applied sciences

Article

An Identity Model for Providing Inclusive Services and Applications

Lourdes Marco *, Álvaro Alonso and Juan Quemada

Escuela Técnica Superior de Ingenieros de Telecomunicación, Universidad Politécnica de Madrid, 28040 Madrid, Spain; alvaro.alonso@upm.es (Á.A.); juan.quemada@upm.es (J.Q.)
* Correspondence: lmarco@dit.upm.es

Received: 1 August 2019; Accepted: 7 September 2019; Published: 11 September 2019

Abstract: Information and Communication Technologies (ICT) need to be accessible for every single person in the globe. Governments and companies are starting to regulate products and services to ensure digital accessibility as a mandatory requirement. A recent example is the European standard EN 301 549, where the functional accessibility requirements for ICT products and services are defined. Especially on the Web, these standards must be integrated throughout the development processes, where the selected architecture models play an essential role. Starting from a model that is based on the OAuth 2.0 protocol, and that allows the complete delegation of authorization (so that an as a service access control mechanism is provided), this paper propose an identity model for providing inclusive services and applications. The model takes advantage of the users' profiles and their functional attributes to determine how to serve web interfaces to them in a specific service. Those attributes are entirely flexible, and can be defined linked to users' functional capabilities, or even a particular skill. We have implemented the proposed model as an extension of an existing open source Identity Manager and tested it with a real use case deployment. We conclude that the proposed solution enables a new identity paradigm that allows service providers to design their interfaces satisfying the diversity requirements in terms of design and development.

Keywords: web accessibility; identity; OAuth 2.0; functional diversity; functional Attributes

1. Introduction

Throughout history different models of disability have been imposed [1]: (1) a moral model which regards disability as the result of sin; (2) a medical model which regards disability as a defect or sickness; (3) a rehabilitation model which regards the disability as a deficiency that must be fixed; (4) a disability model, under which the problem is defined as a dominating attitude and the strong tendency for people to generalize about all persons with disabilities. We can observe how the problems of disabled people have been historically explained in terms of divine punishment, moral failing or even biological deficit. However, other perspectives such as the British political network *Union of Physically Impaired Against Segregation* (UPIAS) ones consider a social model, in which civil rights, rather than charity or pity, are the way to solve the disability problems [2].

Disability needs to be accurately defined, especially because of its historical link with medicine and its social condition. Based on World Health Organization's (WHO) International Classification of Functioning, Disability, and Health (ICF) (WHO ICF: https://www.who.int/classifications/icf/en/) conceptualisation, authors of [3] propose a new definition of disability in which the individual is not the axis of the problem: "Disability is a difficulty in functioning at the body, person, or societal levels, in one or more life domains, as experienced by an individual with a health condition in interaction with contextual factors". They believe this definition includes all aspects of disability, highlights the interactive dynamic nature of disability, and acknowledges the equally important roles of the person's state of health and environmental factors in the production and mediation of the disability experience. Some other approaches have emerged in recent years to challenge the historical oppression and exclusion of disabled people. For instance, authors of [4] propose the use of the new term *functional diversity* to refer to disability.

Trying to quantify the significance of this circumstance, and according to the WHO, around 15 percent of the world's population have some sort of disability [5]. In a digital world as the one in which we live, this fact is worrisome enough to affirm that Information and Communication Technologies (ICT) need to be accessible for everybody. The only way of ensuring an accessible world for all is to bear in mind everybody when designing services or applications. This is the main focus of digital accessibility. The World Wide Web Consortium (W3C) (W3C: https://www.w3.org/), in its official definition of accessibility, explains that the Web can remove barriers to communication and interaction that many people face in the physical world. However, they add that when web sites, applications, technologies, or digital tools are badly designed, they can create even more barriers that exclude people from using the Web (W3C Accessibility definition: https://www.w3.org/standards/webdesign/accessibility).

In February 2019, Web Accessibility In Mind (WebAIM) conducted an accessibility evaluation of the home pages for the top 1,000,000 web sites. In that sample, 97.8% of the sites had automatically detectable Web Content Accessibility Guidelines (WCAG) failures, with an average of 59.6 errors per page [6]. These data show that the actual WCAG 2.0 A/AA conformance level for the home pages for the most commonly accessed web sites is below 1%.

Authors of [7] explain how nowadays the most common accessibility approaches for designing web content in general are those whose methodological basis are grounded in reactive principles, mainly based on either manual or automatic evaluation methods. As a consequence, the designed services and applications are not fully optimal with regard to web accessibility, in most cases increasing social exclusion. As authors conclude, it is absolutely necessary to move forward and propose solutions that contemplate inclusive design from the early stages of development, such as accessible templates or even software architectural models that reflect not just the visual layer of the Web, but the most suitable structure for each diversity case.

In this paper we present a model for providing services based on the functional attributes that a certain person can possess. The model is based on well-known identity and access control mechanisms such us OAuth 2.0 [8] and Attribute Based Access Control (ABAC) [9]. As we will explain with detail later, the existing adaptive user interface researches mainly base their adaptation decisions on process automation and context-of-use (an assembly consisting of user, platform, and environment aspects). However, very few of them are taking in account inclusion as a fundamental premise. On the other hand, other approaches that aim to solve the problem of providing accessible interfaces, just adapt the standard ones to the necessities of the users by following a set of rules and recommendations. Nevertheless, we defend a new approach in which the interfaces adaptation is performed before serving them and based on the identity of the users. Thus, developers can design specific alternatives based on the attributes of the users, concretely on their functional attributes. The classification of these functional attributes is presented above, in Section 4.

Thanks to the model we propose, services have access to the users' attributes related with their functional capabilities before serving web pages. This enables a new paradigm to develop applications in which web interfaces can be designed thinking in the different functional capabilities of the users. Then, when users access the application, a specific interface is provided depending on their capabilities.

We have implemented the proposed model as an extension of an existing open source Identity Manager. Moreover, we have validated the proposal by designing and deploying a use case that exploits the advantages of the new model by providing adaptive web user interfaces for the heterogeneity of the users' profiles. The latter also means that the interfaces are able to adapt itself or to be adapted to a specific context or context change. Specifically, in the implemented use case we have designed three different interfaces, adapted to users with vision, cognitive and hearing difficulties. Depending on the attributes defined for a specific user, a different interface will be provided.

The document is structured as follows. It starts with Section 2, that goes through the literature and reviews related work on web accessibility rules and evaluation methods. Section 3 presents an overview of the accessibility requirements, where functional performance statements for ICT services are going to have a key role. In Section 4 we expose our solution, an OAuth 2.0—based Identity model for providing accessible services and applications. Section 5 presents the validation of the proposed model. Finally, Section 6 concludes with some remarks and an outlook on future work.

2. Related Work

According to the definition of the International Organization for Standardization (ISO) (ISO: https://www.iso.org/) "accessibility is the extent to which products, systems, services, environments and facilities can be used by people from a population with the widest range of characteristics and capabilities, to achieve a specified goal in a specified context of use" [10]. For its part, web accessibility allows a web service to be visited and used successfully by as many people as possible.

There are many definitions of web accessibility in the literature [11,12]. Some of them refer to equal access for all people, and others specifically refer to disabled people or focus on 'usability' properties. Some authors affirm that there are also different views on the relationship between accessibility and usability [12,13]. According to [14], accessibility not only implies the need to provide access, but also the need to facilitate use.

As the framework for this research, this section provides an analysis of the situation of web accessibility rules and guidelines. It also discusses related studies available in web accessibility evaluation methods. As we will explain with detail later, the solution we propose in this manuscript uses a set of users' attributes related with their functional capabilities to determine how to serve web interfaces to them. For achieving this, users have to previously provide their identity profile to the platforms which is typically done using Identity Managers. Therefore, to have a complete view of the state of the art with regard this research, it is important to discuss how Identity Manager solutions achieve accessibility. Finally, we present a comparison between the different approaches we can find in the literature for adapting user interfaces to context changes.

2.1. Web Accessibility Rules

The W3C is probably the largest entity in terms of guarantees when we talk about the World Wide Web. It is an international community where people work together to develop Web standards. Led by Tim Berners-Lee, their mission is to drive the Web to its full potential following two main design principles: *Web for all* and *Web on everything*.

For almost 20 years, the W3C Web Accessibility Initiative (WAI) (W3C WAI: https://www.w3.org/WAI/) has been the main driver to develop standards and support materials to help to understand and implement web accessibility. Recognized as a *de facto* standard, the WCAG 1.0 [15] were published by WAI as a W3C Recommendation on May 1999. WCAG 1.0 are a set of recommendations in order to make Web content more accessible for people with disabilities in the first place, but also for everyone else, ensuring the access through every user agent or device.

The second version of WCAG (WCAG 2.0) [16] was published in December 2008 and became an ISO standard in October 2012 (ISO Standard WCAG 2.0: ISO/IEC 40500:2012). The latest version of the WCAG is 2.1 [17] and was published in June 2018. The new versions extend WCAG with the definition of additional criteria to address low vision, cognitive, language, and learning disabilities, as well as mobile devices support. Although it is possible to conform either to the first or to the second version of WCAG (or both), the W3C recommends that new and updated content use WCAG 2.1. The W3C also recommends that Web accessibility policies reference WCAG 2.1. WCAG 2.1 is backwards compatible with the previous version. Therefore, web pages that conform to WCAG 2.1 also conform to WCAG 2.0.

EN 301 549 [18] is the European Standard on web accessibility. Accessibility requirements applicable to ICT products and services are specified in this document. In addition, a description of the test and evaluation methodology for each requirement is included, to ensure that the public use of the document within Europe. The main objective of the document is to produce a European Standard that sets out in a single source. Besides, is intended to be a detailed and practical report applicable to all Information and Communication Technology (ICT) products and services.

A year ago, the three official European Standards Organizations (ESOs), Comité Européen de Normalisation (CEN), Comité Européen de Normalisation Electrotechnique (CENELEC), and European Telecommunications Standards Institute (ETSI), published an updated version of *EN 301 549*, "Accessibility requirements for ICT products and services", which most significant change is the adoption of the W3C WCAG 2.1 for web content, electronic documents, and non-web software, such as native mobile applications. With these updates, *EN 301 549* addresses the requirements of the EU Web Accessibility Directive in terms of websites and mobile applications in Europe.

