JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

기반 기술

J. Adv. Navig. Technol. 23(4): 328-332, Aug. 2019

웹 어셈블리 시스템 아키텍처 모델

Web Assembly System Architecture Model

박 진 태* · **문 일 영**
한국기술교육대학교 컴퓨터공학과

Jin-Tae Park* · **Il-Young Moon**
Department of Computer Engineering, KOREATECH, Chungcheongnam-do, 31253, Korea

[요 약]

웹 기술의 발전으로 다양한 시스템 환경에서의 기술적 융합을 웹 인터페이스를 통해 수행이 가능해졌다. 웹은 그 역할에 따라 웹 1.0시대부터 4.0시대로 분류할 수 있으며, 정보의 연결 (connects information), 사람의 연결(connects people), 지식의 연결 (connects knowledge), 지능의 연결(connects intelligence)의 특성을 갖는다. 또한, 4차 산업 혁명을 거치면서 모바일 앱을 통한 다양한 기술적 needs가 발생하였고, 단순한 정보의 제공 수단이던 웹에서, 3D, 가상/증강, 비디오/오디오 프로세싱 등의 기능 수행이 가능해졌다. 이러한 시대적 needs를 뒷받침하기 위한 기술 표준이 연구되었다. 본 논문에서는 그중 하나인 웹 어셈블리에 대해 분석하였다. 웹 어셈블리를 기존의 웹 시스템(혹은 플랫폼)과 연계, 융합하여 활용하는 방안에 대해 살펴보고, 다양한 사례를 통해 기술적 의미를 분석 한다. 또한, 기존의 자바스크립트와 웹 어셈블리를 융합할 수 있는 아키텍트에 대한 연구를 진행하고, 추후 연구 방향에 대해 논하고자 한다.

[Abstract]

Advances in web technology have enabled technical convergence in various system environments to be carried out through the web interface. The Web can be categorized from the Web 1.0 to the 4.0, depending on its role, it has the characteristics of connects information, connects people, connects knowledge, and connects intelligence. In addition, various technological needs occurred through the mobile app during the 4th Industrial Revolution, and functions such as 3D, virtual reality, augmented reality, video/audio processing were enabled on the web, which was a simple means of providing information. Technical standards have been studied to support these period needs. In this paper, I would like to mention one of the Web assembly. We will explore ways to link and fuse Web assembly with existing web systems (or platforms) and analyze their technical implications through a variety of examples. In addition, we will conduct a study on the architecture that can fuse the existing javascript with the web assembly, and discuss the future direction of the study.

Key word : WebAssembly, Web architecture, Javascript, WebAssembly management system, WASI.

https://doi.org/10.12673/jant.2019.23.4.328

Received 6 August 2019; **Revised** 9 August 2019
Accepted (Publication) 23 August 2019 (30 August 2019)

***Corresponding Author; Jin-Tae Park**

Tel: +82-10-2368-6591
E-mail: park.jintae@kt.com

Ⅰ. 서 론

웹 기술의 발전으로 다양한 환경에서 동작하는 웹의 역할이 날로 증대하고 있다. 이렇듯 날로 증대하는 웹의 역할에 따라 웹 1.0에서 4.0 시대로 분류할 수 있다.

웹 1.0시대의 웹은 우리가 흔히 알고 있는 월드와이드웹을 의미한다. 이 시대의 웹은 주로 디렉터리 검색을 지원했으며, 모든 자료는 체계적으로 분류되어 사용자들에게 카테고리 형태로 제공되었다. HTML(hyper text markup language)을 통해 문서의 구조를 표현하였으며, 텍스트와 링크가 웹 페이지의 주된 형태를 이루고 있었다. 즉, 링크를 통해 전 세계 웹 페이지 문서를 통합하였고, 정적인 형태의 문서를 화면에 출력하였다. 웹을 통한 정보의 연결(connects information)이 웹 1.0시대의 주된 목표였다.

