Accessibility through Preferences: Context-Aware Recommender of Settings[*]

Andrés Iglesias-Pérez[1,**], Claudia Loitsch[2], Nikolaos Kaklanis[3], Konstantinos Votis[3], Andreas Stiegler[4], Konstantinos Kalogirou[5], Guillem Serra-Autonell[6], Dimitrios Tzovaras[3], and Gerhard Weber[2]

[1] R&D Department, Fundosa Technosite
Technosite C/Albasanz 16, 3-B 28037 Madrid. Spain
aiglesias@technosite.es, andresip@gmail.com
[2] Technical University of Dresden, Germany
claudia.loitsch@tu-dresden.de
[3] Information Technologies Institute, Centre for Research and Technology Hellas, Greece
[4] Stuttgart Media University, Germany
[5] Hellenic Institute of Transport, Centre for Research and Technology Hellas, Greece
[6] Barcelona Digital Technology Centre, Spain

Abstract. A proposal for merging context-awareness and user preferences in the same software system is provided. Several modules from the on-going CLOUD4All project (European Commission Seventh Framework Programme) are enhanced with Context Awareness, including the Semantic Matching Framework, the RuleBased Matchmaker (with new rules) and the Statistical Matchmaker (with new features to be used as predictors). Some other components are created exclusively to deal with context features, as the Context Aware Server (to add context from motes) and the Minimatchmaker (to save computation and network resources for well-known situations)

Keywords: e-Inclusion, Personalization,Context-awareness for universal access.

1 Introduction

Daily life in urban environments tends to force the user to interact with a plethora of machines, each with its own User Interface (UI). Most probably even getting into the workplace will involve checking the smartphone in the morning for email and ToDo lists, getting a ticket to the metro or bus in a Ticket Vending Machine (TVM), reading on a tablet while in transit and finally opening the desktop computer on arrival. Currently, the personalisation of all of these systems would delay users, so they are likely to use systems under suboptimal conditions, because users rarely make any changes

[*] Results presented in this paper have been researched within the Cloud4all project. Cloud4all is an R&D project that receives funding from the European Commission under the Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 289016.
[**] Corresponding author.

on the UI, especially when the interaction becomes more complicated, as when noise and light conditions vary.

Several approaches [1][2][3][4] have addressed the task of improving the interaction between a user and a computer by means of adding knowledge about the surrounding environment. However, not all of them employ the user needs and preferences about the HCI as a source of data on its own. This hinders the possibility of generating personalised UIs, which are especially important for people with disabilities, as it involves auto-activating Assistive Technologies (AT). On the other hand, researchers [6][7][8][9] have produced fruitful results when adapting the UI to the user needs, whether in automatic or semi-automatic fashion, Nevertheless, in these cases the adaptation of the UI is focused on user needs leaving the context reduced to the device that the user is employing [10]. A good universal design [5] on the server side plus the correct AT with the best configuration of settings on the client side will allow individuals with disabilities to complete tasks without the intervention of third persons, providing full accessibility even in environments that are not owned by the user.

2 Related Work

Several approaches and projects investigated semantic context-aware reasoning techniques towards accessibility in the past. [18] motivated the potential of semantic web reasoning in terms of user interface adaptation recently. Beyond that, they propose a general reasoning architecture to support adaptivity of web-based services. Thereby abstract user interface design is translated into a concrete user interface by considering user (e.g. disability, web familiarity, language) and context attributes (e.g. input-output devices, screen capabilities) as well as interaction data (e.g. user actions, navigation paths) represented in OWL and OWL_DL ontologies. A reasoning module and rule engine undertake decision-making about selecting appropriate and concrete interaction elements. Accessible TV applications through adaptive user interfaces, for instance, have been developing in the GUIDE[19] project. After initial adaptation according to the user capabilities have been calibrated, run-time adaptation is performed by rules. Capability-based reasoning, delivered by semantic information has been recently proposed by [20] Ubiquitous devices, adaptable applications and service-based content are targeted. Even if the reasoning approach fits into the matchmaking approach described in this paper, details about the process are not given. Kadouche`s [21] work based on semantic matching between an environmental model (environmental effectors) and a user model (human factors) to provide assistive user-environment interaction and services. Context information from sensors and a user profile are processed to identify potential handicap situations for users by Description Logic (DL) inference reasoning. The context query engine delivers environment effectors that lead to a handicap situation which can be resolved by assistive environment services. A further, not-directly linked domain knowledge approach is presented by [22] Based on a user model (user capabilities), a UI model (information, content, navigation, and styles) and existing guidelines (principles on web accessibilities e.g WCAG2), represented as ICF and WCAG ontologies, semantic matching through a GOAL model is created and specific guideline rationales are determined. Accessibility rationales are considered within the user context, e.g. capabilities or task. The approach does not perform personalization directly but is interesting as origin for specific contextual adaptations.

