2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)

Embedding Web Apps in Mixed Reality

Antti Peuhkurinen
Varjo
Helsinki, Finland
Email: antti.peuhkurinen@varjo.com

Tommi Mikkonen
University of Helsinki
Helsinki, Finland
Email: tommi.mikkonen@helsinki.fi

***Abstract*—Increase in processing capabilities, network connectivity, and other technical advances have enabled new ways to consume digital content. These include technologies such as virtual, mixed, and augmented reality, where new display type and multiple new input technologies are emerging. However, these systems are usually designed such that there is only the pre-installed main application, leaving the true application paradigm – analogous to desktop applications or mobile apps where the paradigms are well-established – inside the virtual reality still undefined. In this paper, we propose that web technologies form a potential dominant design that can be used to enrich mixed reality in a non-invasive fashion. More precisely, we go through the current technologies and open our thinking with examples from possible needs for the web application paradigm to work well with the virtual reality.**

***Keywords*—Web, Applications, Mixed Reality, Digital Twin, Programmable World**

I. Introduction

In the context of cyber-physical system (CPS), physical world actions are intimately linked with those appearing in cyber space. The digital twins paradigm [1] – where the same entity exists in the physical and in the digital world, available in a Virtual Reality (VR) environment – is a promising way to create applications to such setup, where the whole world becomes programmable [2]. The feasibility of such an approach is demonstrated for instance by Google Physical Web, an open approach to enable quick and seamless interactions with physical objects and locations [3].

Obviously, depending on the nature of objects and the role they play on applications, their features and characteristics may vary. Furthermore, when building applications that mix and match their properties – in analogy to web mashup applications [4] – their access rights may be different. However blending them in a mixed reality as independent applications introduces common characteristics such as modularity and security that must be considered at a generic level [5].

In this paper, we study how already existing web technologies can be used in application development in the context of digital twins forming a mixed reality environment. The paper builds on our earlier work on application composition in 3D context, mixed reality, and web applications [6], [7], [8]. The fundamental goal is to demonstrate how web applications can live inside a mixed reality context as full-fledged applications, representing physical items.

The rest of this paper is structured as follows. In Section 2, we present background technologies that can be readily used when realizing the above vision. In Section 3, we describe new technologies related to mapping physical objects to virtual worlds that we have prototyped when creating the technical artifacts presented later on in the paper. In Section 4, we create a concept design from the software system. In Section 5, we present a proof-of-concept implementation that demonstrates the proposed approach. In Section 6, we draw some final conclusions.

II. Background

In the following, we introduce the background of this paper. First, we consider physical systems that are needed for mixed reality, and then, we place the focus on mixed reality systems and web technologies we use as the implementation technique.

A. Physical System

In most of the urban areas, infrastructure for connectivity is commonplace. In addition, the devices which are mobile can access this infra wirelessly and can know their world pose (location and orientation). In Figure 1 we have different hardware that makes the connected and co-operating system possible as a whole. User's body area can have a smart phone, desktop, smart watch and similar devices that can communicate with each other. There is a separate group of hardware that makes possible the connectivity, like WiFi stations and operator networks. In addition, all of these devices are supported by the servers to provide data storage and processing capabilities. Techniques familiar from so-called liquid software [9] could be used to move the physical execution to more suitable location to optimize network lag and other resource load, for example between servers, IoT or user devices.

B. Mixed Reality and Web Technologies

Web applications have several advantages in comparison to their native counterparts when considering embedding them in a virtual world. Their execution model supports the inclusion of several applications inside the same host application in a fashion that is light-weight and scalable. More precisely, we exploit the following features:

Web application paradigm is a full featured application paradigm. This includes programming paradigm, installation and removal of the application when loading and unloading pages, and the application life cycle and the process isolation from the operating system when application is executed.

