A FRAMEWORK FOR FLEXIBILITY AT THE INTERFACE
Joining Ajax Technology and Semiotics

Frederico José Fortuna[1], Rodrigo Bonacin[2] and Maria Cecília Calani Baranauskas[1]
[1]*Institute of Computing, University of Campinas, UNICAMP, Caixa Postal 6176, 13083-970, Campinas, Brazil*
[2]*CTI Renato Archer and Faculty of Campo Limpo Paulista, FACCAMP, MCT*
Rodovia Dom Pedro I Km146, 6 13082-120, Campinas, Brazil

Keywords: Flexible Interfaces, Adjustable Interfaces, Tailoring, Norms, Human-computer Interaction.

Abstract: Different users have different needs and, especially in the web context, the user interfaces should deal with these different needs. In this paper we propose a framework to support the design and development of flexible, adjustable interfaces; the framework is grounded in the semiotic concept of norms and is developed using Ajax technology. The framework was tested within the Vilanarede system, an inclusive social network system developed in the context of e-Cidadania project. The test helped us to identify the advantages and drawbacks of the proposed solution and the users provided us with important feedback to improve it.

1 INTRODUCTION

In a web environment there are different users with singular needs. They have different hardware, software, network infrastructure, native language, culture, geographical location and physical or cognitive ability. These aspects impact on both social and technological issues. Because of these differences user interfaces should be flexible.

Web interfaces today, in general, are not flexible. Most of them are static but there are some exceptions such as iGoogle, Pageflakes, Netvibes and MyYahoo! However the flexibility provided by these interfaces is not enough when a vast and diversified context of users is considered; the adjustments are restricted to a short set of possibilities and they are the same for all the users. In addition, those interfaces do not consider the social issues and they are not able to infer the user needs.

Nowadays, adjustable web interfaces are typically constructed using the Ajax technology (Ajax stands for "Asynchronous JavaScript and XML"). This technology enables developers to make changes to a web interface without the need to reload the page, what makes Ajax a very useful technology for the development of flexible interfaces.

Ajax is a powerful technology; however it doesn't help in informing which interface elements should be adjusted and how it should be done to fit the user needs. Consequently it is necessary a theoretical referential, a design methodology and a development framework to encompass those adjustments.

We argue that Semiotics can provide us with theoretical and methodological fundamentals for this task (Bonacin et. al 2009). This work proposes to employ the concept of norms, from MEASUR (Methods for Eliciting, Analyzing and Specifying User's Requirements) (Stamper, 1993), to describe which elements should be adjusted, and how and in which situation they should be adjusted for every user or group of users.

Norms describe the relationships between an intentional use of signs and the resulting behavior of responsible agents in a social context; they may also describe beliefs, expectations, commitments, contract, law, culture, as well as business. Norms are useful and powerful because they may be used to describe important aspects of the user context.

Our hypothesis is that if we put together Ajax and norms we potentially create really flexible interfaces. So in this paper we present a framework to develop flexible interfaces using Ajax and norms. This framework has been applied in the Vilanarede social network system, in order to validate it. Vilanarede is a social network system developed in

José Fortuna F., Bonacin R. and Cecilia Calani Baranauskas M. (2010).
A FRAMEWORK FOR FLEXIBILITY AT THE INTERFACE - Joining Ajax Technology and Semiotics.
In *Proceedings of the 12th International Conference on Enterprise Information Systems - Human-Computer Interaction*, pages 30-37
DOI: 10.5220/0002889700300037

the context of the e-Cidadania project (Baranauskas, 2007) in order to help creating a digital culture among as many categories of users as possible, people with different needs and interests.

The paper is structured as follows: Section 2 presents the background and clarifies terms and concepts used in this work; section 3 presents the proposed framework including its architecture and implementation technology; section 4 presents and discusses results from the application of the framework; section 5 concludes this paper.

2 BACKGROUND

In this section we detail the main background work used to delineate the proposed framework. Section 2.1 introduces the concept of flexible interfaces, including a short overview on the research work on this field. Section 2.2 presents the Ajax concepts and features. Section 2.3 presents the concept of Norms and prior attempts to use norms in order to provide flexible interfaces, which support this work.