3 Methodology

The presented proposal bridges the gap between Context-Awareness and UI Personalisation by orchestrating a layered matching between a) user needs and preferences and b) device/platform settings. (a) is subdivided into a.1) common terms such as brightness or volume level, a.2) preferred applications e.g.: choose NVDA over JAWS, a.3) application-specific settings e.g.: choose voice "Jorge" when employing Loquendo TTS Engine or place keyboard on top-left corner when employing VirtualKeyboard ; whereas (b) deals with constrains "dpA" (device-platform-application) like b.1) physical constrains of the device, e.g: screen-witdht, maximum volume level or camera availability, b.2) type of architecture (Windows, MacOS, AOSP…) , b.3) available solutions for that architecture. In the Figure 1 the "Preferences Server" deals with (a) and the "Device Manager" and "Solutions Registry" deal with (b). The source of knowledge represented as "Context Manager" is actually provided by two modules that are the Minimatchmaker and the Context Aware Server, explained below.

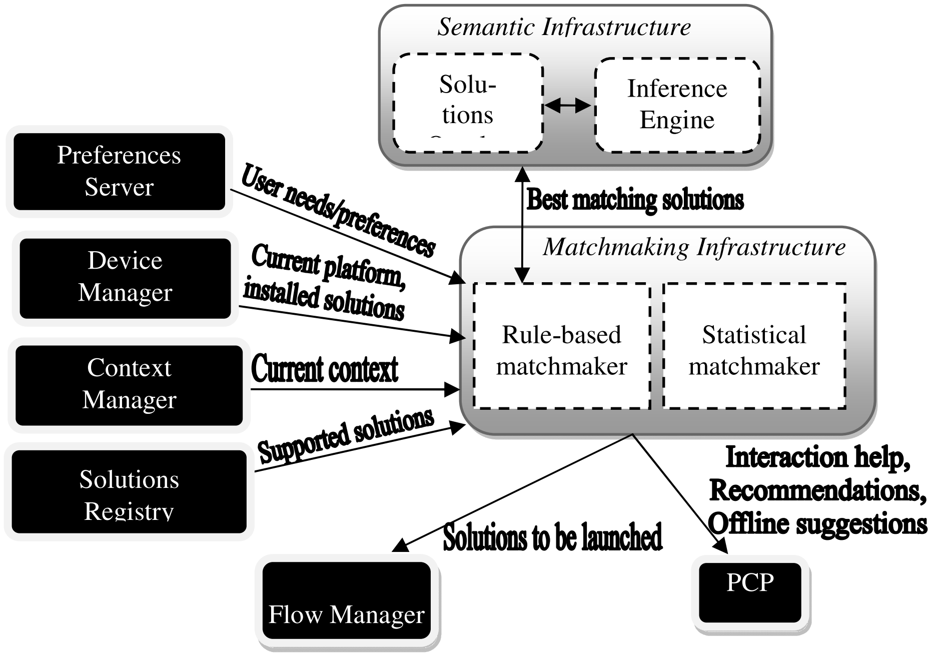
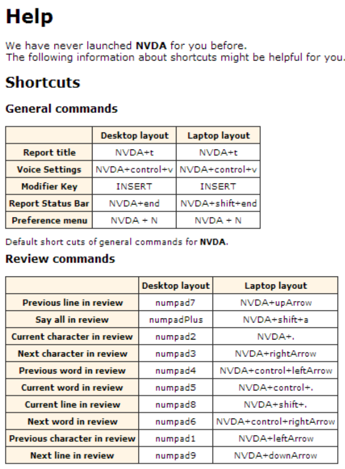