Interfacing host system resources is needed by the applications for more powerful applications. Various interfacing capabilities including both user interfaces, such as abstracted access to display, sound and input systems, file system, as

978-1-5386-5896-3/18/$31.00 ©2018 IEEE

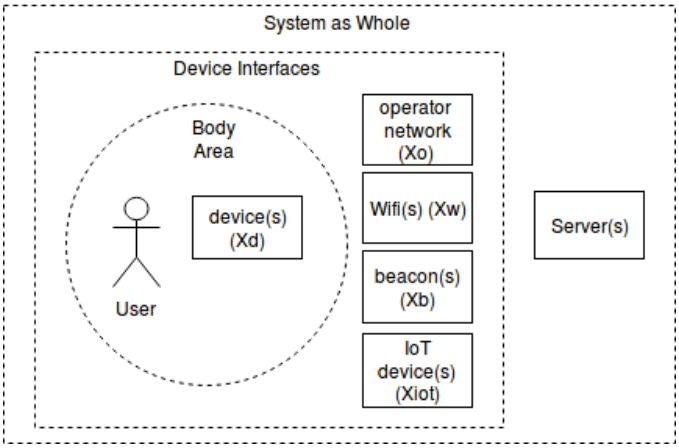

Fig. 1. Physical System

well as access to device specific interfaces. The latter may be offered only in a controlled fashion.

Application programming where flexible user interfaces can be created easily calls for a domain-specific, easy-to-use declarative language, such as HTML and CSS, together with techniques that support adaptation, similarly to responsive web design [10]. To create business logic a scripting language like JavaScript is needed. Usage of declarative language easies creation of mixed reality application. In case of really custom applications more native programming languages can be used to achieve wanted features and performance. In addition, for mixed reality 3D graphical assets open formats like Collada [11] can be used.

Mixed reality computing devices, such as Microsoft Hololens [12] or Magic Leap One [13], are currently bulky, but they are already stand-alone systems, and they can be used to run web applications. However, for simplicity in our research we have used desktop machines and mobile phones to mimic the future mixed reality hardware and its capabilities.

III. Extending Physical World to Virtual Reality

Additional technologies are need to create a fully programmable world, where objects would have their digital twins. Furthermore, some of the mixed reality smart objects can be fully virtual and also these virtual objects need similar technologies. In short term, many of these technologies will most likely emerge inside a silo owned by a single vendor, and it will take time before the technologies are mature enough to form a single new ”Internet of mixed reality”. In addition, some new technologies are needed. Next, we address some of the enabling new technologies.

A. World 3D Map Database

To identify and track places and objects easily, a lot of pre-processed tracking data is needed on servers. Data related to certain location, like point cloud of image features and labeled objects, could be obtained from the servers to help on the identification and tracking. This data could be collected by the network nodes, by user devices, glasses, cars, and phones, for instance. Maintaining real-time world map data requires that

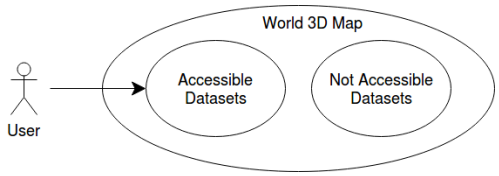

Fig. 2. World 3D Map Dataset Access

moving objects can send updates regarding their changes, or that they can be tracked by the system and updates in their status are sent back to the map. Privacy and security of the data must be kept in mind when creating the 3D map from the tracked data.

Figure 2 demonstrates how the world map for each user is formed from accessible dataset, although also other datasets exists that user cannot access. For example, public space most likely is seen by all users, but private spaces, like tracked insides of user's facilities, are seen only by the user. Different user accounts might be also available for different spaces – for instance home and work accounts might differ, and they might also be user specific. Furthermore, private tracking data could also be shared between user accounts and this needs permission management.

B. Digital Twins from Physical Objects

Automated pose tracking of digital twins is needed to place the virtual visuals of the object in accordance to the real world position [14]. Mixed reality device needs to get information from the pose of the object, including the object's world location and orientation. This is particularly crucial for moving objects. The object may obtain this information independently by itself, for example using network, beacons, GPS information, or its internal magnetometer, accelerometer, gyroscope, and air pressure sensors. The object can request other nearby smart objects, like the smart glasses, to provide information regarding its pose, or to could be tracked by system automatically and this information could be transmitted to become a part of the tracked object itself. This digital information world poses information created by the devices, and the world tracking and mapping data could be updated constantly to the world 3d map. This would make it to be a real-time representation of the world, including meta information like the digital twins at the end.