The Rehabilitation Act Section 508 is a section in a federal law in United States that extends and revises the authorization of grants to States for vocational rehabilitation services. The law applies to all Federal agencies when they develop, procure, maintain, or use electronic and information technology. Related to the latter, the federal agency Access Board (Access Board: https://www.access-board.gov) promotes equality for people with disabilities and is responsible for developing ICT accessibility standards to incorporate into regulations that govern Federal procurement practices. On January 2017, the Access Board issued a final rule that updated accessibility requirements covered by Section 508 and went into effect on January 2018. Thus, the requirements are harmonized with other guidelines and standards both in the U.S. and abroad, including standards issued by the European Commission and with WCAG 2.0. The technical standards included in the Section 508 of the law cover the following products and systems:

- Software Applications and Operating Systems.
- Web-based Intranet and Internet Information and Applications.
- Telecommunications Products.
- Videos or Multimedia Products
- Self Contained, Closed Products.
- Desktop and Portable Computers.

2.2. Web Accessibility Evaluation

A wide discussed topic when it comes to web accessibility is the importance of evaluating web pages. As can be proven by the literature, there has been a significant amount of research on web accessibility evaluation [19–22], especially with regard to the automation of the evaluation process [23]. However, it is not clear whether web accessibility needs to be evaluated by combining different techniques [24]. There are a number of different evaluation methods that can be used to assess the accessibility of a web page [25]. The following subsections explain the five categories [26] in which we can summarize these methods.

Inspection methods. Consists on the inspection of a web page for evaluating its accessibility by an evaluator. The most widely used inspection method is "Conformance Review", where the evaluator uses a set of accessibility guidelines (usually WCAG) that focus on possible accessibility problems and has to decide if a page or web site complies to those requirements.

Automated testing. This evaluation method involves an evaluator using an automated accessibility tool to check conformance of a web site with the accessibility principles encoded in that tool [27,28]. There are many different tools available, whether web-application tools (that allow for the review of the accessibility of a web page through its URL) [21], browser add-ons (software components installed frequently as add-ons in web browsers) or linting packages (linter tools to analyze the code of an application to detect errors, semantic issues, or in this case, accessibility issues).

Screening techniques. This method is based on using a website in a way that some sensory, motor or cognitive capabilities of the user are artificially reduced. Thus, it is possible to simulate some of the conditions that are common for people with disabilities [29].

Subjective Assessment. This method is a process where an evaluator hires a panel of users who are asked to explore/use a website autonomously and send back their opinions; the evaluator then collects such feedback to determine the accessibility level of the web page [30].

User Testing. This method consist of a process where formal or informal experiments are set up with real users, who are individually asked to perform goal-free or goal-oriented navigation on a web site, and whose behavior is observed by evaluators.

2.3. Inclusive IdM

As stated before, the situation of web sites with regard accessibility is disappointing [6]. When it comes to Identity Management (IdM), the scenario becomes even worse. Although there are many international and national standards like digital certificates [31], e-Government [32], or X.509 digital authentication [33], there are huge problems to implement a complete cross-browser scenario for authenticating and authorizing users. Besides that, digital Identity Management assumes that all users are equal and have the same functional attributes to deal with an identification process.

For solving this problem, [34] provide a conceptual framework for inclusive IdM, where several IdM approaches, methods and techniques are analyzed and discussed from the perspective of inclusive design. They talk about an Inclusive Identity Management approach (IIdM) that implies a systematic integration of usability and accessibility concerns in the design and development of Identity Management systems. What we can extract from there is that the field of IIdM "must take the goals, insights and concerns of various disciplines into account, especially those of universal design, security engineering and privacy/legal issues". This proposal throws interesting conclusions to take into account:

- Authentication methods have to be adapted to the specific requirements of users.
- New systems that avoid storing personal disability profiles in persistent databases should be explored.
- The notion of adaptive, user-profiling information systems, and the usability and accessibility of user-controlled identity management are crucial.

A recent proposal [35] has studied a way to break the barriers that the use of digital certificates implies using a USB cryptographic token that can conceal from users all complexity entailed in access to certificate-based applications, while assuring the required security. But nevertheless, this can be a complicated solution to solve the current platform-independent problem.

2.4. Adaptive User Interface Approaches

The User Interface (UI) layer is considered one of the key components of software applications since it connects users to the functionality. UI designers face a number of challenges while designing a UI for interactive systems due to the heterogeneity, that can be defined as a multiplicity of end users, computing platforms, interaction modalities, user working environments, and contextual variability.

An adaptive User Interface (adUI) is a UI that is able to adapt itself or to be adapted, automatically at runtime, to a context change [36]. A context of use varies according to the properties of the users, the platforms and the environment of interaction. This includes the properties related with the functional capabilities of the users. Therefore, a well designed adUI have to be able to adapt itself to the diversity of users in terms of functional capabilities.

However, traditional software architectures fail to provide mechanisms for runtime adaptation to the requirements of users and services. Recent development approaches such as Dynamic Software Product Lines (DSPLs) attempt to face the challenges of the systems' dynamic conditions. An overview of the the state of the art and current techniques that attempt to face the many challenges of runtime variability mechanisms in the context of DSPL can be found in [37].

In this scope, authors of [38] define a DSPL approach for the development of a family of context-adaptable user interfaces. The DSPL paradigm exploits the knowledge acquired in software product line engineering to develop systems that can be context-aware, or runtime adaptable. A similar approach is proposed by authors of [36], in this case mainly focused on the design of adaptive User Interfaces according to SPL engineering.

Other approach to overcome the heterogeneity with regard UI design is the Model-Based User Interface Development (MBUID) paradigm, which defines a set of models to generate multiple interfaces for different context of use. Authors of [39] define a system which can automatically generate interfaces adapted to devices, tasks, users' preferences, and users' abilities. On the other hand, in order to address gaps and limitations such as the integration in legacy systems, Ref. [40] presents Role Based UI Simplification (RBUIS) as a mechanism for increasing usability through adaptive behavior by providing end users with layouts based on the context of use. The work proposed in [41] describes another architectural model approach for generating user interfaces by adding adaptive behavior at runtime and by suggesting the primary heuristics for the practical deployment of UI rendering. For its part, Ref. [42] provides a domain and device-independent model-based adaptive user interfacing methodology that is dependent on the evaluation of user context and user experience (UX). Besides, authors of [43] present an overview of Model-driven engineering systems and a set of criteria to evaluate their strengths and shortcomings, categorized under architectures, techniques and tools.

Table 1 shows an overview of the proposals exposed above, summarizing their main characteristics. The table includes the following aspects of the approaches: (1) if the approach uses a user model (e.g., saving users' profile or recording their navigation activity) to adapt the UI, (2) if the approach uses previously defined roles to adapt the UI, (3) if the approach proposes the delegation of the decision to a centralized component or relies in a third-party entity to get the needed data (e.g., the user profile) to adapt the UI, (4) if the approach proposes the automation of processes to generate adapted UIs, (5) if the approach is proposed as an inclusive solution (taking accessibility into consideration) and (6) the development approach type (Dynamic Software Product Lines (DSPL) or Model-Based User Interface Development (MBUID)).

Table 1. AdUI approaches comparison.

Reference	User Model	Roles	Delegation	Automation	Inclusion	Approach
Gabillon et al. [36]				X		DSPL
Sboui et al. [38]				X		DSPL
Gajos et al. [39]				X	X	MBUID
Akiki et al. [40]	X	X		X		MBUID
Balme et al. [41]				X		MBUID
Hussain et al. [42]	X			X	X	MBUID

As we can see, the works of Akiki et al. and Hussain et al. create a user model as a key for the user interfaces adaptation. The rest of works uses other techniques related to the user, or even other conditions of the context of use to adapt the interfaces, for instance collecting user feedback automatically or the type of device. Akiki et al. offer also the possibility of defining roles to discriminate users but they do not take into account functional attributes for designing inclusive interfaces. The proposal of Hussain contemplates inclusion and is the proposal that better fits the requirements we have exposed. However, the user model is defined and managed internally in each specific service. Delegating the user model management to a centralized identity manager could improve their solution by allowing users to define their profiles once and using them in all the services they consume. The feature of automating the generation of UIs is available in all the studied approaches. However, we consider automation can be detrimental to the objective we pursue.

In contrast, we propose a solution in which web interfaces are individually designed taking into account ranges of capabilities with regard different functional attributes such as capacity of vision, hearing or manipulation. Our proposal bases the decision of which interfaces provide to users on a user profile defined in a centralized identity manager and that includes the functional attributes to ensure inclusion. Taking advantage of OAuth 2.0 protocol services can delegate authorization to such Identity Manager and retrieve the user profile to take the decisions. Finally, we propose to integrate the model with an advanced Attribute and Role based Access Control architecture to enhance the services functionalities and to enforce security and privacy.

3. Accessibility Requirements for ICT Services

As explained in the previous section, the existing approaches to provide accessible web services mainly offer methodologies and tools to evaluate whether the services are well adapted to the functional diversity of users. In this work we present a new paradigm in which interfaces have to be developed accomplishing the accessibility requirements from the first steps of the design phase. Therefore, to start designing our solution we have to analyse those requirements first.

The European Standard [18] mentioned above is presented as a document to specify the functional accessibility requirements applicable to ICT products and services. It is composed of fourteen clauses and five annexes:

- Clauses 0 to 3 include background information.
- Clause 4 contains functional performance statements.
- Clauses 5 to 13 provide specific testable criteria for accessible ICT.
- Clause 14 is about conformance.
- Annexes A to E contain tables with requirements, normative and further resources for cognitive accessibility.

The document places its focus on two main aspects: *functional performance statements* (clause 4), referring to the high level user needs applicable across technologies for all ICT procurements; and *functional accessibility requirements* (clauses 5 to 13), regarding the detailed accessibility requirements for different technologies [44].

3.1. Functional Performance Statements

Functional performance statements are described in clause 4, and they are based on permanent or temporary impairments. It is important to take into account that users with multiple impairments could need the combination of different accessibility solutions. Table 2 shows a summary of the statements that will be more fully explained below.

Table 2. Functional performance statements.