Fig. 1. Rule-based matchmaking strategy

Matchmaking is then the process of adapting a user's preference set to a given context to configure the available solution in a way that matches the preferences of the user [23]. Considering the different kinds of context that one could encounter, this task becomes quite challenging [24].There are two matchmaking strategies working together 14: a statistical approach to infer settings from previous uses of systems and a rule-based approach that exploits the knowledge of the Semantic Framework [13] to gain insights from experts on the domain of AT selection, both of them being able to fetch data from the Preferences Server that users can update thanks to the Preferences Management Tool (PMT) 15. In the device there is also a reasoning module that stores what-if conditions to save computing and network resources. The Flow Manager [16] is responsible for the communication between the modules. The Matchmakers output is divided into

Decision+Interaction Help+Recommendation+OfflineSuggestion.

- *Decision*: some ATs are automatically applied to ensure accessibility.
- *Interaction* Help: decisions to automatically apply AT aims to ensure accessibility but might entail difficulties in utilize, i.e. if the user is not aware of short cuts provided by a screen reader. Important operations are delivered to the user in addition to a decision.
- *Recommendation*: the proposed AT setup is presented in the Personal Control Panel (PCP), a module that acts as a PMT on the user's device. E.g.: preferred speech rate for screen readers.
- *OfflineSuggestion*: When new AT are available and could benefit a specific user, the Matchmakers send an to inform about the new choices.



Help

We have never launched **NVDA** for you before.
The following information about shortcuts might be helpful for you.

Shortcuts

General commands

	Desktop layout	Laptop layout
Report title	NVDA+t	NVDA+t
Voice Settings	NVDA+control+v	NVDA+control+v
Modifier Key	INSERT	INSERT
Report Status Bar	NVDA+end	NVDA+shift+end
Preference menu	NVDA + N	NVDA + N

Default short cuts of general commands for **NVDA**.

Review commands

	Desktop layout	Laptop layout
Previous line in review	numpad7	NVDA+upArrow
Say all in review	numpadPlus	NVDA+shift+a
Current character in review	numpad2	NVDA+.
Next character in review	numpad3	NVDA+rightArrow
Previous word in review	numpad4	NVDA+control+leftArrow
Current word in review	numpad5	NVDA+control+.
Current line in review	numpad8	NVDA+shift+.
Next word in review	numpad6	NVDA+control+rightArrow
Previous character in review	numpad1	NVDA+leftArrow
Next line in review	numpad9	NVDA+downArrow

Fig. 2. Interaction help example

Although the system is designed to let the user apply the settings he/she agrees, the Decision part seems to be unavoidable, as some AT (e.g., screenreader) have to be applied to let the system ask for further conformance. The Recommendation is not applied automatically, to let the user decide on this part, and the OfflineSuggestion is performed only if the user consented through the PMT.

3.1 Rule-Based Matchmaker

The Rule-based matchmaker can resolve cases where the adjustments preferred by the user cannot be directly applied due to limitations of the current system/device. Typical problematic cases include the following: a) user has application-specific preferences for a solution that is not currently available, b) user has application-specific preferences for two or more different solutions of the same type (ex. screen readers) and all of these solutions are currently available. In both cases, the Rule-based Matchmaker exploits the knowledge and the inference capabilities of the Semantic Infrastructure, in order to find the best alternative solution, or to select the most suitable solution between the available solutions of the same type, respectively. The selected set of solutions to be launched is then passed to the Flow Manager.

Beyond that a user feedback loop is triggered by passing information about matchmaking results to the Flow Manager which will be presented to users allowing them to keep control about automatically applied configurations, benefit from additional recommendations, and get useful information as well as interaction help about assistive solutions that have been launched and adjusted. Apart from information representations, users shall be able to alter and assess decisions or apply recommendations on settings and solutions directly. Figure 2 illustrates interaction help information that will be presented for solutions that user has never used before.

