

CSA GROUP™

STANDARDS RESEARCH

Computing Appliances—Real-World Active Mode Benchmarking

February 2022

Authors

Vojin Zivojnovic, Ph.D., AGGIOS, Inc.

Davorin Mista, AGGIOS, Inc.

Project Advisory Panel

Noah Horowitz, NRDC

Toby Lau, BC Hydro

Voytek Gretka, MH Morrison Hershfield

Maroof Siddiqui, CSA Group (Project Manager)

Helene Vaillancourt, CSA Group

Financial Support

This work was generously supported by Fortis BC Electric, BC Hydro, and Northwest Energy Efficiency Alliance.

Disclaimer

This work has been produced by AGGIOS, Inc. and is owned by Canadian Standards Association. It is designed to provide general information in regards to the subject matter covered. The views expressed in this publication are those of the authors. AGGIOS, Inc. and Canadian Standards Association are not responsible for any loss or damage which might occur as a result of your reliance or use of the content in this publication.

Table of Contents

Executive Summary 5

1 Introduction 6

2 Proposed Active Mode Benchmark 7

2.1 Benchmark Requirements 8

2.1.1 Representativeness / Relevance 8

2.1.2 Repeatability of Results 8

2.1.3 Reproducibility of Results 8

2.1.4 Reasonable Effort / Duration 8

2.1.5 Portability / Platform Independence 8

2.2 Emulating Real User Activity 8

2.3 Description of Content Test Applications 9

2.3.1 Webpage Features 9

2.3.2 Webpage Technologies 9

2.3.3 Content Developed for the Benchmark 10

2.4 Description of Execution Environment 11

2.4.1 Impact of Background Activity 12

2.4.2 Impact of Network Delays 13

2.4.3 Web Server Requirements 13

2.5 Web Browsing Benchmark 13

2.5.1 Benchmark Duration 13

2.5.2 Emulation of User Activity 14

2.5.3 Activities Not Covered 14

2.6 Video Streaming Benchmark 15

2.6.1 Benchmark Duration 15

2.7 Audio Streaming Benchmark 15

3 Testing Procedures 15

3.1 Test Objects 16

3.2 Browsers Tested 16

3.3 Testing Procedure 16

4 Validation of Results 18
4.1 Results Overview 18
4.3 Causes of Measurement Errors 18
4.4 Results Reproducibility 19
4.4.1 Execution Environment Reliability 19
4.4.2 Reasonable Effort/Duration 19
4.4.3 Portability 20
5 Conclusions and Next Steps 20
References 22

Executive Summary

Standardized energy testing procedures, like those defined by the Energy Star program for computers, are key tools. They help the industry, energy efficiency advocates, and regulators to assess the energy use of computing appliances and plan improvements. With the evolution of technology and the emergence of mobile phones and tablets, typical idle power demand of key computer components has been reduced significantly. Therefore, the Energy Star tests' key assumptions for computers no longer represent real-world use. It is like measuring the gas mileage of a car while coasting downhill. As such, it may mislead policymakers and consumers, and encourage manufacturers to invest in energy efficiency improvements that may not produce expected reductions in real-world energy use.

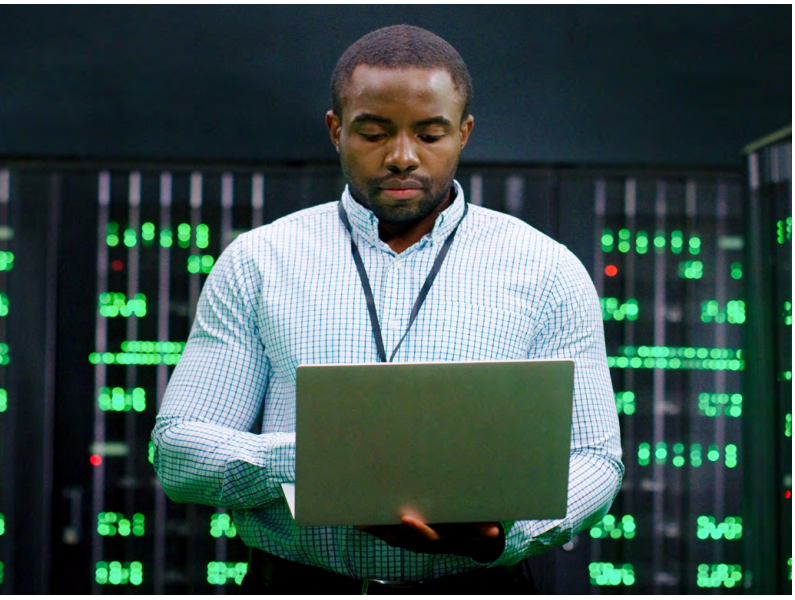
This research project defines a new test procedure to assess the real-world energy consumption of computers in their active state. The aim is to enable the industry, energy efficiency advocates, and regulators to better assess and leverage computers' energy savings potential cost-effectively in residential and commercial buildings.

The new Real-World Active Mode Benchmark proposed here defines the information content and scenarios for executing a standard set of "light active" tasks to measure the active mode power demand of computers while performing common activities. The information content is web based and resides on a local or remote web server that is part of the execution environment. User activity is emulated by using scripted control of the web browsers. The test procedure defines the sequence of steps conducted by laboratory technicians, including the setup of the device under test, benchmark execution, and energy measurements.

This Real-World Active Mode Benchmark can be the basis of an enhanced standardized test procedure for computing appliances representing real-world consumption better and meaningfully in a world where modern standby is broadly adopted in the market. The new test procedure is validated on a set of standard computer types and most popular browsers. The achieved results are reviewed for representativeness, repeatability, reproducibility, reasonability, and portability.

In this project the content is stored locally, on a web server machine within the same local area network (LAN) as the computer tested. Next, to improve benchmark representativeness, the web server machine should be accessed over the wide area network (WAN), i.e., the internet. Additionally, larger scale testing involving other types of desktops and laptops running under other operating systems, like those of Apple, should be included.

Modern computers have vastly improved idle mode power consumption, but disproportionally less than active mode consumption. Additionally, modern computers are 24/7 operating devices spending most of their duty cycle in active mode. Therefore, to avoid misguiding policymakers and the industry, it is necessary to retest modern computers and revise the Energy Start test procedure accordingly. The Active Mode Benchmark proposed here can guide future revisions of the Energy Star test procedure.



"Standardized energy testing procedures are key tools for industry, energy efficiency advocates, and regulators to assess computing appliances' energy use and plan improvements."

1 Introduction

Standardized energy testing procedures, like those defined by the Energy Star program for computers, are key tools for industry, energy efficiency advocates, and regulators to assess computing appliances' energy use and plan improvements. The current Energy Star v8 test procedure is largely based on the 2009 Energy Star v5 framework. At that time, the power demand of computing devices in idle state was sufficiently similar to that during active operation. For simplicity, idle state consumption was used as a proxy for assessing active mode energy use.

As mobile phones and tablets emerged, computing appliance manufacturers have started leveraging power savings techniques from mobile devices to reduce idle power demand. This technology evolution significantly reduced typical key computer components' idle power demand and made the key assumptions of the Energy Star tests for computers no longer representative of real-world use.