웹 2.0시대의 웹은 개방, 참여, 공공의 개념을 바탕으로 사용자가 직접 정보를 생산하고, 웹을 통해 사용자 간 양방향으로 소통하는 시대를 의미한다. 이 시대의 웹은 사용자가 정보를 보기도 하고, 정보를 바꾸거나 직접 생산하면서 웹을 통해 공유되는 콘텐츠를 생산하였다. 웹 2.0시대부터 웹 애플리케이션, 웹 시스템, 웹 플랫폼의 개념이 생겨나기 시작하였으며, 집단 지성을 통한 데이터를 기반으로 사용자 참여 구조 방식의 효과를 얻을 수 있었다. 웹을 통한 사람의 연결(connects people)이 웹 2.0 시대의 주 된 목표였다.

웹 3.0시대의 웹은 시멘틱 기술을 이용하여 웹 페이지에 담긴 내용을 이해하거나 사용자 맞춤형 정보를 제공할 수 있는 지능형 웹 기술을 표현할 수 있었다. 즉, 컴퓨터가 스스로 사용자의 정보를 파악하여 원하는 정보를 보여주는 역할을 하는 것이다. 웹 3.0시대의 웹은 타 시스템과의 융합이 가능해진 시기이며, 거대 플랫폼으로 성장하는 시기이다. 웹을 통한 지식의 연결(connects knowledge)이 웹 3.0시대의 주 된 목표였다.

최근 웹 4.0시대의 웹은 기술의 영역을 확장하여, 인간 영역과의 경계를 희미하게 만드는 시대를 의미한다. 4차 산업혁명을 거치면서 거대 정보 시스템에 대한 needs가 증가하게 되었고, 이를 연결하는 역할로 웹을 활용하게 된 것이다. 인간 중심의 모든 것에 네트워크를 연결하고, 임베디드 된 웹 기술을 통해 정보 습득, 학습, 저장, 생산해 내며 서비스 제공을 가능하게 하였다. 웹을 통한 지능의 연결(connects intelligence)이 웹 4.0 시대의 주 된 목표였다. 이처럼 웹 기술이 기술의 영역 전반에 걸쳐 융합되고, 그 역할이 증대함에 따라, 타 기술과의 융합을 위한 새로운 표준들이 연구되고 있다. 그 중 대표적인 것인 웹 어셈블리이다[1].

웹 어셈블리는 C와 C++과 같은 프로그래밍 언어를 컴파일하여 모든 웹브라우저에서 빠르게 실행될 수 있는 바이너리 형식으로 바꾸는 기술이다. 즉, 웹 애플리케이션, 시스템 혹은 플랫폼 환경에서 네이티브 언어의 퍼포먼스를 활용할 수 있게 바꿔주는 것이다. 전통적으로 웹의 기능 수행을 담당하는 것은 자바스크립트였다. 자바스크립트는 웹 환경에서 동적 페이지를 만들기 위해 고안된 개발 언어로, 웹브라우저 내에 탑재 엔진을 통해 다양한 기능을 개발할 수 있도록 지원해주는 것이다. 이러한 자바스크립트를 대체, 보완하기 위한 기술이 연구되는 원인으로는 첫 번째, 웹을 통해 융합되는 기술이 다양해졌다. 두 번째, 웹을 통해 통신하는 데이터의 양이 기하급수적으로 증가하였다. 이러한 이유로 C언어와 같은 네이티브 언어의 높은 퍼포먼스를 활용하여, 웹 환경에서도 다양한 기술과 대용량의 데이터를 처리하기 위해 웹 어셈블리에 관한 연구가 진행 중이다.

웹 어셈블리는 바이너리로 컴파일 되기 때문에 네이티브 언어에 가까운 성능을 나타낼 수 있다고 알려져 있다. 하지만 현재까지 웹 어셈블리는 자바스크립트를 대체하지 못하고, 보완해주는 역할로 활용 중이다. 자바스크립트와 웹 어셈블리 영역을 분리하고, 서로 통신하며, 상호 보완재 역할을 하는 것이다. 하지만, 자바스크립트 영역과 웹 어셈블리 영역이 기능 동작을 위해 서로 통신을 하며, 데이터를 주고받는 형태에서 네트워크와 주고받는 데이터의 용량에 의해 속도 저하의 문제가 발생할 수 있다.