Table 1 shows an example of a Jena rule executed by the Rule-based MatchMaker. According to this rule, if user has specific preferences for two different solutions of the same type and the first solution is installed while the second is available but not installed (e.g. it may be accessed through the internet), the installed solution is selected as the most appropriate for launching. Context is embodied in the Rule-Based Matchmaker as a set of rules to be executed before any other reasoning. This execution allows the final decision to take into account that under suboptimal conditions the final access mode may differ from the one stated by the user in the "Preferences Server" hence the access mode is weighted to give a chance to other solutions that work on different access modes. It also have a subset of rules dealing with perfomance, the surrounding environment changes very often (light, noise conditions) but not every little change in context drives a change on access mode, so some rules are needed to stop checking for different solutions. Table 2 below shows a JENA environment rule, when a sensors detects a change in the environment variable [25]. Two use cases were defined at [26] highlighting the contextual changes: adquiring the contextual data directly from the sensors embedded in the device or from ambient motes. As an example of the first one the user Märta that has problems seeing the screen of the devices she uses, when the luminosity changes, so when she logs in she will have a visual, white on black scheme. So, when the luminosity changes, Märta would like to automatically have some of the following changes at the interface of her devices. When the brightness of the environment reaches a certain threshold, she will receive a black on white scheme. When the brightness reaches another threshold (higher) she will receive an auditory UI.

Table 1. Jena rule example – Installed solutions have priority over available solutions

```
[InstalledSolutionsTakePriorityOverAvailableSolutions:
(?tmpUser rdf:type ns:TempUsers)
(?tmpUser ns:TempUsers_hasSpecificPreferencesForSolutions
?tmpSolutionsIDOneForWhichUserHasSpecificPreferences)
(?tmpUser ns:TempUsers_hasSpecificPreferencesForSolutions
?tmpSolutionsIDTwoForWhichUserHasSpecificPreferences)
notEqual(?tmpSolutionsIDOneForWhichUserHasSpcificPrferences,
?tmpSolutionsIDTwoForWhichUserHasSpecificPreferences)
(?tmpSolutionsIDOneForWhichUserHasSpecificPreferences
rdf:type ?tmpSolutionClass)
(?tmpSolutionsIDTwoForWhichUserHasSpecificPreferences
rdf:type ?tmpSolutionClass)
(?tmpEnvironment rdf:type ns:TempEnvironment)
(?tmpEnvironment ns:TempEnvironment_installedSolutions ?tmpSo-
lutionsIDOneForWhichUserHasSpecificPreferences)
(?tmpEnvironment ns:TempEnvironment_availableSolutions ?tmpSo-
lutionsIDTwoForWhichUserHasSpecificPreferences)
-> (ns:InstalledSolutionsTakePriorityOverAvailableSolutions
rdf:type ns:TempSolutionsToBeLaunched)
(ns:InstalledSolutionsTakePriorityOverAvailableSolutions
ns:TempSolutionsToBeLaunched_IDs ?tmpSolutionsIDOneFor-
WhichUserHasSpecificPreferences)]
```

Table 2. If sensors detect changes in the environment variables, then trigger a further step

```
[apply_New_Value_On_Trigger:
   (?tmpEnvironment rdf:type ns:TempEnvironment)
   (?tmpEnvironment ns:Temp_environment_ContextChange
              ?tmpContextChange)
   (?tmpEnvironment ns:Temp_environment_ContextProperty
  ?tmpContextProperty)
(?tmpContextProperty ns:Temp_environment_ContextProperty
?val1)
(?val1 rdf:type ns:Temp_environment_ContextProperty)
equal(?tmpContextChange,"true"^^http://www.w3.org/2001/XMLSche
ma#boolean)
 equal(?tmpContextProperty , ?val1)
-> (?val2 rdf:type ns:Temp_environemnt_ContextProperty)
(?tmpContextProperty ns:Temp_environment_ContextProperty
?vall2) ]
```