Usage	**ICT Needs to Provide**
Without vision	Voice User Interface Tactile User Interface
With limited Vision	Magnification Control of contrast Reduction of RFV
Without perception of color	No color-meaning Color-coded (tags)
Without hearing	Subtitles Tactile User Interface
With limited hearing	Audio clarity No background noise Increase volume range
Without vocal capability	Orally-generated sounds Keyboard/pen Touch User Interfaces
With limited manipulation or strength	One-handed operations Sequential key entry Voice User Interfaces
With limited reach	Target height or position
With limited cognition	Simpler Easier to use Timing, errors, focus
Features for accessibility	Privacy

3.1.1. No Vision

If ICT provides visual interfaces, some users need at least one mode of operation that does not require vision, like audio and tactile UIs.

3.1.2. Limited Vision

Some users will need products and services to provide features that enable them to make better use of their limited vision. These features can be for instance magnification, reduction of required field of vision and control of contrast. In the event of depth perception needs, we can provide additional methods of distinguishing between the UI significant features.

3.1.3. No Perception of Colour

With visual interfaces, some users will need the ICT to afford modes of operation that do not require perception of colour. If the UI components are colour-coded, we can provide additional methods of distinguishing between the features, like tags.

3.1.4. No Hearing

In the event of auditory modes of operation, end users need ICT to supply at least one interface that does not require hearing, like visual and tactile UIs.

3.1.5. Limited Hearing

Some users will need enhanced audio features to be provided for their products and services. Some good examples of this are improvements of the audio clarity, reduction of background noise, increased range of volume and greater volume in the higher frequency range.

3.1.6. No Vocal Capability

With vocal input from users, some of them will need the ICT to afford at least one interface that does not require them to generate vocal output. This clause covers the alternatives to the use of orally-generated sounds, including speech, whistles, clicks, etc. Keyboard, pen or touch UIs can contribute meeting this clause.

3.1.7. Limited Strength or Manipulation

If manual actions are required, users will need the ICT to provide alternative actions not requiring manipulation or hand strength. For instance, operations that require fine motor control, path dependant gestures, pinching, twisting of the wrist, tight grasping, or simultaneous manual actions. We can provide one-handed operation, sequential key entry and speech user interfaces to meet this clause. Some users have limited hand strength and may not be able to achieve the level of strength to perform an operation. Alternative UI solutions that do not require hand strength may contribute.

3.1.8. Limited Reach

If ICT products are free-standing or installed, the operational elements will need to be within reach of all users. We can consider the needs of wheelchair users and the range of user statures in the placing of operational elements of the UI to meet this clause.

3.1.9. Limited Cognition

Certain users will need products and services to provide features that make the interfaces simpler and easier to use. That means that we must also take into account the needs of persons with limited cognitive, language and learning abilities by providing adjustable timings, error indication and suggestion, or a logical focus order.

3.1.10. Privacy

End users will need their privacy to be maintained when using those ICT features cited above.

3.2. Functional Accessibility Requirements

Functional accessibility requirements are applicable to the functional performance statements above. As we have seen, Clauses 5 to 13 provide specific testable criteria related to technical requirements for different kinds of ICT, starting with generic requirements in clause 5. The latter explains that computers that do not allow users to adjust settings or install software are functionally closed, and because of this users are precluded from adding peripherals or software in order to access that functionality. Where ICT has closed functionality, it shall meet the requirements set out in clauses 5.2 to 13, as applicable.

From 6 to 13, the ICT requirements clauses are:

- Clauses 6. ICT with two-way voice communication
- Clauses 7. ICT with video capabilities
- Clauses 8. Hardware
- Clauses 9. Web
- Clauses 10. Non-web documents
- Clauses 11. Software
- Clauses 12. Documentation and support services
- Clauses 13. ICT providing relay or emergency service access

As we can observe, requirements in clause 9 apply specifically to web pages. In this regard, the standard describes that conformance with W3C WCAG2.1 Level AA is equivalent to conforming with all of clauses 9.1 to 9.4 of the document.

The *EN 301 549* contains a wide range of requirements to cover a variety of ICT solutions. The list of requirements can be consulted in the Table B.2 of the Annex B in the EN 301 549 document (EN 301 549 document: https://www.etsi.org/deliver/etsi_en/301500_301599/301549/03.01.01_20/en_301549v030101a.pdf). However, the text highlights that it is necessary to understand which requirements are relevant for a specific service in a specific situation or context. Overall, this may result in a problem when trying to specify a requirements list. Although the previously mentioned tables provided by the standard (Annexes A and B) to easily understand the connection between requirements and functional performance statements, services development teams must take into account every functional performance statement to concrete an accessibility requirements list for each service. If we could determine what are the specific needs of the user who is consuming a service, the list of specifications could be divided into as many lists as needed, providing two main advantages: services can be customized for each specific user that is consuming it, and a flexible, much lighter and optimized scenario can be achieved.

4. Proposed Model

We have analyzed the accessibility requirements of ICT services and the existing related works that try to solve the problem of providing Web universal access. We can conclude that the existing solutions provide accessible interfaces by adapting the existing ones following a set of rules and recommendations. As justified above, this approach is not enough for covering the accessibility requirements of ICT services. We need to propose solutions that include the accessibility paradigm from the very beginning of the web interfaces design phase.

However, once the accessible interfaces have been designed, service providers need tools and criteria to decide which users need to consume each one. In this Section we propose a model that allows services to provide different web interfaces based on the identity of the users. We also propose a set of functional attributes related with the capacities of the users and that could be used for deciding the type of interface each of them needs. The model is based on the OAuth 2.0 protocol to let the users defining their identity attributes in a centralized way. Therefore, third party services can access the public information of users to serve the adapted web interfaces.

4.1. Functional Attributes

We cannot assume people have always same identity attributes. Each person is unique in his or her own way of consuming information because of a number of personal attributes, which in this text we will define as *functional attributes*. Our proposal is based on the use of users' functional attributes to decide how to provide the web interfaces in services and applications. Taking the functional performance statements (reviewed in Section 3.1) as starting point, Table 3 defines the functional attributes we propose. This set of attributes will be included in the user model in order to extend the users' profile.

Table 3. Functional Attributes.

Attribute	%	Interface/Feature
Vision	≥85	Voice User Interface Tactile User Interface
	1–84	Magnification Control of contrast Reduction of RFV
Color Perception	1–100	No color-meaning Color-coded (tags)
Hearing	≥85	Subtitles Tactile User Interface
	1–84	Audio clarity No background noise Increase volume range
Vocal Capability	1–100	Orally-generated sounds Keyboard/pen Touch User Interfaces
Manipulation Strength	1–100	One-handed operations Sequential key entry Voice User Interfaces
Reach	1–100	Target height or position
Cognition	1–100	Simpler Easier to use Timing, errors, focus

Typically, a user profile in a web application is composed by attributes like the username, the email address, a photography, the age, the postal address or the phone number. We propose to extend this user user profile with the attributes of Table 3 to include characteristics with regard the functional capabilities of the user. Thus, in the same way users have a value assigned to, for instance, the attribute *age* now they will have also a value assigned to attributes like *vision* or *hearing*. The values assigned to the functional attributes can be self-declared by users or retrieved from official national organizations such us medical or social services. However, this manuscript aims to define the needed functional attributes and to provide a model that enables their use by services to provide accessible interfaces. The way the value of each attribute is assigned in a user profile is out of the scope of this research. Of course, the set of attributes we propose can be enlarged adding new ones depending on the requirements of each service or the laws of each country.

Each functional attribute does not necessarily coincide with a disability caused by a single disease, and is not linked to a specific associated interface or assistive technology. For example, a blind person will require support for using ICT without vision. In this case, in its personal attributes, the attribute *vision* would be defined with a value of 100%. This would imply the use of a voice user interface for consuming the service.

In a similar way we can assess the example of an elderly person with Parkinson's disease, who has severe tremor and will need assistance for using ICT with limited manipulation. In this second case, in its personal attributes, the attribute *manipulation* would be defined with a value of *80%*. This implies the use of a voice user interface for consuming the service too.

Based on the functional performance statements above, and the Spanish law [45] that establishes a procedure for the recognition, declaration and qualification of degrees of disability, we have set a range of values for each functional attribute, and the features or type of interface recommended to each case (Table 3).

However, the decision of providing each type of interface will always depend on the development requirements and eventually on the specific laws of the country of implementation. For instance, in the case of Spain, the aforementioned legal text sets out the rules for the qualification of deficiencies and disabilities, always expressed as a percentage of disability (being zero the absence of disability and 100 the maximum recognized).

To initially validate the proposed functional attributes, we have established a parallelism between the deficiencies of the body's organ systems that are defined in the text, and the possible personal attributes that can determine the personal situation of disability facing the ICT.

Table 4 clearly shows that *Manipulation, Strength and Reach* are the most frequent functional attributes due to the fact that they are affected by the greatest number of diseases. Even ear problems can lead to mobility problems or reach. For their part, *Vision and Color perception* are both under the umbrella of the Visual system. Speech, talk and voice disorders are mainly related to *Vocal capability and Cognition* attributes. The *Cognition* attribute can also be related to intellectual disability or mental illness.

Table 4. Parallelism between attributes and body's organ systems.

Body's Organ System	**Attributes**
Musculoskeletal system Nervous system Respiratory system Cardiovascular system Hematopoietic system Digestive system Genitourinary system Skin and annex Neoplasms	Manipulation, Strength, Reach
Visual system	Vision, Color Perception
Ear, throat and related	Hearing, Manipulation, Strength, Reach
Speech/talk/voice	Vocal capability, Cognition
Intellectual disability	Cognition
Mental illness	Cognition

4.2. Basic Architecture

As we pointed out earlier, the model we propose allows service providers to decide the interface they should serve to the users depending on their functional capabilities. To achieve this, we have considered IAACaaS (IoT Application-Scoped Access Control as a Service) model [46,47] as a starting point. IAACaaS enables the complete delegation of authentication and authorization to enable an as a service access control mechanism for Internet of Things applications.

The authorization mechanism provided by IAACaaS is based on OAuth 2.0 protocol and ABAC to enable the decision making based on the attributes of users and IoT devices. Attributes are defined in a centralized Identity Manager (IdM) as part of the identity profile of the actors (users or devices) registered in such IdM. On the other hand, services are also registered in the IdM in order to enable the delegated authentication and authorization. Thus, when actors consume the services, they log in using their IdM accounts and services can then delegate the access decisions to a Policy Decision Point (PDP) deployed together with the IdM. The PDP takes the decisions based on the actors' attributes and the policies previously defined by the service providers.