The real-world power demand of a typical desktop computer can be an order of magnitude higher than the Energy Star idle mode power demand when performing computer intensive operations like gaming and video editing. Even in low-intensity active mode such as web browsing and media playback, or in "real-world idle" mode with no user activity but applications loaded, windows open, and background processes running, computer power draw can be significantly higher (50% or more) than the Energy Star short idle mode [1].

In short, the current Energy Star test method for computers is like measuring the gas mileage of a car coasting downhill. It may mislead policymakers and consumers, as well as encourage manufacturers to invest in energy efficiency improvements that may not create expected reductions in real-world energy use.

This situation will be exacerbated by the computer industry's forthcoming adoption of "modern standby," a new mode allowing computers to reduce power to very low levels when idle. The broad deployment of this capability is driven partly by the California computer regulations published in 2016 and scheduled to take full effect in late 2021 [2]. By then, the Energy Star v8 test procedure could become largely obsolete. To continue the car analogy, the Energy Star v8 idle mode will measure a car's gasoline consumption when stopped at a traffic light, with the auto-stop feature engaged (gasoline engine turned off). Though an important energy saving capability, this operation mode does not represent overall energy use in a typical driving scenario.

While this project focuses on computers, the new infrastructure and content could be leveraged in the future to measure power demand under real-world conditions for other computing appliances capable of web browsing and video streaming, including game consoles, media players, set-top boxes, and smart TVs.

Once validated, this infrastructure and content could inform an enhanced test procedure for computing appliances more representative of real-world

consumption. That remains meaningful in a world where modern standby is broadly adopted in the market. This light Active Mode Benchmark could provide a draft test method for inclusion in Energy Star for computers and other computing appliance efficiency policies and standards. The new testing methodology can be advocated as standardized test methodology that could help enable more effective efficiency standards and labels, and support better customer purchasing choices.

Additional benefits include:

- Unified testing of active modes for a wider class of computing appliances providing web browsing and audio/video streaming functionality, such as game consoles and smart TVs, thereby potentially increasing the total savings potential.
- Simplified and accelerated voluntary and regulatory efforts based on unified horizontal standards for all computing appliances.

The feasibility of a testing infrastructure that allows execution of benchmarks without relying on compiled code for execution was demonstrated in an earlier study [3]. Content-driven benchmarks differ from traditional benchmarks. They do not need to be compiled for each operating system or CPU architecture used by computing devices under test. Content-driven benchmarks leverage the browser and media player that come by default on the device under test rather than compiled code snippets or applications. This benchmarking technique has been used successfully in the mobile phone industry to compare performance across mobile operating systems without requiring a software application to be compiled for each operating system.

This project is intended to create prototype website content that can be measured using the execution environment. The content shall be tested on the latest versions of Google Chrome and Mozilla Firefox. Furthermore, the execution environment developed in previous work [3] needs to be updated and improved to properly emulate user behaviour. The created content running on the updated execution environment needs to be evaluated by repeatability, reproducibility and representativeness testing and validation.

2 Proposed Active Mode Benchmark

This section describes the proposed methodology and test setup for executing a standard set of benchmark tasks to measure the active mode power demand of computing devices while performing common activities.

The benchmarks are not intended to cover high-intensity active mode for use cases such as gaming, video editing, or crypto-currency mining.

While the proposed benchmark focuses on computers, the infrastructure and content shall be structured to allow measuring the power demand under the same real-world conditions of other computing appliances capable of web browsing and video streaming, including game consoles, media players, set-top boxes, and smart TVs.

The benchmarks shall cover the following light-duty activity use cases:

- web browsing and web-based application use
- video streaming
- audio streaming

Given the prevalence of web-based and cloud-based software, these use cases represent most light-duty activity without requiring any platform specific software applications, thus allowing a side-by-side comparison of different types of devices. The benchmarks defined in this report are used for measuring the device's power draw while performing typical light use activities. All benchmarks are based on content delivered via the HTTP/HTTPS protocol and rendered using standard web browsers (preferably the device's built-in web browser).

Each benchmark is defined by a sequence of content accessed (websites visited, videos or audio played). All benchmarks shall be configured to automatically loop, with a period of the same duration as the length of the measurements, to ensure repeatable results and eliminate timing dependencies. Thus, overall measured content should be identical regardless of when the measurements started.

2.1 Benchmark Requirements

To accurately assess the power consumption of different computers under realistic conditions, a suitable benchmark must meet the following criteria:

- representativeness / relevance
- repeatability
- reproducibility
- reasonable effort
- portability / platform independence

2.1.1 Representativeness / Relevance

The benchmarks shall represent real user activity. The content shall resemble that of real websites and what real video and audio content looks like. Different activities will use different amounts of computing resources and thus cause different power draw in a computer. Interacting with dynamic graphical content will use more power than reading a news article. A representative benchmark shall include a mix of activities reflecting typical user behaviour.

2.1.2 Repeatability of Results

All benchmarks shall be repetitive and selectively randomized to ensure the measurement results are relatively independent of the timing of measurements. Therefore, all media may need to be altered to ensure repeatable test results.

For example, video and audio clips may be shortened, then looped repetitively as segments within a single video stream, or file. Frame sequences within the clip segments shall be randomized to reduce the probability of unexpected optimizations resulting from the looping. Similarly, websites will loop in repetition over the test period. Websites must ensure that caching effects don't introduce timing dependencies, especially for looped content.

2.1.3 Reproducibility of Results

Results must be reproducible by independent entities conducting the tests when following the testing procedures. They shall not depend on environmental factors, measurement equipment, or lab setup. Benchmark, execution environment, and testing procedure must be designed with this in mind.

2.1.4 Reasonable Effort / Duration

The ideal benchmark would test all possible features in an infinitely short time, allowing instantaneous accurate measurement of a device's power consumption performing these activities. The most realistic benchmark, on the other hand, would mimic the average web browsing and streaming sessions exactly and therefore take many hours to complete.

However, a practical benchmark must be short enough to permit obtaining measurements within a reasonable amount of time while also allowing cycling through a representative amount of content to allow projecting the energy consumption of a real user's session.

2.1.5 Portability / Platform Independence

The benchmarks shall be free of any host specific code (i.e., no code compilation on the host shall be required). This will allow maximum portability of the benchmarks, meaning they can be executed on many different platforms and devices.

2.2 Emulating Real User Activity

The purpose of the benchmark created in this project is to mimic real user activity representing what a typical user does when interacting with his web browser, watching videos, etc.

For benchmarks to be practical, their duration shall be shorter than a real user's typical web browsing and streaming session, which can last for hours. It is important, however, that the condensed activity be representative and suitable for projecting a real user's energy use. This will require proper weighting of the content and timing of the activities performed.

Real user activity also includes idle time, when the user consumes content without touching the computer's input devices (i.e., the user is reading, viewing, or listening).

2.3 Description of Content Test Applications

This section describes the underlying features and technologies used for developing the test applications, as well as a brief description of each application website.

2.3.1 Webpage Features

Many popular websites offer more than just static text and images. Features commonly found on websites are display advertisements, auto play videos, forms, conditionally displayed content, as well as content requiring a user account with that website. Furthermore, a lot of the code found on a typical web page is unrelated to the content displayed. Instead, it relates to user tracking by the website itself, but also by third parties.