If multiple separate systems exist, there is a possibility to have multiple digital twins from the same physical object. This can happen for example if multiple vendors are creating their own systems for digital twin application paradigm.

Today, digital twin as a term is used mostly to the object that are already smart. However, the term itself can cover any object found from the real world. Next, we create categories which could be used to group any of the real world objects. Figure 3 describes four different categories for digital twin objects. In next subsections we describe these categories.

No Digital Twin Object. No Digital Twin Object is an object that cannot be recognized or tracked as an individual object. System is not able to recognize the unique instance or type of the object. Systems does not practically recognize this object as unique object; object seen to be part of a bigger

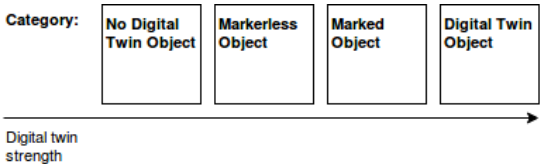

Fig. 3. Digital Twin Categories

object recognized or object cannot be strongly identified as a unique instance of some real world object.

Markerless Object. A markerless object can be identified as an individual object and can be tracked even if is not having any system friendly tag. Examples of such objects include objects that have been built before the Internet of mixed reality. System can recognize the unique object, or the object type, and add information to the object.

Marked Object. A marked object is an object that can be identified as a individual object and can be tracked from marker easily. Marker can be for example an electronic beacon or a visual tag made for the digital twin system purposes. Marked objects can be said to be Internet of mixed reality friendly.

Digital Twin Object. A digital twin object is connected to the system as an unique instance. For example, runs operating system, has sensors to interface with outside world, connected to web itself. It can have also a capability to host and run the digital twin software. These objects are capable themselves to actively update their own world pose to the system by using own sensors or actively requesting other nearby devices to track the object pose.

C. Mixed Reality Application Types

Application types that can be associated with mixed reality can take various forms. Some of the applications live in user space or are used as tools which have no real world associations and some of the applications are blending with the real world more. From the digital twin perspective, the mixed reality applications can be divided to two categories – the ones that are clearly based on digital twins and the ones that are not. Next, we describe these two categories.

Digital Twin Mixed Reality Applications. Digital Twin applications are extending the real world object. One special case also from a digital twin mixed reality application would be a virtual digital twin application. This term could be used for applications that exist visually in mixed reality in certain world location, and could also move while being tracked by the system, but do not have further resemblance in the real world. This kind of applications could be called as pure virtual digital twins. Their only physical world feature is that they are visualized in certain world location. This means that multiple users could still share the same instance of the virtual object.

Non Digital Twin Mixed Reality Applications. Normal applications could be considered to be not a copy of any real world's physical object or physical location. They can be attached to physical object or location or they are visualizing the types of objects from physical world. A sample non-digital twin application is an application that lives in user space floating next to user or is opened and closed anywhere by user, for example a desktop office application, or a shopping application that can show real world object types. It is also possible that the real physical digital twin object is associated with visualizations and associations that are not the same as the digital twin tracked in the real world.

D. Mixed Reality Application Paradigm

Since mixed reality is a somewhat new concept, it is only natural that many of the traditional computing concepts are still unrefined. In particular, this concerns paradigms that the applications must assume in the context of mixed reality. In the following, we address the key concerns associated with this task.

Multiple Simultaneously Visible Applications. When multiple different objects from different sources are showing their visuals to the user, there is a need for a secure and stable way of showing all of them simultaneously to the user. Also new input methods are needed. Some possibilities for this are for example eye-tracking, tracking of hands, voice, or separate hand hold controllers. Also, there is a problem how to manage multiple mixed reality 3D applications. Current browsers use commonly tabs to manage web applications. In mixed reality, many of the applications are visually blending with the real world. This needs special consideration when mixed reality application paradigm is defined. This problem we looked at in our earlier research. [8]

Visualization. Current web application graphics technologies lack de-facto declarative 3D layout routines, animations, asset formats and their loading. The current WebGL standard makes it possible to have 3D content easily on a website but it does not define high level declarative 3D user interface creation like HTML and CSS. This means that there is a need for new declarative programming paradigm for the mixed reality web applications.