In a first approach we have simplified the IAACaaS architecture for providing the users' profile to the services allowing them to decide which interface they should provide depending on the defined attributes. We also propose the extension of the IdM features to permit the definition of the identity attributes in the users' profile. Figure 1 shows the connections between the user, the service and the IdM.

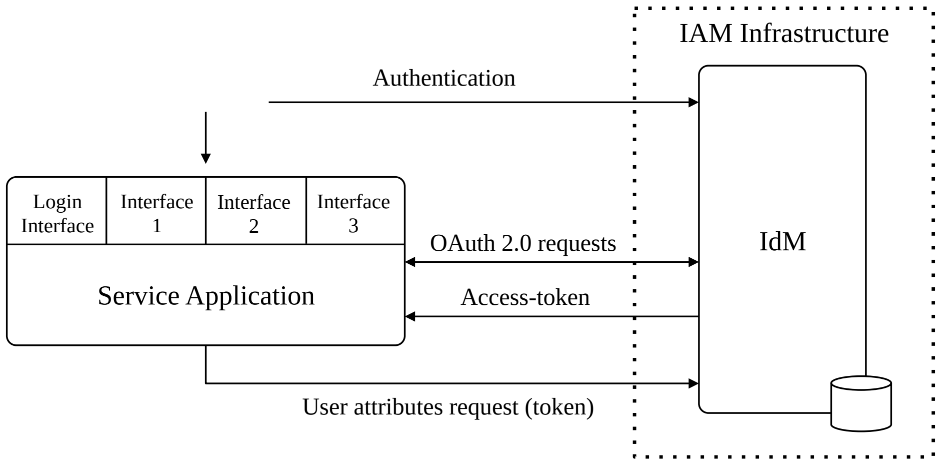


Figure 1. Basic architecture.

The service has to be registered in the IdM and to implement OAuth 2.0 protocol to support delegated authentication. As the model is thought to be used in web applications, services use Authorization Code grant type (OAuth 2.0 grant types: https://oauth.net/2/grant-types), designed for web clients. Developers have to implement adapted interfaces and configure the application business logic to serve them depending on the users' attributes received from the IdM. The IdM acting as OAuth 2.0 server has to support the definition of the functional attributes proposed above and to include their values in the user profile returned to the service when validating an OAuth 2.0 access token. Figure 2 shows the interaction between these entities in a basic flow:

1. The model assumes users have been previously registered in the IdM and that their identity attributes have been configured there. In case the IdM interfaces are not adapted to the functional capabilities of an specific user, the assistance of a third actor could be necessary the first time. Once users have been registered and their functional attributes have been defined, their profile should be modified only if any of the functional capabilities changes.
2. When users access the service, they have to login using their previously registered account in the IdM. For doing this, the service redirects the authentication requests to the IdM login page using OAuth 2.0 protocol. Both the service and the IdM login pages must be as simple as possible and must provide accessible interfaces to facilitate the authentication process to every user.
3. Once users have authenticated in the IdM and following the Authorization Code grant type flow, it sends the OAuth 2.0 authorization code to the service back-end.
4. Using the received code, the service back-end sends a token creation request to the IdM, that creates a token associated to each specific user. This token represents them in terms of authorization and can be used to retrieve their public profile.
5. The access token is returned to the service and
6. The service has to include it as part of every request sent from clients henceforth. The token is typically included in the *X-Auth-Token* HTTP header of the requests.
7. When receiving requests from users, the service extracts the access token from the HTTP header and validates it through the IdM. Caching mechanisms can be used to avoid excesive token validations.
8. When receiving a token validation request, the IdM checks the validity if the token internally and maps it to a registered user. Then, the IdM returns the profile of the user to the service including the identity attributes defined for such user.
9. Using the received functional attributes, the service decides which interface to serve to the user.

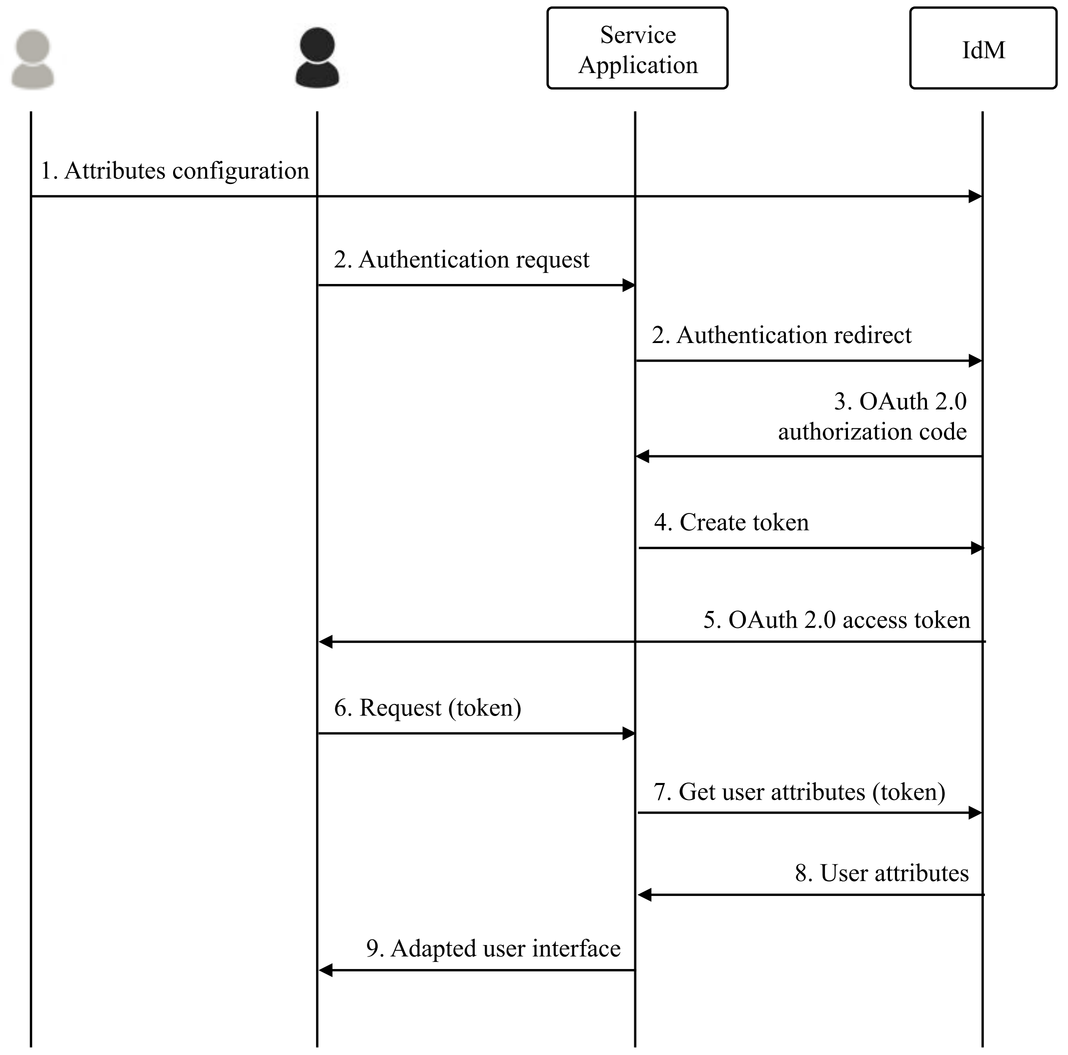

Service Application
IdM
1. Attributes configuration
2. Authentication request
2. Authentication redirect
3. OAuth 2.0 authorization code
4. Create token
5. OAuth 2.0 access token
6. Request (token)
7. Get user attributes (token)
8. User attributes
9. Adapted user interface

Figure 2. Basic flow.

It is important to remark that our proposal is focused on facilitating a mechanism to provide services adapted to the functional capabilities of users based on their attributes. We assume the Identity Managers used for defining those attributes and authenticating users are already designed for been accessible. In case they are not, the assistance of a third actor is required. Therefore, how to design inclusive Identity Managers is out of the scope of this research and, as stated in Section 2.3, it is an open issue in the literature.

4.3. Advanced Architecture

In complex scenarios where services need to include access control checks for determining which users can access the exposed resources, the whole IAACaaS model can be integrated in the proposed architecture. For managing access control policies, IAACaaS defines a set of rules that are composed by a target, an effect and a condition. These rules serve to create permissions and the permissions are grouped in rolesT. Taking advantage of the application-scoped feature of IAACaaS, roles and permissions can be defined in the scope of a service or application.

Figure 3 shows the details of the integration of IAACaaS with our proposed architecture. The resources exposed by the service are typically provided by its back-end component (Service Backend). Apart of the IdM, the following components are needed as part of the Identity and Access Control Management (IAM) Infrastructure:

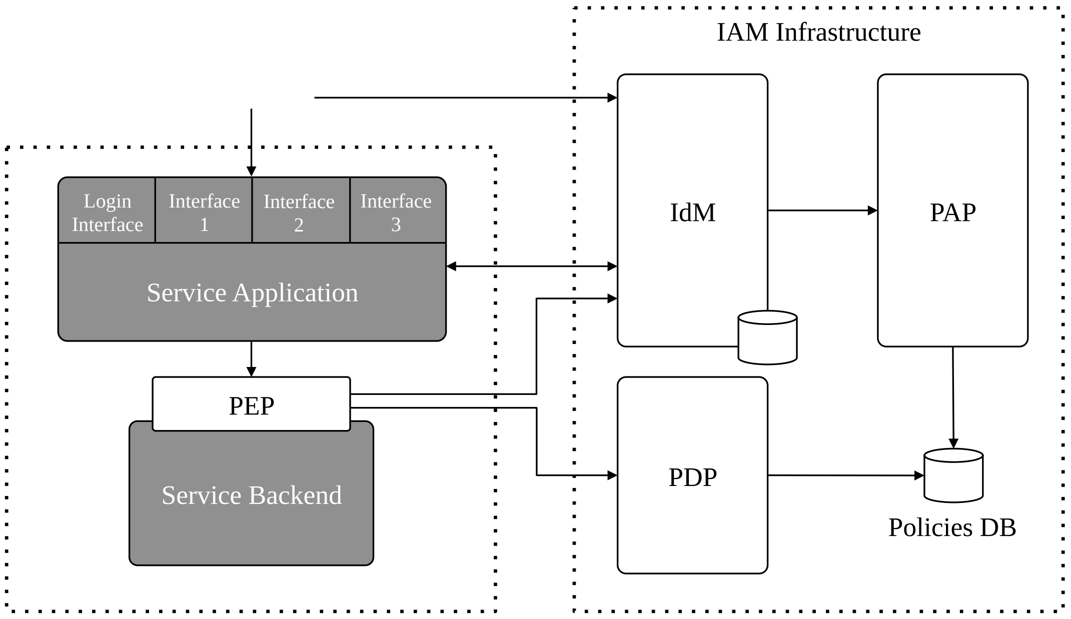


Figure 3. Advanced architecture with access control.

