proposing and or completing the work), company and the email address.

```
    <Contractor>
        <ContractorDetails>
            <SystemIdentifier id="contractor1"/>
            <BusinessInfo>
                <SystemIdentifier id="business1"/>
                <BusinessName>ACME Home Performance Company</BusinessName>
                <BusinessContact>
                    <Person>
                        <SystemIdentifier id="contractorperson1"/>
                        <Name>
                            <FirstName>John</FirstName>
                            <LastName>Doe</LastName>
                        </Name>
                        <Email>
                            <EmailAddress>john.doe@hpxmlonline.com</EmailAddress>
                        </Email>
                    </Person>
                </BusinessContact>
            </BusinessInfo>
        </ContractorDetails>
    </Contractor>
```

Customer

The customer is the homeowner or resident. The `Customer` element should include the customer's name and phone number.

```
<Customer>
    <CustomerDetails>
        <Person>
            <SystemIdentifier id="customer1"/>
            <Name>
                <FirstName>Jane</FirstName>
                <LastName>Customer</LastName>
```

```
            </Name>
            <Telephone>
                <TelephoneNumber>555-555-5555</TelephoneNumber>
            </Telephone>
        </Person>
    </CustomerDetails>
</Customer>
```

Building

There are two `Building` nodes in the Audit and Upgrade document: pre- and post-upgrade. Each document includes a full description the house. The pre-upgrade condition of the house in the Audit dataset is the house prior to any work being completed on it (existing condition). The post-upgrade condition of the house describes the proposed work scope, usually completed by the contractor.

The pre-upgrade condition of the house in the Upgrade dataset is the state of the house prior to the completion of any work (existing condition). The post-upgrade condition of the house state describes the building after work has been completed.

Pre-upgrade

The pre-upgrade `Building` element comes first in the document. It describes the initial state of the building (prior to work being completed). It should have a `ProjectStatus/EventType` of `audit`.

```
<Building>
    <BuildingID id="bldg1"/>
    <ProjectStatus>
        <EventType>audit</EventType>
    </ProjectStatus>
</Building>
```

Many items within the building require a unique `SystemIdentifier` element. The `id` attribute is used to specify this ID within the document (see *ID and IDREF*).

For example, the water heater in the pre-upgrade building has an id of `dhw1`.

```
<WaterHeatingSystem>
    <SystemIdentifier id="dhw1"/>
    <FuelType>natural gas</FuelType>
    <WaterHeaterType>storage water heater</WaterHeaterType>
    <Location>conditioned space</Location>
    <CombustionVentingSystem idref="combvent1"/>
</WaterHeatingSystem>
```

Post-upgrade

The post-upgrade `Building` element appears second in the document. It describes the "after" state of the building. In the audit dataset, that means the *proposed* improvements to the building (e.g., what the building would look like after proposed work has been completed). In the upgrade dataset, that means the *actual* condition of the building after the work is completed. The `ProjectStatus/EventType` element has a different value depending on the scenario:

Table 4.1: Post-upgrade Event Types

Use Case	Event Type
Audit	proposed workscope
Upgrade	job completion testing/final inspection

The post-upgrade building is mostly a duplicate of the pre-upgrade building; components of the building that do not change remain the same. However, each component in the post-upgrade building needs a unique identifier that is different from the unique identifier in the pre-upgrade building. The `sameas` attribute of the `SystemIdentifier` element is used to link identical elements in the pre- and post-upgrade buildings (see *sameas*).

Going back to the water heater example, the water heater in the post-upgrade building has a different `id` than the identical water heater in the pre-upgrade building, but it has a `sameas` attribute to link it back to the pre-upgrade water heater and indicate it is indeed the same equipment.

```
<WaterHeatingSystem>
    <SystemIdentifier id="dhw1p" sameas="dhw1"/>
    <FuelType>natural gas</FuelType>
    <WaterHeaterType>storage water heater</WaterHeaterType>
    <Location>conditioned space</Location>
    <CombustionVentingSystem idref="combvent1"/>
</WaterHeatingSystem>
```

Note: When a measure changes a component between a pre- and post-upgrade building, the `SystemIdentifier/@sameas` attribute is omitted because the measure references the relationship between components.

Project

In this paradigm, the *Pre-upgrade* and *Post-upgrade* building elements describe the state of the building at points in time. The `Project` element describes what was done, or is planning to be done, to the building to get from one scenario to another.