2.3.2 Webpage Technologies

While there are many different web technologies in use today, the ultimate content of virtually all web pages relies on the following underlying core technologies:

- HTML, or HyperText Markup Language, is used to create the basic structure and content of a webpage.
- CSS, or Cascading Style Sheets, is used for the visual design of a webpage – controlling where content elements are placed and how they are formatted.
- JavaScript is used to define the interactive elements of a webpage.

These are the main technologies a web browser deals with when it processes and presents a web page on the screen and lets the user interact with the page. They are often also referred to as front-end technologies. While each has its own strengths and unique features, a substantial overlap exists between them. Basic text can be produced by either, but even many advanced features can be covered by all of them, though with varying effort and limitations. Different websites put varying degrees of emphasis on each of these technologies, depending on the development framework used to create and manage the site. The web benchmark content developed here employs a representative mix of all three core technologies, which are discussed in more detail in the subsequent sections.

Numerous other technologies are used for creating and managing websites, typically either running on the web server or on the computers used by the website developers, therefore not directly impacting the web browser. Those technologies are also referred to as back-end technologies.

Back-end technologies can be grouped into the following categories:

- programming languages (used to program server behaviour, or generation of HTML/CSS/JS code);
- frameworks (automate/simplify managing web pages);
- databases (data storage on the server, accessed through the server).

Programming languages, such as PHP, Ruby, Java, C#, and Visual Basic run on the web server that hosts the web pages in question. That means the code written using those programming languages instructs the web server how to operate and generate HTML, CSS, and JavaScript code for the web pages in question when a request from a web client (i.e., web browser) is received.

Another group of technologies that should be mentioned is libraries, which are collections of pre-defined functions to help developers reuse code for common tasks. They are specific to a programming language. The libraries relevant for the test web content are JavaScript libraries, which may be included in the JavaScript code of popular web pages.

2.3.2.1 HTML Code

The HyperText Markup Language (HTML) is the minimum technology required to make a web page. As HTML does include directives for formatting and positioning of text, originally, entire websites would be created using only HTML for every web page. Websites are structured into pages. Each page typically has its own HTML file.

Basic features of HTML aside from basic text content are:

- tables, can be nested;
- lists, bulleted or numbered;
- forms for the user to fill-out;
- hyperlinks (underlined text linking to other web pages or sites).

HTML5, the most recent version of HTML offers additional features that used to require browser plugins, such as:

- video and audio streaming;
- geo location, allowing to localize content.

2.3.2.2 Cascading Style Sheets

Cascading Style Sheets (CSS) is a language used for describing the visual presentation of a web page written in HTML. CSS enables the separation of presentation and content, including aspects such as the layout, colours, and fonts. In most cases, a single CSS file is used to control the look of multiple pages, or often all pages of a website. Additionally, using CSS, one can easily define different styles for different kinds of display devices (e.g., mobile devices versus computers). Also, the look of a web page can easily be changed by just changing the CSS, without touching the presented content.

The web content developed for power measurement is limited to CSS features supported by all browsers (i.e., CSS 2.1) as well as the following CSS3 modules:

- CSS Color Module Level 3
- CSS Selectors Level 3
- CSS Namespaces Module Level 3
- CSS Media Queries
- CSS Style Attributes

2.3.2.3 JavaScript

JavaScript (JS) is an interpretive programming language used to implement interactive and dynamic content on web pages. It can be used to automatically refresh or change content spontaneously depending on user or server activity. Combined with the rendering capabilities of HTML5, it can even be used to implement platform independent online games within a web browser.

Most website developers do not write all their JavaScript code from scratch. Instead, numerous JavaScript libraries are available to make JavaScript code development easier, eliminating the need to redevelop common functions, ranging from simple operations to complex graphical visualizations. Some libraries are open source; others are licensed commercially.

One of the most popular JavaScript libraries is jQuery. It is an open-source library using the permissive MIT license.

2.3.3 Content Developed for the Benchmark

The content developed for the purpose of this prototype consists of several web pages similar to what one would find in typical popular websites.

The test websites are designed to represent popular websites visited by North American users such as google.com, youtube.com, facebook.com, and wikipedia.org. The web technologies used by those websites are employed in the benchmark content developed. Replicas are used to provide consistent test content, reduce the impact of network traffic, and support test automation. Replicas include a simulation of the major features of the reference websites for the target pages.

2.3.3.1 Calendar Application

This application uses dynamic HTML5 code to allow users to interactively manage a calendar, storing all calendar entries in a JSON file.

Supported actions include:

- creating events
- deleting events
- editing events
- moving events in the calendar using drag and drop

All user actions, except for drag and drop, can be activated by the script processed by the execution environment.

2.3.3.2 Angular JS Application

This application uses the popular angular JS framework, which is based on JavaScript and HTML5.

The application presents search results for a set of phones and tablets with text and images, with animation when being rendered.

Supported actions include:

- scrolling
- clicking on hyperlinks
- clicking the back button

All user actions can be activated by the script processed by the execution environment.

2.3.3.3 Online Newspaper

This application replicates parts of the Los Angeles Times website. It includes text, photos, and embedded video with audio. The technologies used are HTML5 and JavaScript. To properly emulate real user behaviour (browsing for and reading articles), one of the key activities used here is autonomous page scrolling.

Supported actions include:

- scrolling
- clicking on hyperlinks
- clicking the back button

All user actions can be activated by the script processed by the execution environment.

2.3.3.4 HTML5 Studio

This website, created by Google, is used to showcase the abilities of HTML5 with dynamic content, rendering of animations, and 3D effects.

Supported actions include:

- scrolling
- mouse-over hovering
- mouse-drag activities
- clicking on hyperlinks
- clicking the back button

All user actions except for mouse-drag activities can be activated by the script processed by the execution environment.

2.3.3.5 Stock Charts

This website uses an HTML5 page with JavaScript to present stock chart data from a CSV file in dynamic charts. The charts are generated in an animated fashion.

The site is not interactive; it performs a fixed sequence of actions instead.

2.3.3.6 Video Streaming Site

The video streaming site allows streaming of videos similar to what users would see on sites like Netflix or YouTube. Video streaming allows user-controlled playback of videos. Yet it differs from standard video playback, which requires the entire video file to be downloaded to the end user's computer, while streaming allows content to remain on the server, transmitting only the segment the user is currently watching.

The website is based on an HTML5 web page with the video.js JavaScript library. The video.js library consists of two parts, the main video playback library, as well as the streaming extension. The streaming feature additionally requires a video player to produce the stream. The video stream is produced by ffmpeg playing back a 1080p video file. The video stream is then picked up by the web server, which manages the video stream for all connected computers.

2.4 Description of Execution Environment

To enable the execution of the benchmark across all test computers in a well-defined manner an execution environment that can deliver the benchmark content to the test computers via the network is required. This action should not require any special software on the test computers other than the operating system and a standard web browser.

The proposed execution environment includes a server hosting the benchmark content (webpages and media content) to the device under test via a dedicated network. The dedicated network is used to eliminate network delays from influencing the test results. The infrastructure envisions a plug load energy measuring device so the total energy consumed by the device under test during the rendering of the content will be captured.