- Policy Administration Point (PAP): Policies are set of rules and rules consist on a target (typically a resource), an effect (for instance allow/deny) and a condition [48]. In our proposal, rules are used to create *permissions*, where the target is an action (specified by an HTTP verb) and a resource of the service (specified by an HTTP path). More complex policies may be also defined, using the policy-description language. On the other hand, roles are sets of permissions, to allow developers to assign more than one permission at once. Service administrators define permissions, roles and their relationships in the PAP.
- The Policy Enforcement Point (PEP): The PEP is the component that uses the access token (previously obtained by the user from the IdM) to retrieve the roles associated to the user from the IdM. Uisng these roles, the PEP sends the authorization check to the PDP.
- The Policy Decision Point (PDP): The PDP receives an authorization check request from the PDP, retrieves the policies associated to those roles from the Policies DB and decides if a user that owns those roles has the required grants to perform the requested action in the requested resource.

Summarizing, roles, permissions and their relationships are defined in the Policy Administration Point (PAP) and assigned to users in the IdM. As explained before, roles are assigned to users in the scope of the service, so a given user may have different roles depending on the application. During an authorization check, the Policy Enforcement Point (PEP) is the entity that uses the access token included in the requests sent from the application to the back-end to retrieve from the IdM the roles assigned to the User.

Using this architecture, different levels of security can be achieved:

1. Authentication: This level of security allows the access to the protected resource to every user with an active account in the IdM. In this case, the PEP just checks if the token included in the request is valid sending a validation request to the IdM.
2. Basic authorization: This level of security allows the access to the protected resource depending on the roles users have. The check is performed based on the action (defined by an HTTP verb) the user wants to perform in an specific resource (defined by an HTTP path). During the check to validate the token, the PEP obtains the roles associated to the user (being a role a set of permissions) and sends a new request to the PDP, that decides if those roles permit the execution of the requested action in the resource. For taking the decision the PDP fetches the policies from the Policies DB.

3. Advanced authorization: In this case, not only the action and the resource (HTTP verb and path respectively) are checked bu also other advanced parameters such as the body or the headers of the requests. To perform the check, a custom XACML policy request is sent to the PDP.

Figure 4 shows the detail about how authorization checks are performed using this architecture. The first steps are the same than in the basic flow explained above. After users' attributes have been configured in the IdM, they try to authenticate in the application. Then, the application delegates the authentication request to the IdM where users have to introduce their credentials. After the interchange of OAuth 2.0 requests and responses (process simplified in this Figure), the application receives an access token that represent each user in terms of authorization. This token has to be included in every request sent to the service backend to obtain resources.

In the new architecture, a PEP component has to be deployed on top of the service backend. Thus, requests are intercepted by this component (Step 4) that firstly validates the token with the IdM. As result of this validation, the IdM returns the user profile and the roles associated to the user in the scope of the application (Step 6). Using these roles, the PEP can check with the PDP if the user has the needed permissions to perform the requested action into the requested resources (Step 7). The PDP executes the check, in Step 8, in the Policies DB and generates a decision that is returned to the PDP (Step 9). If the validation success, the PEP forwards the original request to the service backend (Step 10). After receiving the response in Step 11, the application includes the retrieved data in the adapted web interface (Step 12), that is selected depending on the user profile in the same way than in the basic scenario.

Thanks to this integration, the possibility of serving adapted web interfaces to users is totally integrated in an advanced identity architecture that offers application scoped access control decision features.

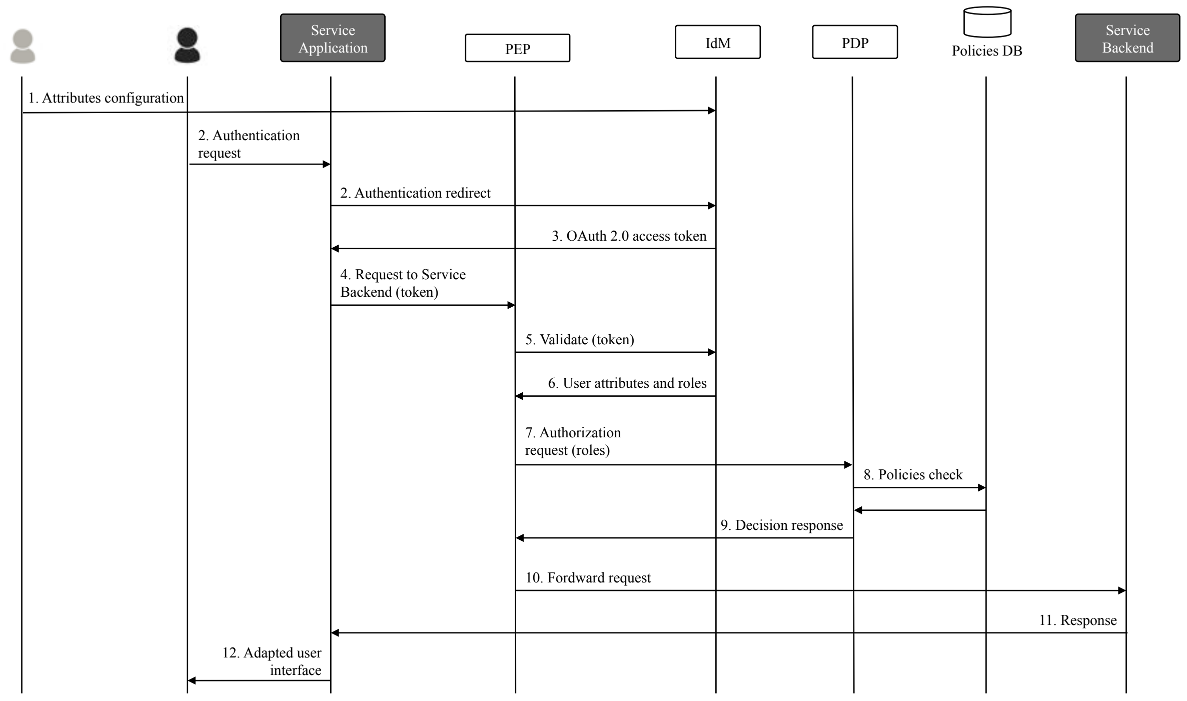

Service Application
PEP
IdM
PDP
Policies DB
Service Backend
1. Attributes configuration
2. Authentication request
2. Authentication redirect
3. OAuth 2.0 access token
4. Request to Service Backend (token)
5. Validate (token)
6. User attributes and roles
7. Authorization request (roles)
8. Policies check
9. Decision response
10. Fordward request
11. Response
12. Adapted user interface

Figure 4. Advanced flow with access control.

5. Validation and Results

For validating our proposal, we have implemented and deployed the proposed model extending FIWARE Generic Enablers. FIWARE (FIWARE: The open source platform for our smart digital future, https://www.fiware.org/) is an independent open community whose members are committed to materialise the mission of building an open sustainable ecosystem around public, royalty-free and implementation-driven software platform standards that will ease the development of new Smart Applications in multiple sectors.

To achieve this mission, the FIWARE platform provides a set of APIs (Application Programming Interfaces) that ease the development of Smart Applications oriented to the Future Internet. The specifications of these APIs are public and royalty-free. Besides, an open source reference implementation of each of the FIWARE components (named Generic Enablers, GEs) is publicly available.

FIWARE Security Framework [47,49] provides several software components for providing Identity and Access control to the FIWARE-based services and applications. Specifically, it offers an OAuth 2.0 server for registering third-party applications delegating the authentication to such server. Furthermore, FIWARE Security GEs offer Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC) mechanisms based on XACML 3.0 [48].

The IAACaaS model is based in FIWARE Security GEs and in this work we have extended them to support our proposed model. Moreover, we have designed and deployed a use case that exploits the proposal by providing an accessible web application adapted to the specific functional capabilities of users.

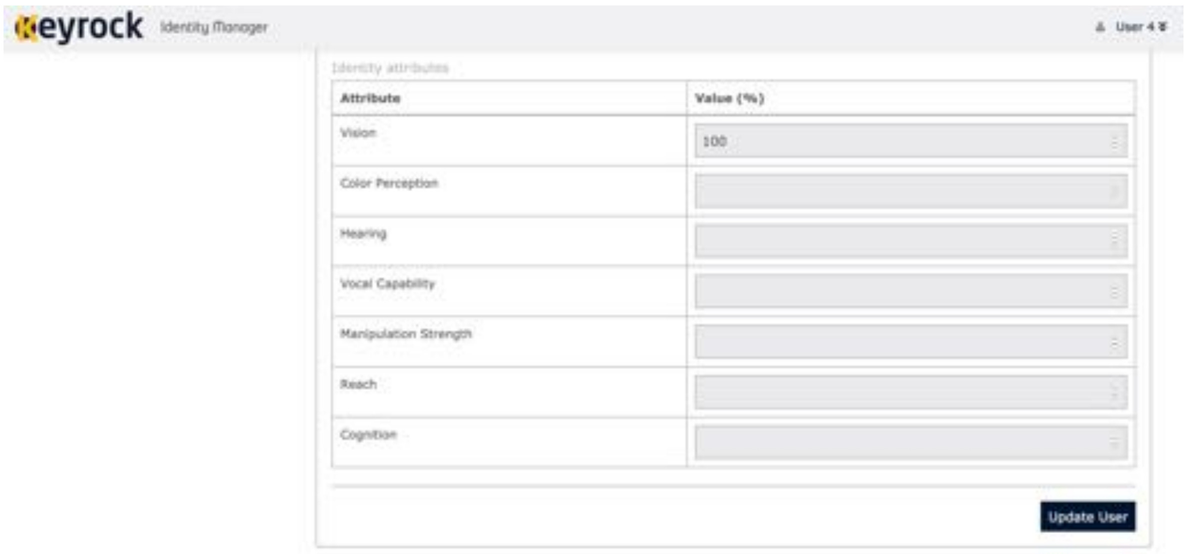
5.1. Implementation

We have implemented the proposed model as an extension of FIWARE Keyrock Generic Enabler (Keyrock Generic Enabler: https://fiware-idm.readthedocs.io). Keyrock is the Identity Management GEri (Generic Enabler reference implementation) of FIWARE and it brings support to secure and private OAuth 2.0—based authentication of users and IoT devices. Keyrock also offers APIs and web interfaces to manage users' profiles, roles and permissions to secure access to third-party services.