Applications in World Location. To have mixed reality applications shown, there is a need to get the set of applications in current world location. This means first, that the device should know it's world position, and second, current world location's application list and their runtimes should be accessible. The user device visualizing the digital twins or some co-operating device nearby can also help to identify the digital twins in user nearby space. Especially this is needed for the moving objects that have their digital twin visualized to the user.

E. Digital Twin Runtime Hardware

To run the digital twin software there is a need for physical hardware that executes it. Currently, most of the real world objects don't contain a hardware of capable of running the mixed reality software. This means that most of the calculations need to be done by other objects if they are wanted to be having digital twins. For example, when user wears mixed reality glasses the device is tracking the environment and recognizing objects from it. The recognized object optionally has its application instance running on the server, or application is installed and run at the local device.

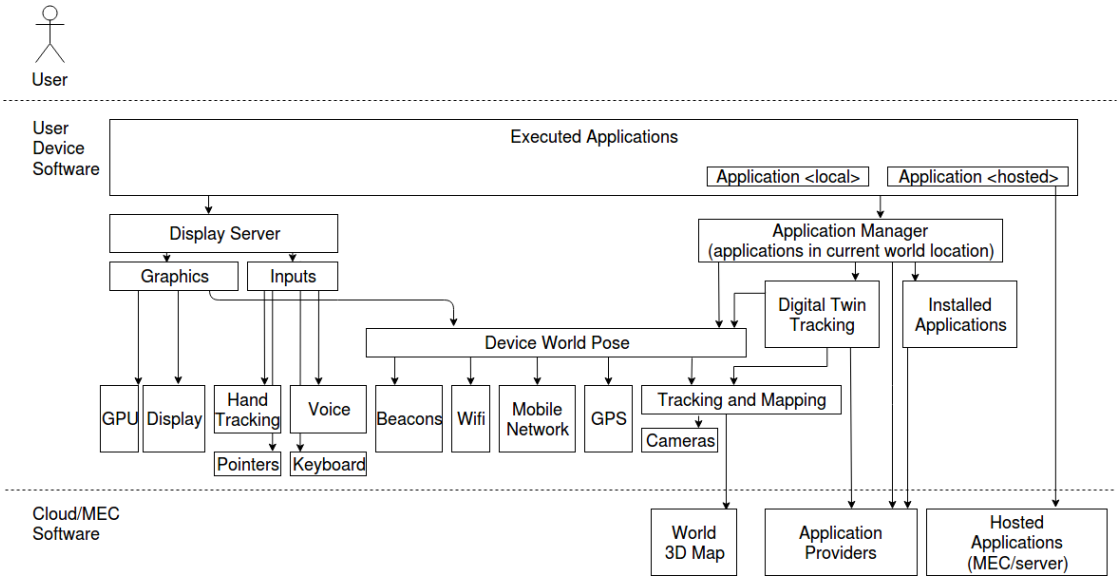

Fig. 4. Target Software Architecture

IV. Target Mixed Reality Software System

To make the mixed reality applications run correctly, there is a need for new software technologies. Figure 4 introduces the architecture of the target mixed reality software system. Next we describe the architecture.

The device world pose, the device orientation, location in the real world, and tracking of the real world plays a key role in the mixed reality software. This is because if we do not know how the physical objects and the user device is related to each other it is impossible to visualize virtual objects embedded to real world for the user. Furthermore, there can also be new digital twins or physical changes in the space that user's device and the system is unaware from. This mean that there is a need to do digital twin tracking and mapping of the space constantly by using all possible sensors. At known location we get current location's applications information from application providers. User might been also placing applications to the current location. Some of the user placed applications might be only visible to the user. Eventually we know the applications which are hinted by the world 3D map, suggested by the applications providers, and have been placed by the user to the space. This means that we know the applications to be run in current user's world pose.