It is assumed that the execution environment is one where the content is provided by a dedicated content server which the system under test accesses via a network connection. The setup used for content selection is outlined in Figure 1 and the elements of the execution environment are described in Table 1.

Figure 1: Test Environment

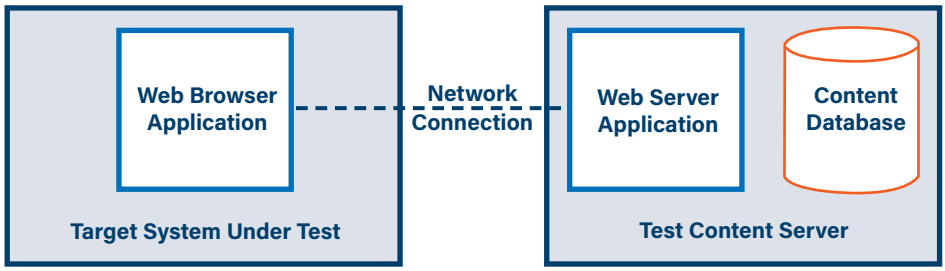


Table 1: Description of the elements comprised in the execution environment

Component	Description
Target System Under Test	The device under examination. It must have network access to the Test Content Server.
Test Content Server	A device dedicated to sending test content to one or more Target Systems. It can be located anywhere on the local network, which helps reduce latency seen with remote networks.
Content Web Server	An HTTP server dedicated to delivering test media to the Web Browser that will include HTML5-compliant web content, audio, video, and JavaScript code.
Web Browser	An HTML5-compliant web browser, with support for MP3 audio, H.264 video and ECMA Script 5-compliant JavaScript.
Network Connection	The network connection between the Test Content Server and the Target System Under Test shall be a LAN connection initially to eliminate the impact of network delays on the power measurements.

The execution environment for each active mode power measurement shall be the device's default, built-in web browser running on top of the default operating system. It is assumed that all computing devices will have a built-in web browser. On Windows 10, all measurements shall be performed using Microsoft's new Edge web browser, for example.

No other foreground applications shall be running during benchmark execution.

2.4.1 Impact of Background Activity

Today's operating systems, like Microsoft Windows, often perform routine maintenance tasks in the background [4], such as indexing or downloading/installing updates. Some operations can be computationally intensive, thus significantly increasing a computer's power consumption even while the device is otherwise idle, or the user is performing only light activity.

To minimize chances of such background tasks interfering with measurements, it is advised to temporarily disable high intensity background tasks while performing the measurements.

Such tasks may be:

- automatic updates
- virus and malware checking
- indexing service

The intent is to record the power consumption of computing devices in a state reflecting what a typical user would observe. One may argue these services should remain enabled while performing the measurements. Considering that the probability of such operations is usually quite low, it has been decided to preventively disable them as the measurement results will be more predictable in that case.

On the other hand, if a computing device is configured in an inefficient manner (e.g., the built-in spamware checker always consumes a significant amount of CPU), then the power measurements should reflect that fact, hopefully motivating the manufacturer to correct the inefficient configuration.

2.4.2 Impact of Network Delays

In an ideal world, the communication between the target system and the test content server would be infinitely fast incurring no delays. However, in reality, the network connection will always cause some delays and exhibit variable data transfer rates. At some point network delays can cause the target system to wait for data rather than process/display the content. Or, it may process less data than intended (e.g., displaying a high-definition video in a low resolution instead). It is conceivable that a system waiting for data or processing less data will consume less power than one processing the intended amount of data.

To minimize delays, the setup proposes to host the content server on the same local area network (LAN) where network connection quality can be controlled and therefore the bandwidth and network latencies be kept closer to the ideal scenario. However, to reproduce a realistic workload, artificial latencies can be introduced either on the server side, or within the content.

2.4.3 Web Server Requirements

The execution environment is agnostic to the actual web server platform used, so any web server software and hardware can be used as long as it supports current web standards.

2.4.3.1 Recommended Web Server Software

The web server software chosen for this prototype is NGINX due to its ability to host both standard web content as well as provide audio and video streaming services without additional 3rd party modules.

2.4.3.2 Web Server Operating System Requirements

The NGINX web server software requires a Linux based operating system. For the prototype developed as part of this project, a Ubuntu 16.04 based server was used but several other Linux distributions are supported as well.

2.4.3.3 Web Server Hardware Requirements

The following minimum hardware specifications are required for each node running the NGINX Controller:

- RAM: 8 GB RAM
- CPU: 8-Core CPU @ 2.40 GHz or similar
- Disk space: 155 GB free disk space.

2.5 Web Browsing Benchmark

The web browsing benchmark should represent a user's activity when performing typical tasks using a web browser application.

The benchmark should cover the following activities:

- sequentially opening multiple different websites;
- scrolling up and down the page;
- entering data in forms and submitting it;
- moving the mouse over some page items to open conditional content;
- downloading files;
- clicking on hyperlinks;
- opening new browser tabs and browser windows.

It is assumed that all the above activity can be emulated using platform independent JavaScript running as an overlay on the webpage(s) in question.

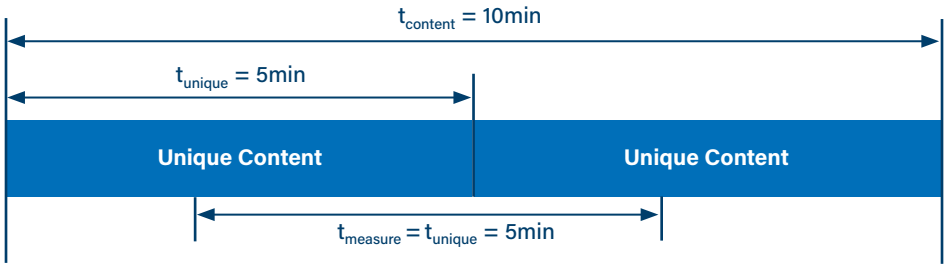
2.5.1 Benchmark Duration

The duration of the unique content should match the length of the measurements to be taken as shown in Figure 2. The web content must repeat the same content at least twice.

To ensure consistent results the interval length of the unique content must match the length of the measurement.

A duration of 5 minutes is suggested to allow the benchmark to cover all required website features. Less time would make it challenging to cycle through all typical features while more time might be considered an undue burden on the measurement procedure.

Figure 2: Web browsing benchmark and content duration vs measurement duration



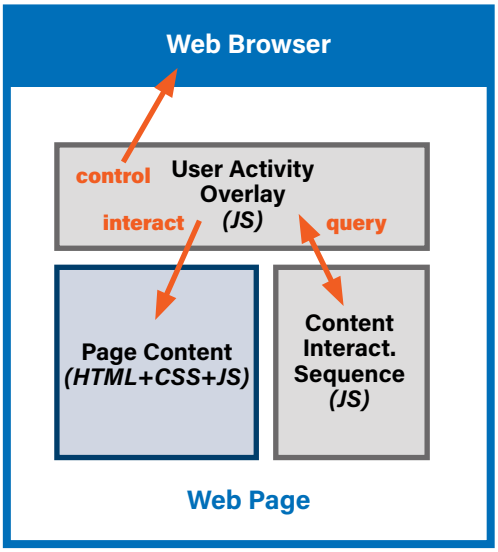
2.5.2 Emulation of User Activity

To fulfill the platform independence requirement, user activity cannot be emulated using utilities running on the target system under test. Instead, user activity shall be embedded in the content served by the test content server.