Keyrock has been developed by the authors of this paper using Node.js (Node.js: https://nodejs.org) and Express (Express: https://expressjs.com) and makes use of an SQL database for persistence. For implementing the extension to support our proposal, we have implemented new views to configure the functional attributes of users. Table 5 shows the relevant existing attributes in Keyrock's users schema. We include the functional attributes as part of the *extra* field of the users' table. Figure 5 shows the web interface where functional attributes can be defined by users. Then, when an application requests the user information using an OAuth 2.0 access token, the defined functional attributes are included in the returned user profile. Figure 6 shows and example of the JSON object included in the response that Keyrock generates when receiving a token validation request. It returns the list of roles assigned to the user in the scope of the service. These roles can be used by the service to enforce the acccess to the exposed resources by the service.

Table 5. Relevant attributes in Keyrock's user profile.

Name	Description
id	UUID that uniquely identifies a user in Keyrock
displayName	Friendly Name that identifies a user in Keyrock
description	Text the users can use to include information about themselves
image	Path to the image of the user
email	Mail address of the user. It is unique in the database
password	Private key for authentication
enabled	Boolean that allows the user or not to perform requests to Keyrock
admin	Boolean that indicates if the user is an administrator of Keyrock
extra	Allows to store extra attributes of the user

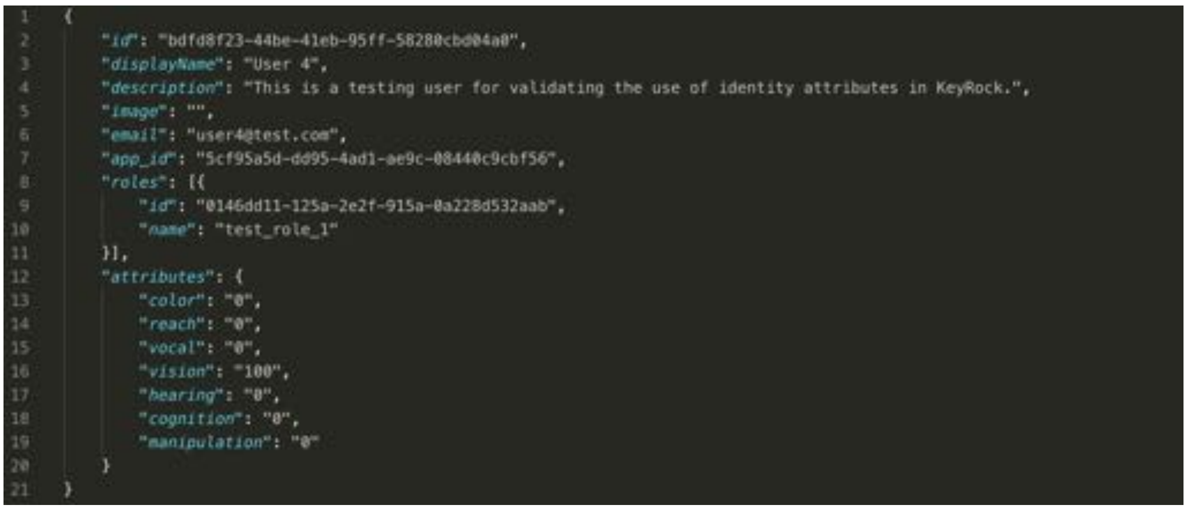


Keyrock Identity Manager User 4

Identity attributes

Attribute	Value (%)
Vision	100
Color Perception	
Hearing	
Vocal Capability	
Manipulation Strength	
Reach	
Cognition	

Figure 5. Keyrock's interface for defining functional attributes.



```
{
    "id": "bdfd8f23-44be-41eb-95ff-58280cbd04a0",
    "displayName": "User 4",
    "description": "This is a testing user for validating the use of identity attributes in KeyRock.",
    "image": "",
    "email": "user4@test.com",
    "app_id": "5cf95a5d-dd95-4ad1-ae9c-08440c9cbf56",
    "roles": [{
        "id": "0146dd11-125a-2e2f-915a-0a228d532aab",
        "name": "test_role_1"
    }],
    "attributes": {
        "color": "0",
        "reach": "0",
        "vocal": "0",
        "vision": "100",
        "hearing": "0",
        "cognition": "0",
        "manipulation": "0"
    }
}
```

Figure 6. User profile returned by Keyrock.

5.2. Use Case

We have developed a simple blog page as a sample OAuth 2.0 client (Use case blog service: https://github.com/Lourdesmarco/a11ymodel-use-case). The blog application serves different web interfaces depending on the users' functional attributes received after authentication. Table 6 summarizes the interfaces we have designed and the conditions under which each of them are provided.

Table 6. Interfaces summary.

Interface	Description	Condition
Interface 1	Article Article high contrast	No modifications Vision>=1 && Vision<100
Interface 2	Article adapted to cognition level	Cognition>=1
Interface 3	Voice User Interface Suggestion High contrast VUI Suggestion	Vision==100, manipulation<=100 Vision>=50 + Manipulation>=50

For testing the service, we have deployed a Keyrock instance with support to our proposed model. We have registered the designed blog application in the Keyrock instance using an OAuth 2.0 library to delegate users' authentication to Keyrock's OAuth 2.0 server. Then, we have created five users which will define their functional capabilities using Keyrock's interface according to the functional attributes enumerated in Table 7.

Table 7. Users & Functional attributes.

User	Functional Attributes
User 1	No modifications
User 2	Cognition = 60
User 3	Vision = 70
User 4	Vision = 100
User 5	Vision = 40, Manipulation = 70

Figure 7 shows an example of how two of the registered users access the service. We can see how they have to sign in the application and how the service is responding according to the personal attributes each user has. In the example, both users are consuming the same service but not through the same interface, as explained below.

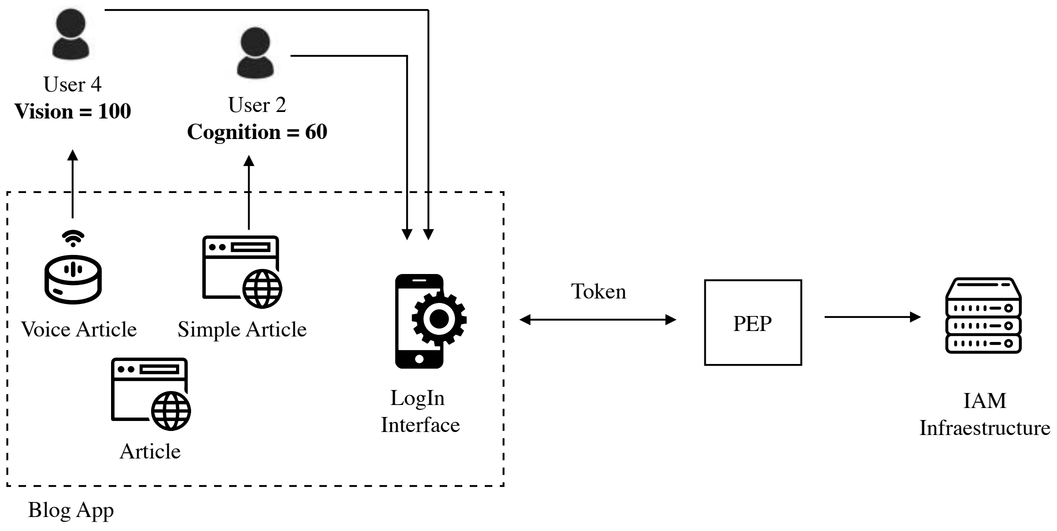
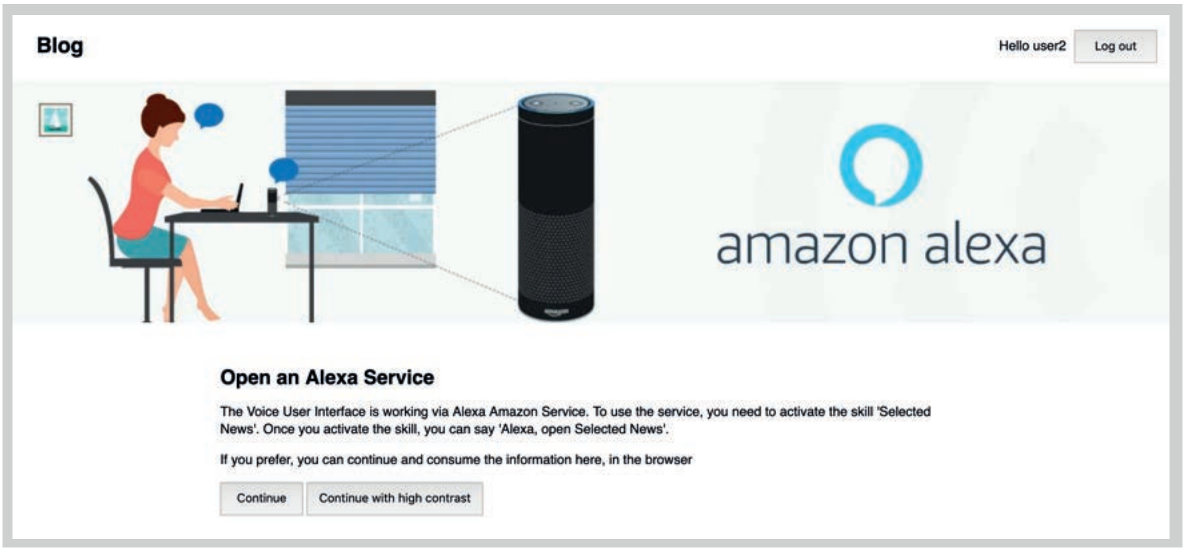


Figure 7. Use case diagram.