2.1 Flexible Interfaces

According to the IEEE standard computer dictionary, flexibility is the ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed (Institute of Electrical and Electronics Engineers, 1990). From this definition, we could expect that flexible user interfaces should be easily modified for different contexts of use. We understand that to meet this requirement, both technical and social issues must be addressed in designing the system and the interaction.

Literature in Human–Computer Interaction (HCI) has pointed out different alternatives to provide flexibility in the user interface. Dourish (1996) argues that we cannot dissociate the technical and social disciplines in systems design. The relationship between them is considerably more intricate than the traditional view suggests, where requirements and constraints do not have consequences beyond the application, and sociological insights do not apply inside the system.

Tailoring is a concept closely related to our notion of flexibility; it can be understood as ‘‘the activity of modifying a computer application within the context of its use’’ (Kahler et al. 2000: 1). Research in tailorable systems usually presupposes that the user should be in charge of the tailoring task (Teege et al. 1999; Morch and Mehandjiev 2000). A challenge in tailoring is how to provide users with high-level interfaces in which relevant changes can be made quickly without the necessity of advanced technical skills.

Among other approaches, the context-aware applications (Moran and Dourish, 2001) collect and utilize information regarding the context in which provision of services is appropriate to particular people, place, time events, etc. Context refers to the physical and social situations in which computational devices are embedded. Fischer et al. (2004) have proposed the concept of ‘‘Meta-Design’’, a vision in which design, learning, and development become part of everyday working practices of end-users.

2.2 Ajax – Rich Interfaces

Ajax can be understood as a new style to develop web applications, which includes several web technologies. It is not a new technology by itself, it emerged in software industry as a way to put together some existing technologies in order to provide “richer web interfaces”.

The term Ajax was coined by Garrett (2005) and incorporates the following technologies:

- Standards-based presentation using XHTML and CSS;
- Dynamic display and interaction using the Document Object Model;
- Data interchange and manipulation using XML and XSLT;
- Asynchronous data retrieval using XMLHttpRequest;
- JavaScript binding everything together.

Ajax changes substantially the client-server interaction. The classic synchronous model where each user request is followed by a page reload is substituted by an asynchronous independent communication where the user does not need to wait in a “black page” for the server processing.

Part of the processing can be executed at the client side. This aspect can minimize the server and mainly the communication workload. Many applications that initially were restricted to desktop platforms due to the limitations of the standard html development can be transposed to the web by using Ajax resources. For example, we can drag and drop images, apply filters on them and change their sizes without reloading the page. Regarding the interface flexibility, Ajax enables to change the features,

remove or include any interface object at a webpage without the necessity of reloading the page.

Some of the first very popular applications using Ajax are Google Suggest, Google Maps and Gmail. Nowadays, Ajax is widely used in web development and many platforms and development tools support Ajax. The World Wide Web Consortium (W3C) is also now working to standardize key Ajax technologies such as the XMLHttpRequest specification (W3C, 2009).

2.3 Norms and Flexible Interfaces

From the conceptual and theoretical points of view, the proposed framework is based on Norm Analysis Method (NAM) from MEASUR. Norms describe the relationships between an intentional use of signs and the resulting behaviour of responsible agents in a social context; they also describe beliefs, expectations, commitments, contract, law, culture, as well as business.

"Norms can be represented in all kinds of signs, whether in documents, oral communication, or behaviour, in order to preserve, to spread and to follow them. However, one cannot always put one's hands conveniently on a norm, as one might grasp a document that carries information through an organization. A norm is more like a field of force that makes the members of the community tend to behave or think in a certain way." (Stamper et al. 2000, p. 15)

Besides the description of the agents' responsibilities in the organisational context, Norm Analysis can also be used to analyse the responsibilities of maintaining, adapting and personalising the system features.

Norms can be represented by the use of natural language or Deontic Logic in the late stages of modelling. The following format is suitable for specifying behavioural norms (Liu, 2000):
<Norm>::= whenever <condition> if <state> then <agent> is <D> to do <Action>
Where <D> is a deontic operator that specifies that the action is obligatory, permitted or prohibited. The norms are not necessarily obeyed by all agents in all circumstances; they are social conventions (laws, roles and informal conventions) that should be obeyed. For example: a norm specifies that the agents are obliged to pay a tax; if an agent has no money it will not pay, but usually there is a cost when an agent does not obey the norms.