This can be achieved using JavaScript. Each of the web browsing activities to be emulated can be triggered by a JavaScript function. The content itself will need to be instrumented to allow the JavaScript overlay to recognize elements to interact with just like a real user would.

The web pages shall be constructed as depicted in Figure 3.

Figure 3: Page content layers including user activity overlay



Each web page will have a content specific interaction sequence defined in a separate JavaScript function. This JS function can be queried by the user activity overlay. The user activity is then emulated by the JS overlay. The latter controls the web browser (e.g., scrolling the page) and interacts with the content (e.g., follow hyperlink to open a new page or trigger dynamic content of the page).

2.5.3 Activities Not Covered

While most typical activity can be covered using JavaScript based automation, the tests will deviate from a real user manually performing those same activities.

2.5.3.1 Mouse Movement

When a real user performs the above web browsing activities, they typically do so by using their mouse as the input device. This includes mouse movements, mouse button presses, and scroll wheel activity. When using JavaScript instead of mouse input, the mouse pointer will never move, nor will scroll wheel activity occur.

However, all effects caused by mouse activity (clicks, mouseover events, and scrolling) can be triggered via JavaScript. The only part that cannot be emulated is the hardware interrupts triggered by mouse activity and that of the device driver responding to those interrupts. But as far as the browser is concerned all consequences of such activity are equivalent.

Lack of mouse movement activity is expected to have a very small impact on power consumption. That means the power measured using JavaScript based activity will be minimally lower than that based on real mouse activity.

Yet, since mouse activity consequences (resulting browser events) can be emulated using JavaScript, the impact should be negligible.

2.5.3.2 Other Activities

The automated benchmarks will not include the same long pauses that may be observed during a real user's web browsing session (e.g., while reading page content). Some activity pauses shall be included, but to cover a representative amount of different content during a reasonable amount of time, the pauses will need to be shortened.

2.6 Video Streaming Benchmark

The video streaming benchmark shall emulate a user watching video content on a streaming website such as YouTube.

Factors influencing power consumption are the video encoding standard used, the resolution of the video, the size of the video image on the screen (full screen vs small window), and the video content. For instance, fast moving sections require more processing than slow moving or even still images.

The video streaming benchmark shall automatically cycle through a set of video clips of different resolution and content.

2.6.1 Benchmark Duration

The duration of the unique content should match the length of the measurements to be taken (see Figure 4). The web content must repeat the same content at least twice.

To ensure consistency of results the period of the unique content must match the length of the measurement.

A duration of 5 minutes is suggested to allow the benchmark to cover all required website features. Less time would make it challenging to cycle through all typical features while more time might be considered an undue burden on the measurement procedure.

The length of each video clip will depend on the number of different video clips required to cover the relevant video types. JavaScript embedded on the web page shall be used to automatically play the next clip when the previous one ends.

2.7 Audio Streaming Benchmark

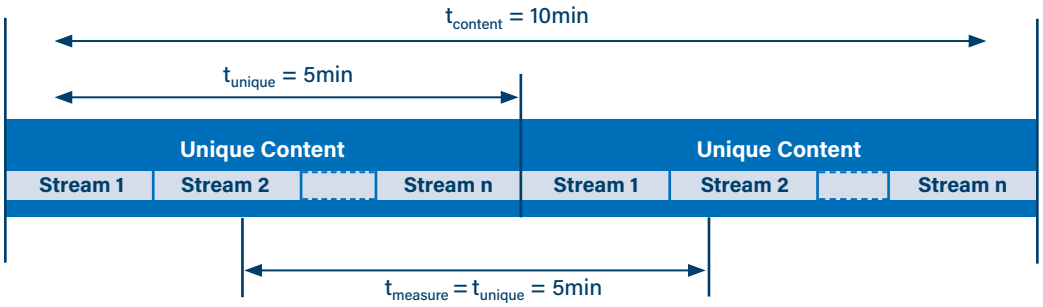
The audio streaming benchmark will emulate a user listening to audio on websites like Pandora or Spotify. Factors influencing power consumption are the audio encoding standard used, the bitrate, as well as the audio content.

3 Testing Procedures

A prototype of the proposed execution environment described in Section 2 was developed and tested across a set of different computer configurations.

The objective was to verify that the proposed methodology would produce repeatable and reproducible results, and that the execution environment was reliable.

Figure 4: Video streaming benchmark and content duration vs measurement duration



3.1 Test Objects

To cover a broad spectrum of different computer configurations, the tested computers include a typical consumer laptop as well as a desktop computer with integrated graphics and the same desktop computer with a discrete graphics card.

The specifications shown in table 2 are from the computers used for testing.

These computers were selected to demonstrate the applicability of the new benchmark across different classes of PCs to demonstrate repeatability and reproducibility.

3.2 Browsers Tested

The benchmark has been validated on the following web browsers:

- Microsoft Edge
- Mozilla Firefox
- Google Chrome

These three web browsers have been selected for their popularity among Windows 10 users. The most recent versions have been used, without any browser plugins or extensions.

3.3 Testing Procedure

The test procedure has been refined to ensure testing meets requirements defined for this benchmark, making measurements repeatable and reproducible while keeping the measurement effort reasonable.

The test procedure consists of the following steps:

- Initialize the measurement device;
  - Configure it to conduct power measurements for 5 minutes.
  - Configure a 100 ms sample interval.
  - Measurement output should be a CSV file containing 3000 data samples.
- Initialize DUT;
  - Ensure DUT power cable is connected as a load to the measurement device's measurement power strip.
  - Ensure no other devices (e.g., monitor) are connected to the measurement device's measurement power strip.
  - For desktop computers ensure only USB keyboard and mouse, as well as a 1080p monitor is connected, either via an HDMI or Display Port cable.
  - For laptop computers ensure battery is removed or fully charged as battery charging current measurements are to be avoided.