The `ProjectSystemIdentifiers` are used to reference the pre- and post- building ids. The redundant `BuildingID` element should reference the post- building.

```
<Project>
    <BuildingID id="bldg1p"/>
    <ProjectDetails>
        <ProjectSystemIdentifiers id="bldg1"/>
        <ProjectSystemIdentifiers id="bldg1p"/>
        <ProjectStatus>
            <EventType>proposed workscope</EventType>
        </ProjectStatus>
        <EnergySavingsInfo>
        </EnergySavingsInfo>
        <Measures>
        </Measures>
    </ProjectDetails>
</Project>
```

Energy Savings

`EnergySavingsInfo` is used to transfer either or both the estimated or measured energy use and savings achieved

in an upgrade.

```
<EnergySavingsInfo>
    <EnergySavingsType>estimated</EnergySavingsType>
    <FuelSavings>
        <Fuel>electricity</Fuel>
        <Units>kWh</Units>
        <TotalSavings>3000</TotalSavings>
        <TotalDollarSavings>55</TotalDollarSavings>
        <PctReduction>0.1</PctReduction>
    </FuelSavings>
    <FuelSavings>
        <Fuel>natural gas</Fuel>
        <Units>therms</Units>
        <TotalSavings>100</TotalSavings>
        <TotalDollarSavings>123</TotalDollarSavings>
        <PctReduction>0.3</PctReduction>
    </FuelSavings>
    <AnnualPercentReduction>0.25</AnnualPercentReduction><!-- 25% -->
</EnergySavingsInfo>
```

Note: All percentages are expressed in the form of fractions (i.e., 10% => 0.1)

Measures

The `Measure` element describes work completed for a job. Each measure references one or more replaced components in the pre-upgrade building and one or more (usually one) installed components in the post-upgrade building.

In cases where a measure was installed without replacing an existing measure or component, the `ReplacedComponent` can be omitted. Similarly, if a component was removed and nothing was installed in its place, `InstalledComponent` would be omitted. The measure cost is also included in these files.

From the example file below, this measure replaces this furnace in the pre-upgrade building with this one.

```
<Measure>
    <MeasureSystemIdentifiers>
        <SystemIdentifiersInfo id="furnacereplacement"/>
    </MeasureSystemIdentifiers>
    <Cost>3000</Cost>
    <ReplacedComponents>
        <ReplacedComponent id="htgsys1"/>
    </ReplacedComponents>
    <InstalledComponent id="htgsys1p"/>
</Measure>
```

```
<HeatingSystem>
    <SystemIdentifier id="htgsys1"/>
    <UnitLocation>basement - conditioned</UnitLocation>
    <CombustionVentingSystem idref="combvent1"/>
    <HeatingSystemType>
        <Furnace/>
    </HeatingSystemType>
    <HeatingSystemFuel>natural gas</HeatingSystemFuel>
    <AnnualHeatingEfficiency>
        <Units>AFUE</Units>
```

```
        <Value>0.80</Value>
    </AnnualHeatingEfficiency>
    <FractionHeatLoadServed>0.7</FractionHeatLoadServed>
</HeatingSystem>
```

```
<HeatingSystem>
    <SystemIdentifier id="htgsys1p"/>
    <UnitLocation>basement - conditioned</UnitLocation>
    <CombustionVentingSystem idref="combvent1p"/>
    <HeatingSystemType>
        <Furnace/>
    </HeatingSystemType>
    <HeatingSystemFuel>natural gas</HeatingSystemFuel>
    <AnnualHeatingEfficiency>
        <Units>AFUE</Units>
        <Value>0.92</Value>
    </AnnualHeatingEfficiency>
    <FractionHeatLoadServed>0.7</FractionHeatLoadServed>
</HeatingSystem>
```

Home Energy Score

DOE's Home Energy Score rates a home's energy performance on a scale of 1 (least efficient) to 10 (most efficient). The score is determined using the assessed characteristics of the building that are either entered into a web interface by a qualified assessor or submitted via software through a SOAP API.

The API requires data inputs to be submitted in terms of the data model set forth by Home Energy Score. Therefore, any users of the Home Energy Score API must translate their data into the appropriate location and representation in the Home Energy Score input array.