3.2 Statistical Matchmaker

There is no simple way to automatically translate application specific settings into the context of another application. While it would be possible to state a transformation of a setting from one application to another, it is difficult to automatically find these relations with algorithms. Font size, for instance, might be relatively easy to measure, so it would be possible to create a transformation expressing that operating system A always renders text five percent smaller than operating system B. Yet, both operating systems come with individual metaphors and UI styles that a matchmaker should try to maintain. Operating system B might always come with a relatively dense user interface, so font size should be even larger to make the information easier to read and separate. These subjective requirements cause an offset in the target configuration we want to infer. The goal is not just to reconstruct settings of application A for application B, but to infer settings for application B that make it 'feel and look like' application A. The strong subjective influence of 'feel and look like' is hard to describe with automated mathematical algorithms. Figure 3 illustrates this problem. A direct translation of user preferences from one application to another is possible if you can define a mathematical mapping. A very simple example would be that the font size rendering on Operation System A is 25 percent larger than rendering it on Operation System B (ignoring other influences on the actual physical font size in this example, like the display pixel density). Yet, those objective transformations will not always match what a user desires, as there are other influences to the "perceived clarity" of text items in an operation system. Operation System B might have a very clutter style, so that a user might want a larger font size for representing the same information. This also illustrates that those "subjective transformations" are user-specific.

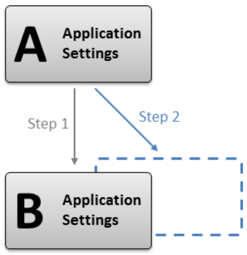


Fig. 3. Application Settings Transformation. Step 1 represents a direct transformation. Step 2 visualizes the offset originating in subjective requirements.

The task of the statistical matchmaker is to find a transformation that is as close to Step 2 in Figure 3 as possible. To solve this task, the statistical matchmaker deploys statistical inference. This process is based on recommender systems [27] and consists of several steps, subdivided in an offline and online section.The offline section of the statistical matchmaker, called the Statistical Inference module, iterates over all preferences sets known to the system and clusters them. This step is necessary, as the runtime section, running on a specific client device, might not have access to the preference servers at all. Further, there are obvious security limitations in User A accessing the preferences of User B.The idea behind the statistical matchmaker is to benefit of adaptations of other users that have similar preferences. If we continue the library example from above, the system might be able to infer a meaningful transformation, if it is able to find users that have similar settings for their smartphone, but also include settings for the library pc. Or, to put it more general: The statistical matchmaker tries to find preference sets of users that are similar for seen contexts, but also include settings that are close to the unseen context. Figure 4 illustrates this process.

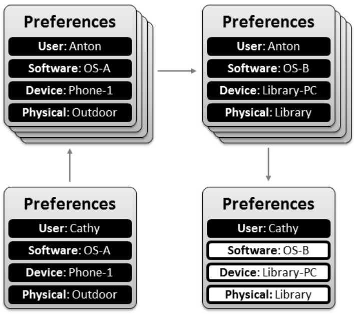


Fig. 4. Statistical Preference Matching

The offline clustering step takes care of reducing the amount of preference sets to measure against.The system starts with the preferences for user Cathy specific to her smartphone (preferences bottom left) and the target context it is supposed to infer to (Black on white context bottom right). In the first step, it compares the preference set to the known clusters of user preferences, trying to find a preference set that is close to Cathy's preferences for the smartphone. From all these preference sets that are close to Cathy's, those are extracted that also contain preferences for a context that is close to the target context. In this example, User Anton is found. It is now possible to either use Anton's preferences directly for Cathy's preference set, or to infer the transformation it required to get from Anton's smartphone preferences to Anton's library pc preferences and apply the same transformation to Cathy's preferences. Either process should generate a preference set that brings the system into a state so Cathy can interact with it and adjust it. Her adjustments will then serve as preferences for further inferences, just as Anton's just did.