Table 2: Specifications of the test objects

Test Object #	1	2	3
Type	Laptop PC	Desktop PC Integrated graphics	Desktop PC Discrete Graphics
Model Name	Dell Latitude E5570	Dell XPS8940	Dell XPS8940
CPU	Intel i5-6200U 2.3GHz (up to 2.8GHz max) 15W TDP	Intel i7-10700 2.9GHz (up to 4.8GHz max) 65W TDP	Intel i7-10700 2.9GHz (up to 4.8GHz max) 65W TDP
RAM	8GB DDR4, 2133MHz	16GB DDR4, 2666MHz	16GB DDR4, 2666MHz
Storage	1TB Solid State Drive, Samsung EVO860 SATA	512GB Solid State Drive, NVMe M.2 PCI-Express 3.0	512GB Solid State Drive, NVMe M.2 PCI-Express 3.0
Graphics	Integrated Intel HD Graphics 520	Integrated Intel UHD Graphics 630	Dell NVIDIA RTX 2060 GPU

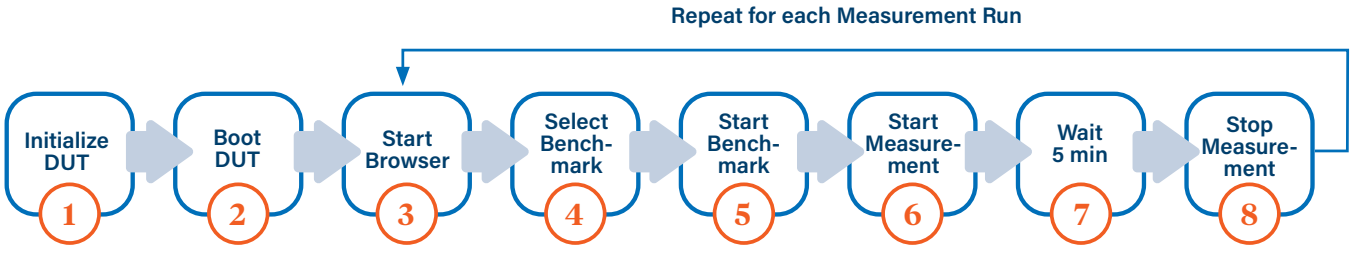


"The test procedure has been refined to ensure testing meets requirements defined for this benchmark, making measurements repeatable and reproducible while keeping measurement effort reasonable."

- Boot DUT;
  - Perform normal Windows boot, ensure no other end user applications are started.
  - Temporarily disable updates, in "Settings," "Windows Update".
  - Leave the DUT idle for at least 10 minutes after the initial boot to allow the operating system to complete periodic system maintenance tasks.
- Start the browser;
  - Open the browser, without any other end user applications open.
- Select the benchmark to start, if multiple are available;
- Start the benchmark;
- Start the measurements;
  - Ensure the benchmark is already running before starting the measurements.
  - Start the measurements within 5 minutes after starting the benchmark.
  - The benchmark will restart from the beginning after exactly five minutes, ensuring the exact timing when measurements start will not affect the overall result.
- Wait 5 minutes;
  - The intended length of the measurement interval is 5 minutes.
  - The benchmark is designed to produce repeatable results as long as the measurement interval is kept at 5 minutes.
- Stop the measurements;
  - Either stop the measurements manually or limit the measurement data collection to 5 minutes if the measurement device or software permits.
- Repeat from the "start browser" step to conduct the next measurement.

The testing procedure's sequence of steps is summarized in Figure 5.

Figure 5: Testing procedure



To minimize measurement errors, we suggest a minimum of at least 3 measurement runs for each DUT and browser combination.

4 Validation of Results

The benchmark was run on each of the computers, and the power was measured. To quantify the repeatability of the results each measurement was conducted 10 separate times.

4.1 Results Overview

For each DUT a series of ten runs was performed, where the benchmark was run and the power was measured following the procedure described in section 3.3. Results are summarized in Table 3.

As shown in Figure 6, most measurement series show very consistent results between different runs, except for only one series, shown in blue, which had 3 runs exhibiting significantly higher power numbers than the remaining runs.

4.3 Causes of Measurement Errors

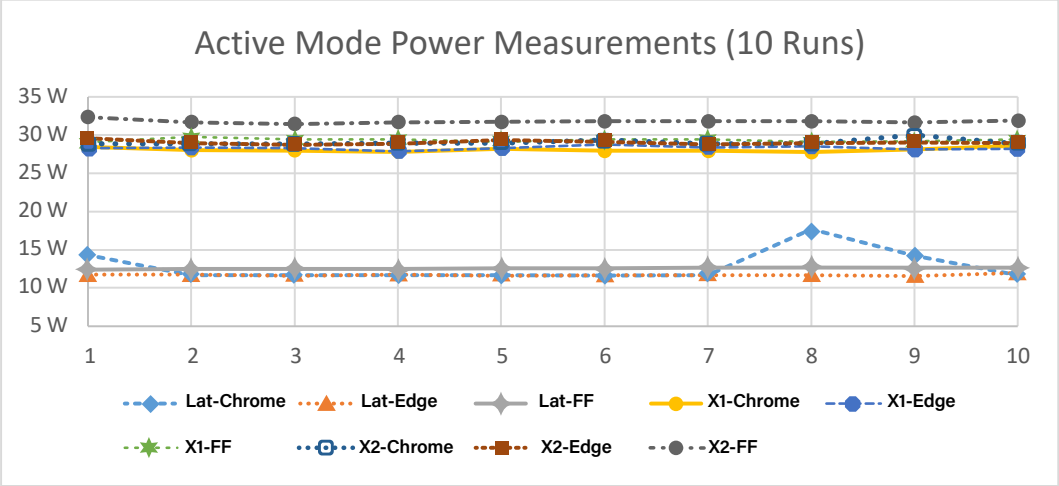
The key factor likely to affect the standard deviation of the results is the behaviour of the many tasks running as part of the operating system. While disabling system updates was intentionally added to the list of steps for preparing the DUT, there are many other processes that may occasionally become active. Even a fresh Windows 10 install can have more than 100 tasks showing in Windows Task Manager. Many will be dormant almost all the time but it is very difficult to ensure they remain dormant during the test. Using Windows Task Manager, it is possible to spot the tasks that contribute to increased power consumption by looking at their CPU utilization. Any tasks with sustained CPU utilization of 1% or more can be investigated and potentially shut down. Tasks that appear sporadically are more difficult to track down, however.

Another component complicating the matter are Windows' services. Occasionally one or more of the numerous Windows Services will become active and cause increased power consumption. In that case

Table 3: Measured power and standard deviation based on 10 measurement runs

DUT (PC)	OS	Browser	Power	Standard Deviation
Dell Latitude E5570 Laptop (Lat)	Windows 10	- (idle)	9.4 W	0.2 W (2%)
		Google Chrome	12.8 W	2.0 W (16%)
		Microsoft Edge	11.7 W	0.1 W (1%)
		Mozilla Firefox	12.6 W	0.1 W (1%)
Dell XPS 8940 Desktop, internal graphics (X1)	Windows 10	- (idle)	23.2 W	0.3 W (1%)
		Google Chrome	28.1 W	0.2 W (1%)
		Microsoft Edge	28.3 W	0.2 W (1%)
		Mozilla Firefox	29.3 W	0.3 W (1%)
Dell XPS 8940 Desktop, discrete graphics (X2)	Windows 10	- (idle)	24.0 W	0.2 W (1%)
		Google Chrome	29.1 W	0.4 W (1%)
		Microsoft Edge	29.0 W	0.3 W (1%)
		Mozilla Firefox	31.8 W	0.2 W (1%)

Figure 6: Power measurement results for each of the 10 runs



it is much more difficult to track down the service responsible for the power increase as CPU utilization isn't broken down for individual services. Instead, many services will have their CPU use reported as part of the "svchost" task or even the system task.

4.4 Results Reproducibility

The combination of execution environment and testing procedure developed in this project ensures reproducible power measurement results across all benchmarks. Given the same set of computers to be tested we expect the same results to be obtained regardless of the entities conducting those measurements.

The execution environment in Section 2.4 is fully self-contained without any dependencies on external tools, technologies, or content. It uses only locally present immutable content hosted via a local area network (LAN), eliminating the risk of network latencies affecting the results.