따라서 본 논문에서는 자바스크립트와 웹 어셈블리가 상호 보완적으로 동작하는 형태에서 공유하는 데이터를 관리하는 서버를 별도로 두어, 웹 어셈블리가 현재 수준보다 높은 성능을 나타내도록 하고, 다양한 기기와 기술의 융합에 이식될 수 있도록 하고자 한다.

Ⅱ. 웹 어셈블리 연구 동향

본 장에서는 웹 어셈블리와 관련된 최근의 연구 동향에 대해 살펴보고, 자바스크립트와 웹 어셈블리의 성능에 대해 논하고자 한다.

2-1 웹 어셈블리 표준

서론에서 언급했듯이, 웹 어셈블리는 웹브라우저에서 실행할 수 있는 새로운 형태의 코드이며, 네이티브 프로그래밍 언어를 컴파일 할 수 있도록 고안되었다. 이는 웹 플랫폼에서 기존에 동작할 수 없는 응용 프로그램을 사용할 수 있도록 해주며, 네이티브에 근접한 속도로 실행하는 것을 지원한다.

웹 플랫폼은 코드를 동작시키는 가상머신, 하드웨어 기능을 호출해서 웹/앱이 무언가를 동작할 수 있게 해주는 API(application programming interface) 집합으로 분류할 수 있다. 이전의 웹브라우저의 가상머신은 오직 자바스크립트만 불러올 수 있었다. 동적 페이지를 구성하는데 자바스크립트는 큰 역할을 했지만, 3D(three dimensions) 게임이나 가상/증강, 영상처리, 이미지/비디오 편집 등의 기능은 성능상의 문제가 존재하였다. 이를 해결하고자 고안한 것이 웹 어셈블리이며, 이는 자바스크립트를 대체하기 위해 만들어 진 것이 아닌, 상호 보완적으로 동작하도록 설계되었다. 웹 어셈블리의 가장 중요한 2원칙은 이식성과 보안이다. 같은 코드를 다양한 플랫폼에서 실행할 수 있도

록 설계(이식성)되었으며, 샌드박스를 이용하여 운영체제와 직접 통신이 아닌 간접적으로 통신(보안)하게 설계되었다. 이러한 원칙을 지키기 위해 웹 어셈블리 표준에서는 다양한 환경에서 웹 어셈블리가 동작할 수 있도록 하는 것을 목표로 하여, 표준 명세를 2개의 계층으로 분리하였다[2].

· **핵심 명세:** 임베디드 환경과 상관없는 웹 어셈블리의 언어와 문법을 정의하는 명세로 웹 어셈블리 모듈의 구조, 명령어, 바이너리 표현, 텍스트 표현 등에 관한 정의를 포함