User 4 has 100% vision disability, meaning he is blind. To get a custom experience with the service, the user (helped by a guide-interpreter if needed) completes his functional attributes (*vision* = 100). When accessing the blog service, he gets the log in interface. Easily, he can log in using Keyrock interface that is adapted to be usable. After the authentication process, Keyrock returns an OAuth 2.0 access token that represents the user. Using this token, the service requests the user profile that is returned including the defined functional attributes. As a result, the blog service can evaluate the user profile and responds to the user with a custom interface based on his personal attributes. In the implementation of the service we include all the designed interfaces as views (in this case using Embedded JavaScript templates). Then, we include in the application business logic a correspondence between the ranges of the different attributes and the interfaces according to the details of Table 6. Thus, when the application has to render the requested view, checks this correspondence to decide which one has to be selected. In the use case and for *User 4*, *Interface 3* (Figure 8) is served to suggest the user to use Voice User Interface. The latter has been developed using Amazon's cloud-based voice service (Alexa skills: https://developer.amazon.com/alexa). To use the skill, the user needs to activate it and just ask Alexa to open the service.

The second user (*User* 2) has a completely different scenario. She has a 60% cognition disability, meaning she has problems reading and understanding very long or complex texts. In this case, when the user logs in, a custom interface based on her cognition level (*Interface 2* in Figure 9) is served.

Finally, according the attributes defined in Table 7, the other users (1, 3, 5) also have specific needs. *User 1* will access the *Interface 1* of the blog service because no functional attributes have been defined. *User 3* will access the *Interface 1* as well, but including high contrast due to his low vision (vision = 70) disease. Finally, *User 5* will access high contrasted *Interface 3*, in this case because of low vision (vision = 40) and manipulation (manipulation = 70) diseases are defined.



Open an Alexa Service

The Voice User Interface is working via Alexa Amazon Service. To use the service, you need to activate the skill 'Selected News'. Once you activate the skill, you can say 'Alexa, open Selected News'.

If you prefer, you can continue and consume the information here, in the browser

Figure 8. Interface 3. Voice User Interface suggestion.

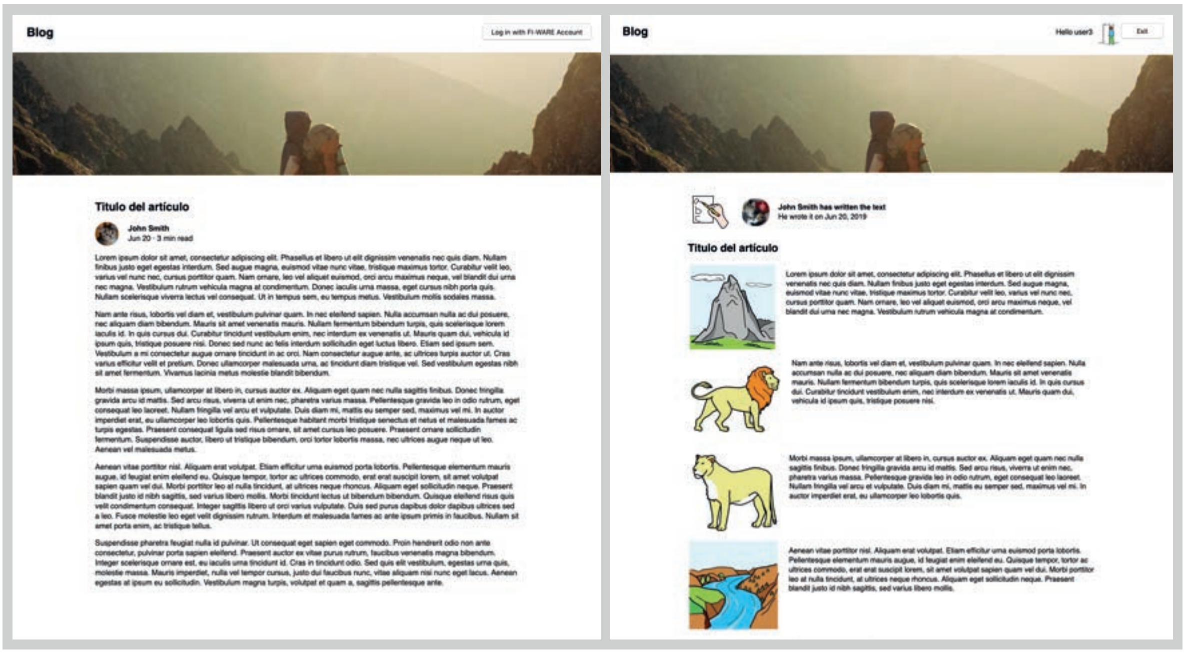


Figure 9. Log In Interface (**left**) & Interface 2 (**right**).

5.3. Discussion

The proposed model is designed by taking two main factors into account: the usual practices in web development with regard to web accessibility and the limitations of existing adaptive UI model-based systems. Even though the approaches analyzed in the field of MBUID are automated and capable of generating UI at runtime, most of them require redeployments to add new rules, and just [40] clearly presents a user model and a role-based solution. In addition, only two of them [39,42] are proposed as an inclusive solution. On the other hand, most common accessibility approaches for designing web content are mainly based on evaluation methods, which results in social exclusion increase in most cases.

Our approach goes beyond the [40] role-based solution, taking into account functional attributes for designing inclusive interfaces and delegating the user model management to a centralized identity manager, thus allowing users to define their profiles once and using them in several services. Moreover, our proposal stands in contrast to the prior work in web accessibility development, on the basis of a proactive solution that enables a completely flexible scenario.

Analyzing the use case we have developed we can see how the detected gaps in the related work are covered. Web interfaces have been individually designed taking into account the conditions exposed in Table 6. In this example the three designed interfaces take into account vision, cognition and manipulation conditions of users. However, in real services other interfaces can be progressively included to cover other type of conditions. In contrast with the automated generated UIs solutions found in the existing related works, the customized design ensures the same contents are differently provided to users taking into account the way they are going to consume them.

On the other hand, the users profiles where the functional attributes are defined (as stated in Table 7), are created in a centralized Identity Manager. This enables the use of the profiles to every service that implements OAuth 2.0 protocol to delegate authentication to an identity server. Therefore, users have to define their profile just one time and can take advantage of it in every service they use. The use of a standard like OAuth 2.0 permits the integration with well-known identity services such as the ones provided by the social networks Facebook or Twitter.

We have implemented this use case to show how interfaces are provided to users depending on their functional attributes, previously defined in their IdM profile. However, in more complex use cases where users need to get access to protected resources served by a backend (typically RESTful servers) the integration with the advanced IAM architecture presented above can be achieved. In such cases, the services take into account the functional attributes of users to decide which interfaces provide to them and other authorization parameters (permissions, roles and attributes) to decide if users have the required grants to access the protected resources.

6. Conclusions

Information and Communication Technologies (ICT) need to be accessible for every single person in the globe. However, the approaches we can find in the literature with regard to web accessibility development just adapt the standard UIs to the specific necessities of users following a set of rules and recommendations. In addition, other related fields such as MBUID do not provide inclusive and flexible enough solutions to deal with the users diversity challenge. In this paper, we demonstrate the necessity of integrating the current standards throughout the development processes, where the selected architecture models play an essential role. Therefore, the web interfaces must be adapted to the functional capacities of users from the very beginning of the design phase.

For facilitating the task to developers, we have presented an OAuth 2.0—based model that enables inclusive contexts by using the identity attributes of a person held. Web interfaces can be designed taking into account the attributes of the users and their capacities to use visual, voice-based or tactile interfaces. For designing them we also propose a list of functional attributes and their correspondence with different types of interfaces. Thanks to the model we propose, users define their attributes in a centralized way so different services can take advantage of them by delegating the authentication using the well-known standard OAuth 2.0. Moreover, the model is integrated in an advanced Identity and Access Control model that provides RBAC and ABAC mechanisms based on XACML.

The use case explained above illustrates how the model we propose accomplishes the requirements exposed in Section 3 with regard accesibility in ICT services. The blog service, able to serve very different interfaces, is a good example of how the model covers all the functional performance statements defined previously. On the other hand, thanks to Keyrock Identity Manager, any service capable of creating an OAuth 2.0 token can delegate authentication obtaining the public profile of users defined in the IdM. This allows the decisions on how to serve the web interfaces adapted to the functional capabilities of users. Moreover, the definition of functional attributes results in a highly flexible tool to address several inclusive scenarios.

Finally, it is important to state that the proposed solution enables a new identity paradigm that satisfies the diversity requirements in terms of design and development. This shift in perspective enables a proactive approach necessary to tackle web accessibility challenges.

As a future research line, some improvements could be considered, such as measuring the performance of the proposed solution by applying the model in different and wider scenarios. On the other hand, as we have seen, it is an imperative need to design inclusive Identity Managers to allow users to be completely independent for managing their digital identity. Lastly, some other approaches are being considered, as the inclusion of the proposed solution as part of the Connecting Europe Facility European program and the eIDAS (electronic IDentification, Authentication and trust Services) regulation [50].