Context in the Statistical Matchmaker is both simpler and more complex to embody. The simple part is that the context data is just added as more features in the feature vector. The complex one is that the ability of predicting the correct settings for a bigger feature vector [28] requires more samples or a better feature selection to be part of the predictor set [29.]

3.3 Context Aware Server

The context mechanism for fetching data from sensors might vary depending on the use cases. A first case is when the device that interacts with the user has itself the sensors – such as smartphones –, the second use case is when the device that the users interacts with, has not sensors to gather data – such as ATM –, then data is provided by motes around the device, collected and aggregated using a Context Aware Server platform.

The Context Aware Server (CAS) is developed in NodeJs provides a platform that collects, stores and process the data from sensors and when some triggers are fired it sends the processed data to a client in JSON format. Configuration, Sensors and Users are stored in MongoDb and new data is cached in Redis and it can be eventually saved to MongoDb depending on configuration.

The CAS is designed to be fully compatible to RESTful architectural design. It's structured in four main APIs: Sensor API, Device API, User API and Configuration API. The Sensor API offers the means for sending streams of new data from sensors and getting data stored in the CAS, it also permits advanced searching of data and the retrieval and search of sensors.

The CAS implements an adaptable and configurable triggering system that avails the definition of basic collection, aggregation and triggering rules that are applied when new user is in the system or when new data from sensor is received. All the triggering system is configurable through the configuration API and the Trigger API and it's applied in real time. When new data arrives to the Context awareness server, it fires an "onNewData" event. Afterwards, a listener gets the sensor configuration from the database.

Table 3. Sensor RESTFul API (excerpt)

Command	Parameters	Response sample	Description
GET /sensors/:id	?populate	{ "__v": 0, "_id": "524abe89d34ac49f24000001", "_last": { "value": 4, "at":"2013-10 03T14:18:49.554Z" }, "devid": "1", "type": "light" }	Gets sensor using the :id from database. _last references the last retrieved data stream, devid it's the internal sensor id (related to device) and type it's the type of sensor. ?populate=true parameter will populate the sensor with _last data timestamp and value.
GET /sensors/:id/ data	None, ?all, ?new,	[{ "at": "2013-04-22T00:35:43.12Z", "value": 1 },{ "at": "2013-04-22T00:55:43.73Z", "value": 2 }]	If the parameter is ?new, it retrieves new data since last POST operation from sensor :id, otherwise or with ?all parameter, retrieves all data from sensor :id.

3.4 Minimatchmaker

The Minimatchmaker is a component able to execute simple if-then-else software code. It is named after the Matchmakers but it doesn't share their computational power. The Minimatchmaker receives a series of "what-if" [30] rules from the real Matchmaker that allow it to react to minor changes in the environment.

Recognised Environment:When the data sent by the Environmental Reporter can be managed by the Minimatchmaker, there is no need for querying an external Matchmaker. The information with the new adaptation is composed inside the Minimatchmaker and sent to the Settings Handler.

Unrecognised Environment: Sometimes the data sent by the Environmental Reporter cannot be managed by the Minimatchmaker. In these cases the Minimatchmaker communicates to the internal flow manager (The orchestrator) that it is not able to find a set of settings to respond to the changes in the context, and the orchestrator sends the same context data to the Cloud Flow Manager through a http proxy (HTTP Client). What happens in the cloud is transparent to the components inside the user device architecture, but nowadays the Flow Manager send the data to the Matchmaker module, which is composed by a series of different Matchmakers, including a Rule Based Matchmaker, a Statistical Matchmaker and a Flat Matchmaker. The response from the http proxy is formed by the new settings plus a new set of rules to deal with the context internally. The settings are directly sent to its handler (System Settings Handler) while the rules are passed to the MiniMatchmaker. The MiniMatchmaker erases any previous set of rules and stores those that are being received.

4 Results and Discussion

After a manual pre-load of the desired response of the MMM for the users, the lab tests had the following setup: 1) two separate devices were held by the user's assistant, one configured with a visual-magnified UI and the other one with an auditory UI. 2) the auditory one was given to the user, and he starts an interaction to get a trip ticket from the UI. 3) the observer switches on a music player with loud volume; when the MMM receives this noise, it produces a speech "switching to visual UI". 4) the assistant switches the mobile with the user, and the user ends the ticket purchase.