· **임베디드 환경 관련 명세:** 임베디드 환경에 따라 웹 어셈블리 모듈과 상호작용을 하기 위한 API를 정의하는 명세

2-2 구글 Squoosh

구글 I/O 2018에서 구글이 발표한, 웹 기반 이미지 최적화 도구 Squoosh는 이미지 압축 툴이며, 이미지 해상도와 파일 크기, 포맷 변경 작업이 가능하다. 웹 어셈블리 기술을 이용하여 일반 브라우저에서 지원하지 않는 이미지 코덱 지원을 가능하게 하였으며, 이미지 로딩 후, 네트워크 오프라인 상태가 되어도 모든 작업이 가능하도록 한다[3]. 초기의 Squoosh는 자바스크립트 기반으로 구현하였고, 이미지 각 픽셀을 읽어 들여 다른 위치로 복사하는 형태의 알고리즘을 적용하였다. 이후 웹 어셈블리로 전환하여 구현하였다[4]. 기존 자바스크립트 기반의 시스템에서는 4094픽셀 x 4094픽셀 크기의 이미지를 회전하는데, 천 6백만 번의 반복 작업을 수행하게 되는데, 브라우저별 대체로 2초 내외로 수행되었음을 확인할 수 있다. 웹 어셈블리 기반의 시스템에서는 사용하는 언어별로 차이는 있지만, 대체로 1초 내로 수행됨을 확인할 수 있다. 다음의 그림 1은 언어별 처리 속도를 나타낸다. 자바스크립트 기반으로 웹 시스템을 구현할 때, 가장 중요한 것 중 하나가 폴백이란 개념이다. 웹 기술이 적용되는 환경마다 여러 가지 브라우저가 사용되고 있고, 사용하는 버전 역시 다양하므로 HTML 크로스 브라우징 이슈가 발생할 수 있다[5].

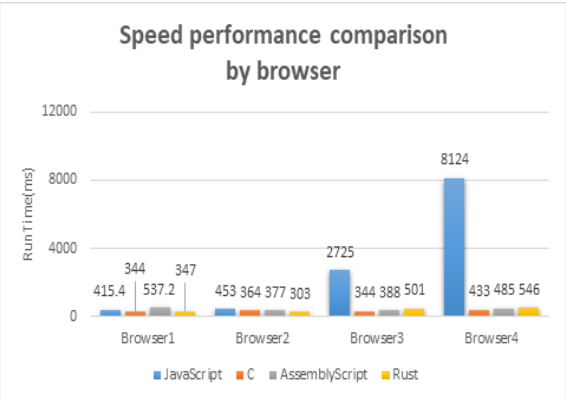
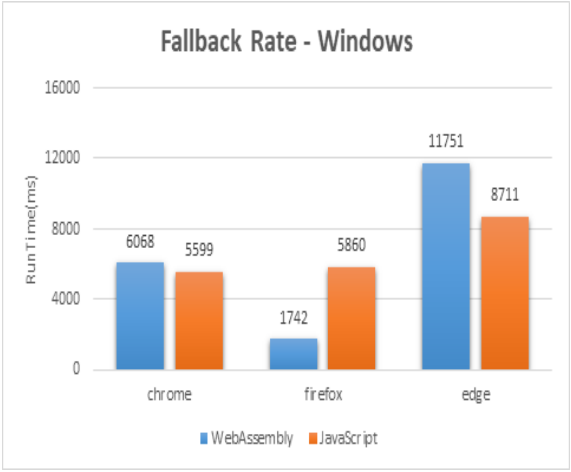


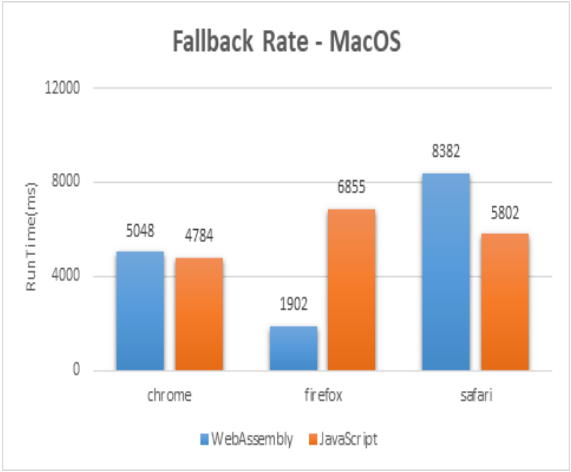
그림 1. 프로그래밍 언어별 이미지 처리 수행 속도 비교
Fig. 1. Comparison of Image Processing Performance by Programming Languages.