Author Contributions: Conceptualization, Á.A., L.M. and J.Q.; methodology, Á.A.; software, L.M. and Á.A.; validation, L.M.; formal analysis, L.M.; investigation, L.M.; resources, J.Q.; writing—original draft preparation, L.M. and Á.A.; writing—review and editing, L.M., Á.A. and J.Q.; supervision, Á.A.; project administration, J.Q.; funding acquisition, J.Q.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kaplan, D. The definition of disability: Perspective of the disability community. *J. Health Care Law Policy* **1999**, *3*, 352.
2. Shakespeare, T. The social model of disability. *Disabil. Stud. Read.* **2006**, *2*, 197–204.
3. Leonardi, M.; Bickenbach, J.; Ustun, T.B.; Kostanjsek, N.; Chatterji, S. The definition of disability: What is in a name? *Lancet* **2006**, *368*, 1219–1221. [CrossRef]
4. Romañach, J.; Lobato, M. Functional Diversity, a New Term in the Struggle for Dignity in the Diversity of the Human Being. In *Independent Living Forum*; European Network on Independent Living: Valencia, Spain, 2005.
5. WHO. *World Report on Disability*; WHO: Geneva, Switzerland, 2011.
6. WebAIM. The WebAIM Million, an Accessibility Analysis of the Top 1,000,000 Home Pages. 2019. Available online: https://webaim.org/projects/million/ (accessed on 10 September 2019).
7. Marco, L.; López-Pernas, S.; Alonso, A. Accessibility review for web-based learning tools and materials. In Proceedings of the ICERI2018—11th Annual International Conference of Education, Research and Innovation, Seville, Spain, 12–14 November 2018; pp. 2393–2402. [CrossRef]
8. Hardt, D. *The OAuth 2.0 Authorization Framework*; Technical report; Microsoft: Washington, DC, USA, 2012.
9. Yuan, E.; Tong, J. Attributed based access control (ABAC) for web services. In Proceedings of the IEEE International Conference on Web Services (ICWS'05), Orlando, FL, USA, 11–15 July 2005.
10. ISO. *Ergonomics—General Approach, Principles and Concepts*; Standard; Spanish Organization for Standardization: Geneva, Switzerland, 2012.
11. Olalere, A.; Lazar, J. Accessibility of US federal government home pages: Section 508 compliance and site accessibility statements. *Gov. Inf. Q.* **2011**, *28*, 303–309. [CrossRef]
12. Yesilada, Y.; Brajnik, G.; Vigo, M.; Harper, S. Understanding web accessibility and its drivers. In Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, Lyon, France, 16–17 April 2012; p. 19.

13. Petrie, H.; Kheir, O. The relationship between accessibility and usability of websites. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, San Jose, CA, USA, 28 April–3 May 2007; pp. 397–406.
14. Nielsen, J. Beyond Accessibility: Treating Users with Disabilities as People. The Alertbox: Current Issues in Web Usability. (Work Prepared by the Nielsen Norman Group). 2001. Available online: https://www.nngroup.com/articles/beyond-accessibility-treating-users-with-disabilities-as-people (accessed on 10 September 2019).
15. Chisholm, W.; Vanderheiden, G.; Jacobs, I. Web content accessibility guidelines 1.0. *Interactions* **2001**, *8*, 35–54. [CrossRef]
16. Caldwell, B.; Cooper, M.; Reid, L.G.; Vanderheiden, G. Web content accessibility guidelines (WCAG) 2.0. *WWW Consort. (W3C)* **2008**, *11*, 1–34.
17. Kirkpatrick, A.; O Connor, J.; Campbell, A.; Cooper, M. Web content accessibility guidelines (WCAG) 2.1. *WWW Consort. (W3C)* **2018**.
18. ETSI. *Accessibility Requirements Suitable for Public Procurement of ICT Products and Services in Europe (EN 301 549)*; ETSI: Sophia Antipolis, France, 2015.
19. Harper, S.; Yesilada, Y. *Web Accessibility: A Foundation for Research*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008.
20. Moreta, C.D.O.; Baena, L.R. Pautas, métodos y herramientas de evaluación de accesibilidad web [Guidelines, methods and tools for web accessibility evaluation]. *Ventana Inform.* **2013**, *1*, 99–115.
21. López Zambrano, J.H.; Pico, M.; Joffre, R.; Alava Cagua, N.V. Metodología para valorar y clasificar herramientas de evaluación de accesibilidad web. *e-Ciencias de la Información* **2018**, *8*, 1–18. [CrossRef]
22. Abou-Zahra, S. Web accessibility evaluation. In *Web Accessibility*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 79–106.
23. Vigo, M.; Brajnik, G. Automatic web accessibility metrics: Where we are and where we can go. *Interact. Comput.* **2011**, *23*, 137–155. [CrossRef]
24. Brajnik, G.; Yesilada, Y.; Harper, S. Testability and validity of WCAG 2.0: The expertise effect. In Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility, Orlando, FL, USA, 25–27 October 2010; pp. 43–50.
25. Jaeger, P.T. Assessing Section 508 compliance on federal e-government Web sites: A multi-method, user-centered evaluation of accessibility for persons with disabilities. *Gov. Inf. Q.* **2006**, *23*, 169–190. [CrossRef]
26. Brajnik, G. Beyond conformance: The role of accessibility evaluation methods. In *International Conference on Web Information Systems Engineering*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 63–80.
27. Vigo, M.; Arrue, M.; Brajnik, G.; Lomuscio, R.; Abascal, J. Quantitative metrics for measuring web accessibility. In Proceedings of the 2007 International Cross-Disciplinary Conference on Web Accessibility (W4A), Banff, AB, Canada, 7–8 May 2007; pp. 99–107.
28. Vigo, M.; Brown, J.; Conway, V. Benchmarking web accessibility evaluation tools: measuring the harm of sole reliance on automated tests. In Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility, Rio de Janeiro, Brazil, 13–15 May 2013; p. 1.
29. Bigham, J.P.; Brudvik, J.T.; Zhang, B. Accessibility by demonstration: Enabling end users to guide developers to web accessibility solutions. In Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility, Orlando, FL, USA, 25–27 October 2010; pp. 35–42.
30. Federici, S.; Micangeli, A.; Ruspantini, I.; Borgianni, S.; Corradi, F.; Pasqualotto, E.; Olivetti Belardinelli, M. Checking an integrated model of web accessibility and usability evaluation for disabled people. *Disabil. Rehabil.* **2005**, *27*, 781–790. [CrossRef] [PubMed]
31. Brands, S. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*; MIT Press: Cambridge, MA, USA, 2000.
32. Shahkooh, K.A.; Saghafi, F.; Abdollahi, A. A proposed model for e-Government maturity. In Proceedings of the 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, 7–11 April 2008; pp. 1–5.
33. Myers, M.; Ankney, R.; Malpani, A.; Galperin, S.; Adams, C. *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol-OCSP*; Technical report; RFC 2560; Internet Engineering Task Force (IETF): Washington, DC, USA, 1999.
34. Fritsch, L.; Fuglerud, K.S.; Solheim, I. Towards inclusive identity management. *Identity Inf. Soc.* **2010**, *3*, 515–538. [CrossRef]

35. García, B.; Gómez, A.; Conde, R.; Hernández, Y.; Valero, M.Á. Breaking the web barriers of the e-Administration using an accessible digital certificate based on a cryptographic token. *Adv. Softw. Eng.* **2015**, *2015*, 568087. [CrossRef]
36. Gabillon, Y.; Biri, N.; Otjacques, B. Designing an adaptive user interface according to software product line engineering. *Proc. ACHI* **2015**, *15*, 86–91.
37. Capilla, R.; Bosch, J.; Trinidad, P.; Ruiz-Cortés, A.; Hinchey, M. An overview of Dynamic Software Product Line architectures and techniques: Observations from research and industry. *J. Syst. Softw.* **2014**, *91*, 3–23. [CrossRef]
38. Sboui, T.; Ayed, M.B.; Alimi, A.M. A UI-DSPL Approach for the Development of Context-Adaptable User Interfaces. *IEEE Access* **2017**, *6*, 7066–7081. [CrossRef]
39. Gajos, K.Z.; Weld, D.S.; Wobbrock, J.O. Automatically generating personalized user interfaces with Supple. *Artif. Intell.* **2010**, *174*, 910–950. [CrossRef]
40. Akiki, P.A.; Bandara, A.K.; Yu, Y. Engineering adaptive model-driven user interfaces. *IEEE Trans. Softw. Eng.* **2016**, *42*, 1118–1147. [CrossRef]
41. Balme, L.; Demeure, A.; Barralon, N.; Coutaz, J.; Calvary, G. Cameleon-rt: A software architecture reference model for distributed, migratable, and plastic user interfaces. In *European Symposium on Ambient Intelligence*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 291–302.
42. Hussain, J.; Hassan, A.U.; Bilal, H.S.M.; Ali, R.; Afzal, M.; Hussain, S.; Bang, J.; Banos, O.; Lee, S. Model-based adaptive user interface based on context and user experience evaluation. *J. Multimodal User Interfaces* **2018**, *12*, 1–16. [CrossRef]
43. Akiki, P.A.; Bandara, A.K.; Yu, Y. Adaptive model-driven user interface development systems. *ACM Comput. Surv.* **2014**, *47*, 9. [CrossRef]
44. Martínez, L.; Pluke, M. Mandate M 376: New software accessibility requirements. *Procedia Comput. Sci.* **2014**, *27*, 271–280. [CrossRef]
45. Agencia Estatal. *REAL DECRETO 1971/1999, de 23 de Diciembre, de Procedimiento para el Reconocimiento, Declaración y Calificación del Grado de Minusvalía*; Ministerio de Trabajo y Asuntos Sociales: Madrid, Spain, 1999.
46. Alonso, A.; Fernández, F.; Marco, L.; Salvachúa, J. Iaacaas: Iot application-scoped access control as a service. *Future Internet* **2017**, *9*, 64. [CrossRef]
47. Fernández, F.; Alonso, A.; Marco, L.; Salvachúa, J. A model to enable application-scoped access control as a service for IoT using OAuth 2.0. In Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), Paris, France, 7–9 March 2017; pp. 322–324.
48. OASIS. *eXtensible Access Control Markup Language (XACML) Version 3.0*; OASIS: Burlington, MA, USA, 2013.
49. Alonso, A.; Pozo, A.; Cantera, J.M.; De la Vega, F.; Hierro, J.J. Industrial Data Space Architecture Implementation Using FIWARE. *Sensors* **2018**, *18*, 2226. [CrossRef]
50. Alonso, A.; Pozo, A.; Choque, J.; Bueno, G.; Salvachúa, J.; Diez, L.; Marín, J.; Alonso, P. An Identity Framework for Providing Access to FIWARE OAuth 2.0-Based Services According to the eIDAS European Regulation. *IEEE Access* **2019**, *7*, 88435–88449. [CrossRef]