The test comprised a generic concept validation questionnaire and specific questions about the perceived usefulness of the solution and the level of satisfaction of the user. One of our main concerns is to make sure that the adaptation capabilities of the system developed does not conflict with Nielsen's first usability heuristic (the user has to be always in control of the interaction), specific questions were asked regarding this issue. One out of three expressed their concern that the system autoadapting without their permission may be annoying, and that changes in the UI should be configured by users. Nevertheless, three users considered the autoadaptation capabilities as useful (mean 4 in a 5-point Likert scale) and considered that, if they can configure the adaptations they may use it in the long term (mean 3,67 in a 5-point Likert scale).

A prospective implementation on Android can be found at 17 though it lacks integration with the Flow Manager but served to ask users for general acceptance and notion of control issues. These questions were refined during the second pilot stage of the Cloud4all project and as soon as the processed responses are available, the matchmaking mechanism will be updated accordingly.

The Context weighting mechanism has the drawback of a loss of performance, tighter integration at the ontology level is being tackled to decrease response time. Finally new algorithms are being explored to improve the hybrid approach that is employed to decide which of the matchmakers provides the final output.

References

1. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5, 4–7 (2001)
2. Dey, A.K., Abowd, G.D., Salber, D.: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Journal Human Computer Interaction 16, 97–166 (2001)
3. Zimmermann, A., Lorenz, A., Oppermann, R.: An operational definition of context. In: Kokinov, B., Richardson, D.C., Roth-Berghofer, T.R., Vieu, L. (eds.) CONTEXT 2007. LNCS (LNAI), vol. 4635, pp. 558–571. Springer, Heidelberg (2007)
4. Zimmermann, A., Lorenz, A., Specht, M.: Applications of a Context-Management System. In: Dey, A.K., Kokinov, B., Leake, D.B., Turner, R. (eds.) CONTEXT 2005. LNCS (LNAI), vol. 3554, pp. 556–569. Springer, Heidelberg (2005)
5. Stephanidis, C.: Designing for All in Ambient Intelligence Environments: The Interplay of User, Context, and Technology. Intl. Journal of Human-Computer Interaction 25, 441–454 (2009)
6. Karim, S., Tjoa, A.M.: Towards the Use of Ontologies for Improving User Interaction for People with Special Needs. In: Miesenberger, K., Klaus, J., Zagler, W.L., Karshmer, A.I. (eds.) ICCHP 2006. LNCS, vol. 4061, pp. 77–84. Springer, Heidelberg (2006)
7. Gajos, K.Z., Weld, D.S., Wobbrock, J.O.: Automatically Generating Personalized User Interfaces with SUPPLE. Artificial Intelligence 174, 910–950 (2010)
8. INREDIS. Project website, `http://wiki.inredis.es`
9. ISO/IEC 24752-3:2008. Information technology – User interfaces – Universal remote console – Part 3: Presentation template (2008)
10. Iglesias-Pérez, A., Linaje, M., Preciado, J.C., Sánchez, F., Gómez, E., González, R., Martínez, J.A.: A Context-Aware Semantic Approach for the Effective Selection of an Assistive Software. In: Proceedings of IV International Symposium of Ubiquitous Computing and Ambient Intelligence, pp. 51–60 (2010)
11. SERENOA. Project website, `http://www.serenoa-fp7.eu`
12. Iglesias-Pérez, A., Peinado, I., Chacón, J., Ortega-Moral, M.: Frontiers in Context Modelling to Enhance Personalisation of Assistive Technologies. In: Assistive Technology: From Research to Practice – Proceedings of AAATE 2013. IOS Press (2013), doi:10.3233/978-1-61499-304-9-829
13. Koutkias, V., Kaklanis, N., Votis, K., Tzovaras, D., Maglaveras, N.: An Integrated Semantic Framework Supporting Universal Accessibility to ICT. Universal Access in the Information Society. Special Issue: 3rd generation accessibility: Information and Communication Technologies towards universal access, Springer