2-3 크로스 브라우징 폴백

HTML5 마크 업뿐만 아니라 HTML5 API는 웹 표준이라 하지만, 실제 모든 브라우저에서 동시에 지원을 못 하고 있다. HTML5가 등장하기 전의 브라우저들이 그 대표적이다. 등장 이후 나온 브라우저 일지라도 그 버전별로, 벤더사에 따라 지원하는 내용이 다른 것이 현실이다. 이렇듯 브라우저마다 기능 동작 여부가 나뉘는 현상을 브라우저 파편화라고 한다. 이러한 파편화를 줄이기 위해 사용하는 방법을 폴백이라고 한다[6]. 이렇듯 폴백은 웹 시스템, 더 나아가 플랫폼을 구현하는데 중요한 개념이며, 웹 어셈블리를 활용할 때에도 필요한 기능이다. 자바스크립트와 웹 어셈블리에 각각 폴백을 수행했을 때, 그 성능 차이를 분석한 기존 연구가 있다. 다음의 그림 2의 (a)와 (b)는 성능 비교 결과를 나타낸다.



(a) 윈도우 OS 기반 폴백 기능 수행 속도



(b) MacOS 기반 폴백 기능 수행 속도

그림 2. OS 별 폴백 기능 수행 속도
Fig. 2. Comparison of FallBack Processing Performance by OS.

관련 연구들을 통해 알 수 있듯이, 자바스크립트와 웹 어셈블리의 성능은 상호 보완적이며, 서로 부족한 부분을 담당하는 역할을 한다. 따라서 대부분 시스템에서는 각각의 영역을 분류하고, 서로 필요한 데이터를 공유하며, 기능을 수행시키고 있다[7]. 하지만 대용량 데이터와 다양한 데이터 종류를 다루는 시스템 특성상, 이러한 과정에서 성능을 저하할 수 있는 요소들이 존재한다.

Ⅲ. 웹 어셈블리 통신 시스템 제안

3-1 시스템 개요

관련 연구와 많은 사례를 통해 알 수 있듯이, 웹 어셈블리는 자바스크립트와 상호 보완적으로 동작한다. 본 논문에서는 이러한 동작 형태에서 웹 어셈블리와 자바스크립트가 효율적으로 데이터를 통신할 수 있도록 하는 시스템을 제안하고자 한다.

기존에 자바스크립트는 WAS(web application server)에 내에서, 웹 시스템의 동적 페이지 구성 및 기능을 담당한다. 만약 웹 시스템의 기능이 네이티브 언어를 사용했을 때, 더 높은 속도 성능을 나타낼 수 있는(예를 들어 3D 게임, 이미지 또는 동영상 처리, 증강/가상 현실 기능 등) 것이라면 웹 어셈블리를 통해 해당 기능을 개발하여, 자바스크립트에서 임포트 하는 방식으로 사용되고 있다. 이러한 방식은 동적으로 페이지를 구성할 때, 많은 제약사항을 발생시킬 수 있으며, 다양한 서비스를 지원하고, 대용량의 데이터를 다루는 웹 시스템(혹은 플랫폼)의 특성상, 로드 되는 웹 어셈블리 모듈을 중앙에서 관리하고, 관리 주체를 통해 통신할 수 있는 구조를 제안한다. 제안하는 시스템에 대한 구성은 다음의 그림 3과 같다.

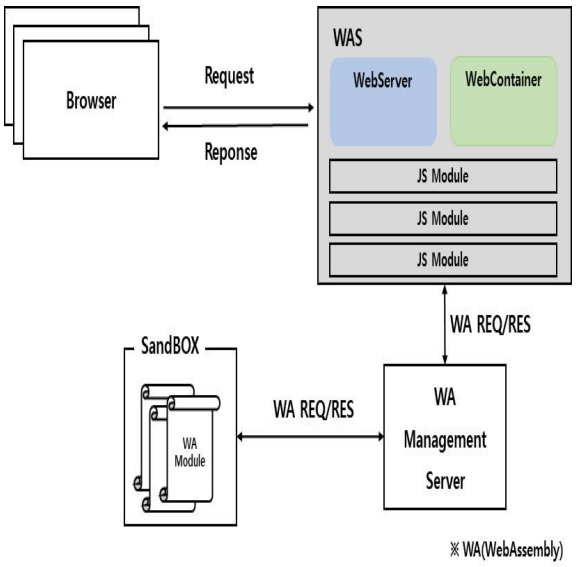


그림 3. 시스템 구조 제안
Fig. 3. Proposal of System Architecture.