To execute the current world pose applications we need to first either download them for the local execution or trigger host machines to execute them. Some of the applications could be already locally installed. The locally missing applications are fetched from their needed parts to the local device. Some of the applications might be based on runtime that is a separate package coming from application provider and some of the applications might be a complete package that can run on the system independently. In addition, the some of the applications might run themselves in a hosted mode where majority of the application business logic is running outside of the user's device. This kind of applications are especially in our interest because their nature resembles how the web works now on desktop and mobile platforms.

The running applications are visualized at the end for the user. We used our earlier shared scene idea in this research for efficient multi-application visualization [8] which is especially suitable for declarative visualizations.

V. Implementation

In our example implementation we focused to test the feasibility of a mixed reality web applications run on a separate host machine and how they could be programmed to work together. User device receives the 3D visuals of the application and user is able to give input back to the application. In programming view applications have an overlay where their outputs and inputs are visible to the user. Example implementation setup is illustrated in Figure 5. We used laptops to mimic the user device and the host machine. The application host servers and the programming server are run at the same host machine. For the connection we used WiFi. Display server draws the visuals from the application. The programming manager can show the application's inputs and outputs for the programming view in addition to other logical elements and connections added by the user. Programming manager also creates the connections informing related applications and connection server. Connection server runs the programmed logic when getting input. Application instances can give input to connection server and connection server can give input to related applications.

For the Display Server part and for the connection over the WiFi we have implemented a novel prototype system by extending Ubuntu 16.04 operating system. In addition to the native C++ interfaces we have enabled usage of JavaScript programming language on the application side for easy creation of mixed reality 3D applications. The applications can be also made declaratively with JSON where both layout and logic can be described.

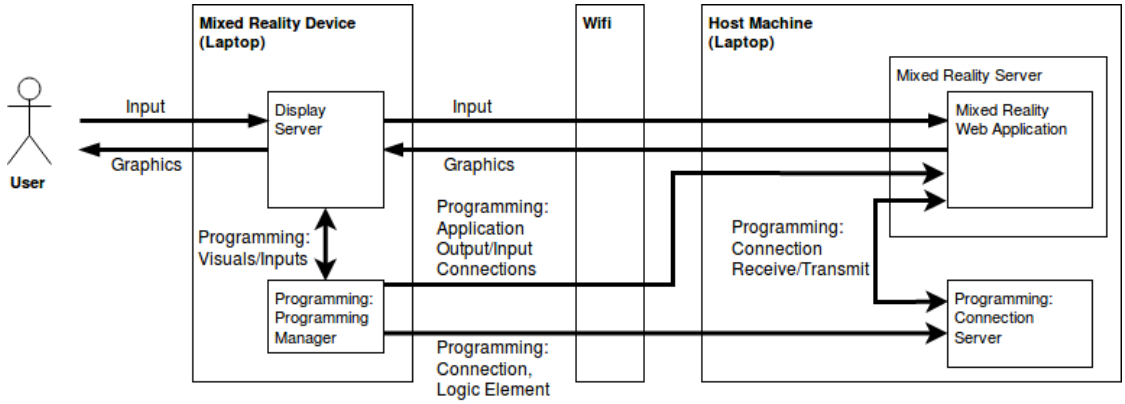

Fig. 5. Physical View of the Example Implementation

Fig. 6. Digital Twin of an Alarm Clock Visualizing Itself

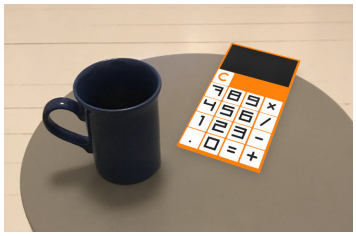

Fig. 7. Real Object next to Virtual 3D Object Visualized by Mixed Reality Application