14. Loitsch, C., Stiegler, A., Strobbe, C., Tzovaras, D., Votis, K., Weber, G., Zimmermann, G.: Improving Accessibility by Matching User Needs and Preferences. In: Assistive Technology: From Research to Practice – Proceedings of AAATE 2013. IOS Press (2013), doi:10.3233/978-1-61499-304-9-1357
15. Melcher, V., Krüger, A., Chalkia, E.: Managing Preferences in the Cloud – Requirements and UI Design. In: Assistive Technology: From Research to Practice – Proceedings of AAATE 2013, IOS Press (2013), doi:10.3233/978-1-61499-304-9-1372
16. Clark, C., Basman, A., Markus, K., Zenevich, Y.: A Cloud-Scale Architecture for Inclusion: Cloud4all and GPII. In: Assistive Technology: From Research to Practice – Proceedings of AAATE 2013. IOS Press (2013), doi:10.3233/978-1-61499-304-9-1366
17. Iglesias-Pérez, A., Peinado, I., Chacón, J., Ortega-Moral, M.: Architecture for Adding Context-Aware Capabilities to Preferences-Oriented User Interfaces. In: Proceedings of DRT4All, 5th edn. Fundación ONCE, Madrid (2013)
18. Partarakis, N., Doulgeraki, C., Leonidis, A., Antona, M., Stephanidis, C.: User Interface Adaptation of Web-Based Services on the Semantic Web. In: Stephanidis, C. (ed.) UAHCI 2009, Part II. LNCS, vol. 5615, pp. 711–719. Springer, Heidelberg (2009)
19. Jung, C., Hahn, V.: GUIDE–Adaptive user interfaces for accessible hybrid TV applications. In: Second W3C Workshop Web & TV (2011)
20. Atkinson, M.T., Bell, M.J., Machin, C.H.C.: Towards Ubiquitous Accessibility: Capability-based Profiles and Adaptations, Delivered via the Semantic Web, pp. 2–6
21. Kadouche, Mokhtari, M., Giroux, S., Abdulrazak, B.: Semantic approach for modelling an assistive environment using description logic, pp. 222–231 (2008)
22. Ponsard, C., Beaujeant, P., Vanderdonckt, J.: Augmenting Accessibility Guidelines with User Ability Rationales. In: Winckler, M. (ed.) INTERACT 2013, Part I. LNCS, vol. 8117, pp. 579–586. Springer, Heidelberg (2013)
23. Wassermann, B., Zimmermann, G.: User Profile Matching: A Statistical Approach. In: CENTRIC, pp. 60–63 (2011)
24. Middleton, S.E., De Roure, D.C., Shadbolt, N.R.: Capturing knowledge of user preferences: ontologies in recommender systems. In: Proceedings of the 1st International Conference on Knowledge Capture, K-CAP 2001, New York (2001)
25. Serra, G., Iglesias, A., Kalogirou, K., Montalvá, J.: Context-related profile adaptation algorithms, Cloud4all deliverable, p. 52 (2013)
26. Iglesias, A., Peinado, I., Kalogirou, K., Chalkia, E.: Rule sets for the automatic adaptation of the user profile to context-related features, Cloud4all deliverable, pp. 20–22, pp. 31–41 (2013)
27. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel Methods in Machine Learning. The Annals of Statistics 36(3), 1171–1220 (2008)
28. Ahonen, T., Hadid, A., Pietikäinen, M.: Face recognition with local binary patterns. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 469–481. Springer, Heidelberg (2004)
29. Gomez-Verdejo, V., Martinez-Ramon, M., Arenas-Garcia, J., Lazaro-Gredilla, M., Molina-Bulla, H.: Support Vector Machines With Constraints for Sparsity in the Primal Parameters. IEEE Transactions on Neural Networks 22(8), 1269 (2011)
30. Chaudhuri, S., Narasayya, V.: AutoAdmin "what-if" index analysis utility. In: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, SIGMOD 1998, pp. 367–378. ACM, NY (1998)