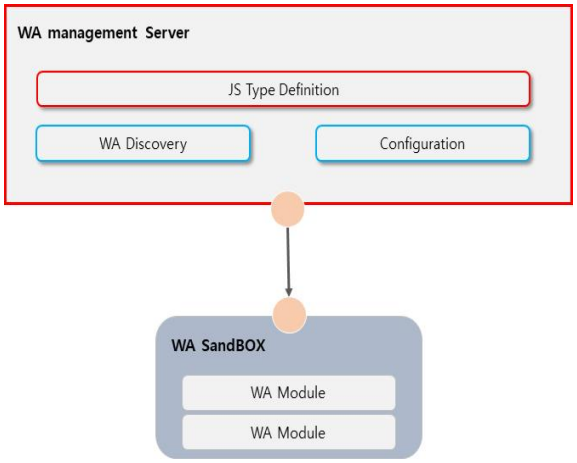


그림 4. 웹 어셈블리 관리 서버 구조
Fig. 4. Architecture of WebAssembly Management Server.

위의 그림 4는 웹 어셈블리 관리 서버의 구성을 나타낸다. 사용자는 WAS를 통해 사용하고자 하는 기능을 요청하게 된다. 사용자의 요청을 수립한 WAS는 정의된 웹 어셈블리의 타입을 관리 웹 어셈블리 관리 서버로 요청하게 되고, 요청을 받은 웹 어셈블리 관리 서버는 WAS로부터 받은 요청에 자신이 관리하는 웹 어셈블리 모듈을 결합하여 응답하게 된다. WAS는 응답받은 데이터를 사용자의 브라우저 응답 메시지로 전달하게 되고, 사용자 브라우저는 응답 메시지를 확인 후 기능을 동작하게 된다.

웹 어셈블리 관리 서버에는 WAS에서 사용하는 자바스크립트 모듈을 정의하고 있고, 이와 매칭 되는 웹 어셈블리 모듈을 요청에 따라 선택적(동적)으로 결합하는 역할을 한다. 또한, 자바스크립트와 결합 이력을 관리함으로써, 시스템(혹은 플랫폼) 확장에 따라 관리의 용이성을 제공할 수 있으며, 각기 다른 WAS에서 공유할 수 있게 할 수 있다. 다음의 그림 5는 브라우저가 WAS를 통해 웹 어셈블리 관리 서버로부터 기능을 요청하고 응답받는 과정을 나타낸다.

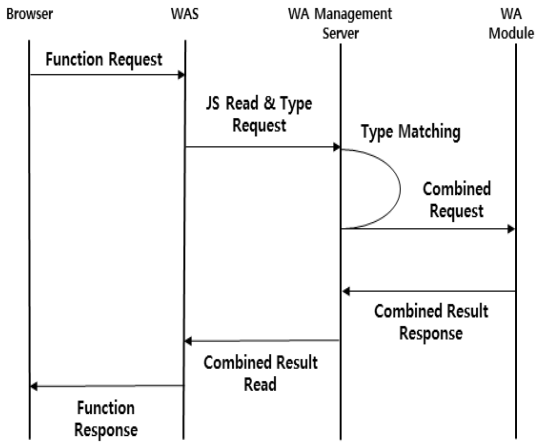


그림 5. 제안하는 시스템의 요청 처리 과정
Fig. 5. Function Flow of Proposal System.

Ⅳ. 결 론

본 논문에서는 웹 어셈블리와 자바스크립트의 상호 보완적 구조의 효율적 데이터 통신을 위한 시스템에 관한 연구를 진행하였고, 웹 어셈블리와 자바스크립트 간 데이터를 통신할 수 있는 시스템을 제안하였다.