The Home Energy Score API provides the capability for an HPXML import. This receives an HPXML file as input and translates the user-specified *Building* element (whether pre- or post-upgrade) into corresponding Home Energy Score inputs and populates the Home Energy Score input array.

By using this import capability, software developers can leverage their investment in HPXML to provide Home Energy Score functionality at a minimum incremental development cost.

See *Home Energy Score* for additional information.

Home Energy Score API

For more information on how to use the Home Energy Score API, see the API documentation[20]. The API method, `submit_hpxml_inputs` provides the HPXML import capability.

Data elements required and translation details

The *HPXML Data Selection Tool* includes Home Energy Score. It represents the minimum required data fields for successful import. It is a good starting point.

The HPXML import into Home Energy Score can accept a larger variety of data elements. The details of the translation and required HPXML elements are documented separately[21].

[20] https://developers.buildingsapi.lbl.gov/hescore
[21] http://hescore-hpxml.readthedocs.org/

Additionally, The translator has been released open source on GitHub[22]. Example HPXML files are available in that repository.

Home Performance Certificate (BPI-2101)

BPI-2101-S-2013 Standard Requirements for a Certificate of Completion for Residential Energy Efficiency Upgrades ("Home Performance Certificate") is a BPI standard that identifies a subset of HPXML data elements for certificates that document the completion of an energy efficiency project, either whole-house or individual measures (See *Home Performance Certificate*).

The Home Performance Certificate standard dataset is the most complex use case because it contains each of the following *Building* nodes.

- Pre-upgrade
- Proposed workscope
- Post-upgrade

Two *Project* nodes are included as well that describe the proposed project and the completed project. It can be thought of as a combination of the *Audit and Upgrade* use cases.

Finally, *Consumption* and *Utility* nodes are also included to provide a utility billing history.

See the `examples/bpi2101.xml` document for more details on what data elements are included.

[22] https://github.com/NREL/hescore-hpxml

CHAPTER 5

Appendix A: HPXML Tools and Resources

A number of tool and resources exist to help software developers, programs, and other stakeholders understand the value of HPXML and how to implement the standard.

HPXML Website

The Home Performance Coalition hosts a website that has information, tools, and resources on HPXML and its business cases. The website is: http://www.hpxmlonline.com.

HPXML Implementation Guide

The HPXML Implementation Guide[23] provides detailed information to programs and software developers on how to implement HPXML.

HPXML Data Selection Tool

The *HPXML Data Selection Tool* was created to assist program administrators in identifying the required data associated with HPXML use cases. The tool is a Microsoft Excel worksheet that can be downloaded from the HPXML Implementation Guide website. Users have the option of selecting one or more use cases from a dropdown menu. Once the use case(s) have been selected, the list of required fields with HPXML path will populate the sheet.

[23] http://hpxml-guide.rtfd.org

HPXML Toolbox

The HPXML Toolbox[24] provides a set of tools for software developers to validate and inspect their HPXML formatted data. Users can upload their HPXML files and check these file against the HPXML schema and its four standard datasets. The validator returns specific error messages with feedback on missing elements. There is also a searchable tree view of HPXML data to assist with navigating and inspecting errors. Validation is available through a website or an API that can be incorporated into HPXML data transfer workflows.

There is also a data dictionary for HPXML included in the Toolbox that documents the HPXML schema, the standard datasets, links to other data standards such as BEDES and RESO, and allows for searching.

HEScore Translator

The Home Energy Score Translator generates HEScore inputs from HPXML files. The translator is incorporated into the HEScore API. By using this translator, software developers can leverage their investment in HPXML to provide HESCORE functionality with minimum development cost.

Home Upgrade Program Accelerator

In May 2015, the DOE launched the Better Buildings Home Upgrade Program Accelerator for residential energy efficiency program administrators. This Accelerator is designed to increase the overall effectiveness of programs by leveraging data management strategies that minimize costs. The Accelerator supports the use of HPXML as a tool for achieving its goals and is working with Accelerator partners to provide assistance with adopting HPXML.

Github

The working group uses Github[25] as a forum for tracking changes to the HPXML schemas (including debating and resolving issues). The HPXML schema documentation resides on GitHub as well as links to HPXML resources.

[24] https://hpxml.nrel.gov
[25] https://github.com/hpxmlwg/hpxml