The testing procedures summarized in Section 3 are structured to be sufficiently comprehensive to eliminate factors such as different lab conditions, or potential user errors. The testing procedures include detailed step by step instructions for how to prepare the computers to be tested uniformly, as well as how to conduct the measurements in a standardized fashion.

4.4.1 Execution Environment Reliability

The execution environment developed has been running continuously for several weeks while being accessed by multiple test computers simultaneously. Since the execution environment is based on a tried and tested web server technology, the results were not affected by how the web server was being exercised.

Having multiple test devices access the web server concurrently did not impact the results; yet, only four concurrent devices have been tested.

4.4.1.1 Known Limitations

Any web server installation has a maximum number of requests it can handle per second, based on the server's hardware specification and network bandwidth limitations. For a local web server servicing a handful of devices under test, web server capabilities far exceeded this project's needs. But if such a server were used for testing many devices concurrently it would have to be scaled up accordingly.

Scaling web services to service a larger number of devices is a well understood problem with many solutions available, commercial and open source based.

4.4.2 Reasonable Effort/Duration

The benchmark was limited to 5 minutes to achieve a test procedure with a reasonable duration as well as a low effort in conducting the complete procedure.

This allows a representative mix of web technologies and use cases to be exercised, including the full test procedure, with three iterations of the power measurements completed in under 30 minutes.

4.4.3 Portability

This project focused on Windows PCs only so portability was only tested regarding compatibility with different web browsers on the PCs tested.

However, the benchmark and execution environments being purely web browser based without any platform/device specific code should ensure the execution environment and the developed benchmarks will be portable to many other devices.

The web standards used in the benchmarks are widely supported by a large variety of devices, including different operating systems like Linux and MacOS, portable devices like mobile phones tablets, as well as TVs and game consoles. They all include web browsers supporting today's web standards. Therefore, the same benchmark tests would be applicable to many other operating systems and devices with either no changes required, or only minimal ones.

4.4.3.1 Known Limitations

The biggest uncertainty about the compatibility of the known browsers and other devices considered for testing is the changes vendors introduce with newer versions. Some features such as automatic video playback may require additional steps in future browser versions. However, in most cases this can be mitigated by following the guidelines and recommendations specified in the release notes or user manuals accompanying the browsers or devices.

Nevertheless, it is impossible to predict what kind of changes browser or device vendors will introduce, so there is always a small chance some websites included in the benchmark will require modifications or, worst case, will need to be replaced or removed from the test.

5 Conclusions and Next Steps

The new test procedure presented in this report has introduced the Active Mode Benchmark for Computing Appliances with its content and scenario and accompanying execution environment encompassing the device under test, remote web server, and user emulator. The main value of the Active Mode Benchmark is its simplicity. It consists of typical web-based applications used by everyday computer users, like news browsing, shopping, or watching videos. Contrary to other similar projects, the benchmark proposed here is not synthetic in its composition, but replicates real computer use as closely as possible. It is focused, rather than attempting to cover corner case applications.

Validation testing has confirmed the repeatability and reproducibility of the results on a small set of test computers. While the sample size was small, the test method is based on standard web browsers known to be available and to exhibit the same behaviour on any computer and even other devices. Thus, it is expected this test method will be applicable to a large set of computing devices. Validation testing has furthermore demonstrated the reliability of the execution environment.

All past and current Energy Star tests for computers use the idle mode (short idle, long idle, and sleep) to represent computer's power consumption in all modes, effectively eliminating active mode tests. The logic behind is using the idle mode as a proxy for the active mode when the error introduced stays below 15%.

It is important to test computers beyond the idle mode since modern computers have hugely improved idle mode power consumption, but disproportionally less than that of the active mode. Additionally, modern computers are 24/7 operating devices spending most duty cycle in active mode. Therefore, to avoid misguiding policymakers and industry it is recommended to retest the idle proxy error on modern computers and revise the Energy Start test procedure accordingly. The Active Mode Benchmark proposed in this report can be useful guidance in future revisions of the Energy Star test procedure.

Through the years, the Energy Star testing procedure has been regularly modified to reflect changes in how computers operate. As the Energy Star testing procedure already includes all definitions for workload testing, it would be straightforward for Energy Star

to incorporate the test procedure described here. As for the actual testing conducted by the industry, the proposed Active Mode Benchmark is a fully automated procedure, so a minimal additional effort on top of the current idle mode measurements shall be required.

As computers further evolve into devices operating 24/7 equipped with performance intensive artificial intelligence computing at the edge, testing computers just in idle mode shall become irrelevant. Consequently, the expected energy and carbon savings implemented through current regulations and EnergyStar procedure shall be much lower than anticipated.

This project receives content locally coming from a web server machine in the same local area network (LAN) as the computer under test. As the next steps, to improve benchmark representativeness, the web server machine should be accessed over the wide area network (WAN). Additionally, larger scale testing involving other types of desktops and laptops running under other operating systems, like those of Apple, should be included.

References

[1] SEAD Computer Working Group, "Power mode assessment," 2016. [Online]. Available: https://superefficient.org/publications/power-mode-assessment

[2] NRDC, "California approves nation's 1st computer energy standards," 2016. [Online]. Available: https://www.nrdc.org/experts/pierre-delforge/california-approves-nations-1st-computer-energy-standards

[3] EnergyStar, "CSA Group computer real-world energy use test procedure," 2018. [Online]. Available: https://www.energystar.gov/sites/default/files/CSA_Computers_Presentation.pdf

[4] Microsoft, "Automatic Maintenance," May 31, 2018. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/taskschd/task-maintenence.

Other Resources

[1] Energy Star, "Energy Star program requirements for computers v. 8.0," Apr 2020. [Online]. Available: https://www.energystar.gov/sites/default/files/ENERGY%20STAR%20Computers%20Final%20Version%208.0%20Specification.pdf

[2] V. Zivojnovic and D. Mista, "All-in-one computer idle power analysis," National Resource Defense Council, Apr 28, 2016. [Online]. Available: https://www.nrdc.org/sites/default/files/aggios-aio-report-20160429_0.pdf.

[3] N. Galov, "Cloud adoption statistics for 2021," Aug 2021. [Online]. Available: https://hostingtribunal.com/blog/cloud-adoption-statistics.

[4] S. J. Vaughan-Nichols, "The PC is dead. Long live the cloud PC.," Dec 18, 2019. [Online]. Available: https://www.zdnet.com/article/the-pc-is-dead-long-live-the-cloud-pc/.

[5] W3C, "HTTP – Hypertext Transfer Protocol," 1 Jun 2014. [Online]. Available: https://www.w3.org/Protocols/.

[6] J. Vitek and T. Kalibera, "Repeatability, reproducibility, and rigor in systems research," presented at the 11th International Conference on Embedded Software, Taipei, Taiwan, Oct 9-14, 2011.

[7] WHATWG, "HTML living standard," Aug 12, 2021. [Online]. Available: https://html.spec.whatwg.org/.

[8] W3C, "CSS specifications," 15 Jul 15, 2021. [Online]. Available: https://www.w3.org/Style/CSS/specs.en.html.