앞서 서론에서 언급하였듯이, 웹 기술은 다양한 기기와 기술, 데이터와 사람, 데이터와 데이터 등을 연결하는 융합의 기술로 그 중요성이 날로 증대하고 있고, 이를 지원하기 위한 기술과 표준들이 연구 중이다.

기존에 웹 기술에서 동적 기능 구성을 위해 활용되던 자바스크립트로는 융합되는 기기, 기술들의 다양성, 통신하는 데이터의 용량을 다루기에 제한적이었고, 이를 해결하고자 연구된 것이 웹 어셈블리이다.

웹 어셈블리는 기존 네이티브 언어를 웹 환경에서 동작할 수 있게 하도록 바이너리 파일을 이용하여 컴파일하고, 이를 웹 엔진에서 동작하도록 지원해주는 기술이다. 하지만 기존의 연구 결과에서도 확인할 수 있듯이, 아직 웹 어셈블리로 자바스크립트를 대체하기에는 많은 제한 사항이 존재한다.

본 논문에서는 이러한 제한 사항을 완화하고자, 자바스크립트와 웹 어셈블리의 연동을 담당하는 서버를 별도로 구축하여, 서버를 통해 메타 데이터를 관리하고, 연동을 지원하는 시스템을 설계, 연구를 진행하였다.

웹이라는 기술은 특정 기술, 환경에 종속적이지 않고, 웹브라우저와 네트워크가 동작하는 환경이라면, 동작할 수 있도록 하는 표준 인터페이스의 역할을 하고 있다. 하지만 웹 어셈블리 기술을 웹브라우저 내로 종속시키면서, 웹이 지켜야 하는 독립적 구성, 이식성 등의 개념을 위배할 수 있는 위험성이 존재한다.

따라서 웹 어셈블리를 브라우저 외부에서 독립적으로 실행할 수 있는, 즉, 표준 인터페이스에 관한 연구가 필요할 것이며, 이를 위해 WASI(web assembly system interface)에 대한 연구를 진행할 것이다.

Acknowledgments

이 논문은 2019년도 정부(교육부)의 재원으로 한국 연구재단의 지원을 받아 수행된 기초연구사업 (No. 2018R1D1A3B07049722) 및 2019년도 한국기술교육대학교 연구 연간제 연구비 지원에 의하여 연구되었음

References

[1] Web Assembly Community Group, Web Assembly Structure, [Internet] Available: https://webassembly.github.io/spec/core/

[2] Web Assembly Standard Group, Web Assembly Specifications, [Internet] Available: http://webassembly.github.io/spec/.

[3] Google Chrome Labs, githubs of Squoosh, [Internet] Available: https://github.com/GoogleChromeLabs/squoosh.

[4] Google Developer Community Group, Replacing a hot Path in Your App`s Javascript with WebAssembly, [Internet] Available: https://developers.google.com.

[5] PSDDFKIT, A Real World Web Assembly Benchmark, [Internet] Available: https://pspdfkit.com.

[6] G. Steve, M. Fanning, Cross-browser interactivity recording, Playback, and Editing, U.S. Patent Application, Washington, pp. 187, 2011.

[7] J. Abhinav, “Not So Fast: Analyzing the Performance of WebAssembly vs. Native Code,” in *USENIX Annual Technical Conference,* Washington DC, pp. 107-120, 2019.

박 진 태 (Jin-Tae Park)
2019년 8월 : 한국기술교육대학교 컴퓨터공학과 (공학박사)
2018년 12월 ~ 현재 : KT 융합기술원 전임연구원
※관심분야 : 웹 표준, 웹 플랫폼 아키텍처, AI 융합 플랫폼

문 일 영 (Il-Young Moon)
2005년 2월: 한국항공대학교 항공통신정보공학과 (공학박사)
2015년 3월 ~ 현재 : 한국기술교육대학교 컴퓨터공학과 교수
※관심분야 : 무선인터넷 응용, 무선 인터넷, 모바일IP