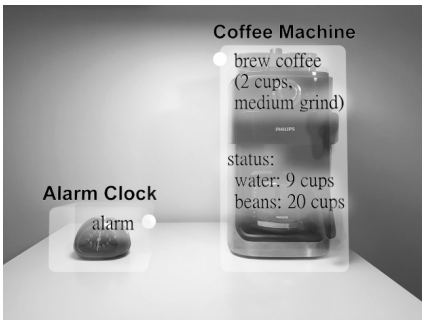

Fig. 8. Digital Twin Programming View

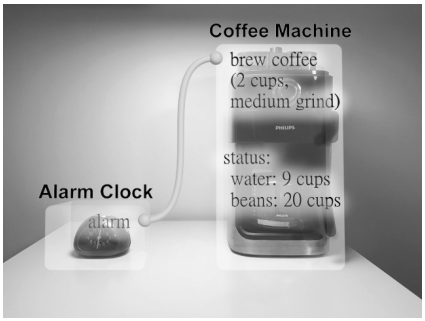

Fig. 9. Connected Digital Twins

A. Application Examples

An example digital twin is shown in Figure 6. The alarm application adds additional visualization of the alarm time and control to unset the current alarm set.

A pure virtual application is shown in Figure 7. In this example the calculator application visualizing a calculator does not have it's real world counterpart and thus it can be called to be pure virtual digital twin. Multiple users could see the same instance of the application running at the same time.

An example from a programming view enabling programmable world is shown in Figure 8. In this view each digital twin application is visualizing a logical element having inputs and outputs. The alarm clock's digital twin has an "alarm" output for alarm event. The coffee machine has a "brew coffee" input. In Figure 9 the user has connected the "alarm" output to the "brew coffee" input. In addition to the applications ready logical elements or elements having scripting language can be added to the programming view. These elements also contain inputs and outputs. This enables creation of complex logic having multiple digital twins.

B. Example Single File JavaScript Application

To create a 3D layout we enabled creation of the application's visual scene graph using JavaScript. The interface makes possible to create, update and remove the scene graph nodes. This can be used to create user interface layout and relationships between the resources. Creation of basic 3D shapes used in visualization can be done by describing them with text. Triangulated data can be used to create more complex shapes. For material creation text can be also used to describe basic material features like color or reflection factor by describing them by text. In addition, image files can be used in materials to texture the shapes. The location, rotation, and scale of the shape can be controlled using the transformation. Transformation is relative to the parent node enabling user

interface elements layout. Transformation can be described easily by text. In addition, event callbacks can be added to the nodes to make the application react to the user input or system events. Other components like animation or physics can be easily added to this programming paradigm.

Next, we have a short example JavaScipt source code creating an application. This application creates its visual scene having a single cube that changes its color when being pressed by the user. The client's main loop acts as the event handling loop for this application.

```
// example.js

// Init client with application volume size (w*h*d).
var client = new Client(2, 2, 2);

// Callback function.
var myCallback = "function pressed() \
{ client.updateMaterial(1, \"color: blue\")\; }";

// Init scene.
// Create a node under root node. Root node id is 0.
client.updateNode(1, 0);
// Create a cube mesh with size 2 x 2 x 2 to node 1
client.updateMesh(1, "cube: 2, 2, 2");
// Set node material to be diffuse color red
client.updateMaterial(1, "color: red");
// Set node location to origo
client.updateTransformation(1, "location: 0, 0, 0");
// Set "pressed" signal callback function
client.updatePressedCallback(1, myCallback);
// Flush scene udpates to screen
client.updateScene();

// Enter mainloop. Blocks here and receives events.
client.mainloop();
```

As described in Figure 5, when we run the above source code, we have the JavaScript being run by our native mixed reality platform at the host machine using V8 JavaScript engine [15]. The native platform then connects the client created to the user device's application manager. The application's visuals are sent to the user device. Then the Display Server at user device renders the visible mixed reality applications to the user. User inputs to the application are sent to the host machine where application logic can react to them.