[9] ECMA, "ECMAScript® 2018 language specification," Jun 2018. [Online]. Available: https://262.ecma-international.org/9.0.

[10] PHP Community Foundation, Inc., "PHP Language Specification," Apr 25, 2020. [Online]. Available: https://phplang.org/.

[11] Ruby Community, "Ruby documentation," Jul 2021. [Online]. Available: https://www.ruby-lang.org/en/documentation/.

[12] J. Gosling, B. Joy, G. Steele, G. Bracha, A. Buckley, and D. Smith, "The Java® language specification," Aug 21, 2018. [Online]. Available: https://docs.oracle.com/javase/specs/jls/se11/html/index.html.

[13] Microsoft, "C# introduction," Jul 1, 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/introduction.

[14] Microsoft, "Visual Basic language specification," Aug 2, 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/visual-basic/reference/language-specification/.

[15] W3C, "HTML5," Apr 5, 2011. [Online]. Available: https://www.w3.org/TR/2011/WD-html5-20110405/.

[16] W3C, "Cascading style sheets level 2 revision 1 (CSS 2.1) specification," Jun 7, 2011. [Online]. Available: https://www.w3.org/TR/2011/REC-CSS2-20110607/.

[17] W3C, "CSS color module level 3," Jun 19, 2018. [Online]. Available: https://www.w3.org/TR/2018/REC-css-color-3-20180619/.

[18] W3C, "Selectors level 3," Nov 6, 2018. [Online]. Available: https://www.w3.org/TR/selectors-3/.

[19] W3C, "CSS namespaces module level 3," Mar 20, 2014. [Online]. Available: https://www.w3.org/TR/css-namespaces-3/.

[20] W3C, "Media queries," Jun 19, 2012. [Online]. Available: https://www.w3.org/TR/css3-mediaqueries/.

[21] W3C, "CSS style attributes," Nov 7, 2013. [Online]. Available: https://www.w3.org/TR/css-style-attr/.

[22] ECMA International, "ECMA-404 the JSON data interchange syntax," Dec 2017. [Online]. Available: https://www.ecma-international.org/publications-and-standards/standards/ecma-404/.

[23] Google, "AngularJS - superheroic JavaScript MVW framework," 2018. [Online]. Available: https://angularjs.org/.

[24] Brightcove, Inc., "Video.js – Make your player yours," [Online]. Available: https://videojs.com/.

[25] Wikipedia, "Advanced video coding," Jun 2019. [Online]. Available: https://en.wikipedia.org/wiki/Advanced_Video_Coding.

[26] F5 NGINX, "NGINX controller tech specs," [Online]. Available: https://docs.nginx.com/nginx-controller/admin-guides/install/nginx-controller-tech-specs/.

[27] W3C, "Cascading style sheets level 2 revision 1 (CSS 2.1) specification," Apr 12, 2016. [Online]. Available: https://www.w3.org/TR/2011/REC-CSS2-20110607/.

[28] B. Urban, K. Roth, M. Singh, and D. Howes, "Energy consumption of consumer electronics in U.S. Homes in 2017," Consumer Technology Association, Dec, 2017. [Online]. Available: https://cdn.cta.tech/cta/media/media/advocacy/pdfs/energy-consumption-of-consumer-electronics-in-u-s-homes-in-2017-(fraunhofer-usa,-commissioned-by-cta,-december-2017).pdf

[29] Intertek, "About Intertek," [Online]. Available: http://www.intertek.com/about/.

[30] C. Cheng and J. Santo Domingo, "The green machines," PC Magazine, pp. 66-72, Apr 2008.

[31] *Measuring the Energy Consumption of Personal Computing Products,* ECMA 383, ECMA International, Geneva, Switzerland, 2010.

[32] IEC 62623, "Desktop and notebook computers – Measurement of energy consumption," 2012. [Online]. Available: https://webstore.iec.ch/publication/7271

Appendix A – Benchmark Installation

This appendix discusses some implementation details of the Active Mode Benchmark prototype as well as installation instructions for the execution environment.

A.1 Directory Structure and Files

The implementation files consist of:

- files containing the benchmark and user activity emulation framework
- files implementing the content
- files containing the user actions specific to the respective content
- support libraries used by content and benchmark code

A.1.1 User Activity Emulation Framework

The framework for emulating user activity is a set of custom JavaScript functions developed fully from the ground up to replicate user activity, located in the files *"Benchmark.js"* and *"BenchmarkActions.js"* (top-level directory). It relies on configuration files in JSON format and a JavaScript file to specify user activity sequence for each website.

A top-level configuration file contains references to individual user action files for each website, establishing the relationship between the user activity specification and corresponding website.

A.1.2 Content Websites

The source code for each of the content websites/applications can be found in the "app" folder. Each website is contained within its own sub-folder featuring a primary HTML file as well as supporting files, grouped into sub-folders where appropriate.

A.1.3 Supporting Libraries

Some of the used libraries are kept separate from the website content relying on them. This includes the jQuery library, as well as the Video.js library. Other libraries that are specific to the content website using them, such as Angular.js, are included within the directory structure of the content application.

A.2 Execution Environment Setup

The following describes how to install and configure the benchmark execution environment.

A.2.1 Web Server Installation

The instructions for installing the NGINX web server depend on the operating system version used. On Ubuntu 20.04 NGINX can be installed by running the following command:

```
sudo apt install nginx
```

By default, all content and application files for NGINX must be placed into /var/www/html, while the main NGINX configuration file will be in /usr/local/nginx/conf.

A.2.2 Web Server Streaming Module Installation

Like most web servers, the NGINX web server does not natively support streaming video and audio files. Therefore, an additional module must be installed to add streaming support. For NGINX, the most popular option is the NGINX RTMP module (Real Time Messaging Protocol). Complete installation instructions can be found on 3rd party websites, such as "HowtoForge":

How to Install Nginx with RTMP Module (howtoforge.com)

A.2.3 Media Player Installation

A media player is required to feed the audio and video streams to the web server. One open-source option is ffmpeg, which can be invoked to produce a stream using this command and options:

```
ffmpeg -re -stream_loop 100 -i benchmark.mp4 -vcodec copy -loop
    -100 -c:a aac -b:a 1 60k -ar 44100 -strict -2 -f flv
    -flvflags no_duration_filesize rtmp://server.ip/streamfolder
```

A.2.4 Video File Installation

The video file used for the benchmark can be copied either directly to the web server directory or to any other location, as long as that location is directly referenced when invoking the video player. For this phase the open-source video "Big Buck Bunny" was used. It can be downloaded here:

Big Buck Bunny (blender.org)

A.2.5 Web Content Installation

All web content files must be copied into the web server directory, taking care to preserve the content's original directory structure. By default, this location is /var/www/html.

CSA Group Research

In order to encourage the use of consensus-based standards solutions to promote safety and encourage innovation, CSA Group supports and conducts research in areas that address new or emerging industries, as well as topics and issues that impact a broad base of current and potential stakeholders. The output of our research programs will support the development of future standards solutions, provide interim guidance to industries on the development and adoption of new technologies, and help to demonstrate our on-going commitment to building a better, safer, more sustainable world.

CSA GROUP™

© 2022 Canadian Standards Association. All Rights Reserved.