C. Example Application Loading Scene from JSON File

In addition to JavaScript the application can be also implemented by using JSON file format. This is a declarative way of describing the application's visual layout and internal logic. Below, we list the corresponding above JavaScript example in JSON:

```
// example.json
{
  "volume": [ 2, 2, 2 ],
  "myCallback": "function pressed() \
  { client.updateMaterial(1, \"color: blue\")\; }"
  "node": {
    "id": 1",
    "mesh": "cube: 2, 2, 2",
    "material": "color: red",
    "transformation": "location: 0, 0, 0",
    "pressedCallback": "myCallback"
  }
}
```

VI. Conclusion

In this paper, we propose a mixed reality application paradigm enriched with digital twins and programmable world. Furthermore, we demonstrated how the current web technologies can be used to as part of the mixed reality application paradigm, thus gaining many of the benefits of web applications over their native counterparts [16]. Creating simple digital twin or other mixed reality applications would be easier by using declarative and scripting languages which are now being used to create the current web applications. In addition, it would be possible to run the mixed reality applications partially or fully in the cloud like current web applications.

In the process, we learned that the markerless tracking of the world, recognition and tracking of the physical objects, and the pre-processed database to help on these tasks plays a key role to make the system as whole perform adequately. In addition, we learned that the digital twin paradigm enables complex user input and programmability to any physical world object. Furthermore, as expected, digital twins from physical objects can provide more information that is possible in the physical world, but also can clutter the user's view if used in excessively large numbers.

References

[1] E. H. Glaessgen and D. Stargel. The Digital Twin paradigm for future NASA and US Air Force vehicles. 53rd Struct. Dyn. Mater. Conf. Special Session: Digital Twin, Honolulu, HI, US, pp. 1–14, 2012.

[2] A. Taivalsaari and T. Mikkonen. A roadmap to the programmable world: software challenges in the IoT era, IEEE Software, 34(1), pp. 72–80 2017.

[3] Google, Inc. The Physical Web. Available at https://google.github.io/physical-web/, referenced Jan 2., 2018.

[4] F. Daniel and M Maters. Mashups: Concepts, Models and Architectures. Springer, 2014.

[5] A. Taivalsaari and T. Mikkonen. Mashups and modularity: Towards secure and reusable web applications. In 23rd IEEE/ACM International Conference on Automated Software Engineering: Workshops (ASE Workshops'08). pp. 25–33. IEEE, 2008.

[6] A. Peuhkurinen, A. Fedorov, and K. Systä. Operating System Compositor and Hardware Usage to Enhance Graphical Performance in Web Runtimes. In International Conference on Web Engineering, pp. 381-388. Springer, 2016.

[7] A. Peuhkurinen, T. Mikkonen, and M. Terho. Using RDF data as basis for 3D Window management in mobile devices. Procedia Computer Science 5 (2011): 645-652.

[8] A. Peuhkurinen and T. Mikkonen, T. Mixed Reality Application Paradigm for Multiple Simultaneous 3D Applications. In Proceedings of 16th International Conference on Mobile and Ubiquitous Multimedia, Stuttgart, Germany, Nov. 26–29, 2017.

[9] A. Gallidabino, C. Pautasso, V. Ilvonen, T. Mikkonen, K. Systä, J.-P. Voutilainen, and A. Taivalsaari. "On the architecture of liquid software: technology alternatives and design space." In Proceedings of 2016 13th Working IEEE/IFIP Conference onSoftware Architecture (WICSA), pp. 122-127. IEEE, 2016.

[10] E. Marcotte. Responsive Web Design, A Book Apart, 2011.

[11] Khronos, Collada. Available at https://www.khronos.org/collada/, referenced Jan 21., 2018.

[12] Microsoft Corporation, Hololens. Available at https://www.microsoft.com/en-us/hololens/, referenced Jan 6., 2018.

[13] Magic Leap, Inc. One. Available at https://www.magicleap.com/, referenced Jan 6., 2018.

[14] R. Azuma. Tracking requirements for augmented reality. Communications of the ACM, 36(7), pp.50-51, 1993.

[15] Google, Inc. V8. Available at https://developers.google.com/v8/, referenced Jan 31., 2018.

[16] T. Mikkonen and A. Taivalsaari. Apps vs. open web: The battle of the decade. In Proceedings of the 2nd Workshop on Software Engineering for Mobile Application Development, pp. 22-26. Santa Monica, CA, USA, 2011.