Bonacin et al. (2009) present the foundations, a framework, and a set of tools for personalized service provision using norms simulation. Part of their framework is used in this proposal; however their framework is limited by using "thin client" solutions, since a page reload is necessary for each interface change. This work expands the possibilities from the technological and practical points of view by using the Ajax technology. In addition it is provided an easier and more productive way to develop flexible interfaces. In this work norms can be evaluated during interaction with the system and be immediately reflected in interface changes.

We propose to simulate the norms and deduce the expected behavior of users and organizations during the user interaction (real-time). For example, if someone accesses an e-Government portal, and is obliged to pay a tax, then he/she will probably be interested in accessing information about the payment procedure. So the system can be immediately adapted according to the user needs and preferences.

Norms are susceptible to changes in the organizational context and in carrying the intentions of agents in society. In the proposed approach, domain specialists, designers and users maintain the norms specifications according to the changes in their socio-pragmatic (Stamper, 1973) context.

3 THE FRAMEWORK PROPOSAL

In order to help the development of flexible interfaces we propose FAN: an Ajax framework using norms. FAN stands for Flexibility through Ajax and Norms. The framework's architecture is described in section 3.1 and its execution aspects are explained in section 3.2.

3.1 Framework Architecture

The framework we propose provides powerful capabilities but it has a relatively simple architecture which is shown in Figure 1. Basically, the framework is composed of four modules: Perception, SOAP Client, Action and Users' Facts Storage. These modules are responsible for tasks such as creating, storing and loading contexts and facts about the users, capturing events on the presentation layer (interface, which is usually constructed with HTML and CSS), accessing NBIC/ICE web services (which maintain and interpret Norms) and customizing the interface.

Regarding the development technologies, the Perception, SOAP Client and Action modules were

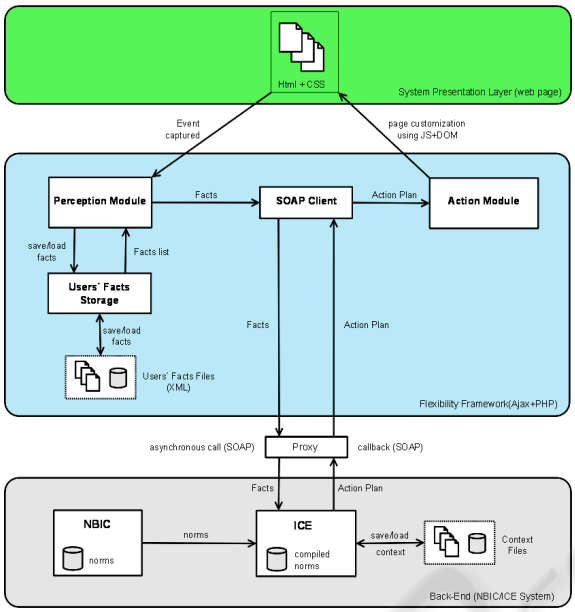

Figure 1: Framework architecture.

developed using Ajax, running on the client side, because of its potential and capacity of giving developers capabilities to modify a web page without the need of reloading it, and the Users' Facts Storage module was developed in Ajax and PHP. The PHP code runs on the server side to store facts related to every user.

The Perception module is responsible for capturing events generated by the users on the presentation layer, such as mouse clicks over buttons or images. Also it is responsible for generating facts related to these events. Examples of facts: the user has difficulty to use the mouse due to many clicks on non-clickable regions or the user is a sign language reader due to many demands on sign language videos.

The SOAP Client module is responsible for the communication between the proposed framework and NBIC/ICE System, which is responsible for maintaining and evaluating the norms. This module handles synchronous and asynchronous requests made by the Perception module and it accesses NBIC/ICE web services using SOAP calls.The third module, Action, is responsible for changing the system's interface according to an Action Plan generated by ICE. An Action Plan is a XML that stores interface objects IDs that should be modified and how the modifications should be done, which attributes of an interface object should be modified, what are new values of these attributes or a JavaScript code that must be executed.

The last module, Users' Facts Storage, is in charge of storing, loading and erasing users' facts generated by the Perception module. It stores facts on XML files associated to each user and/or each system page.

The NBIC/ICE System has two components: NBIC and ICE. NBIC stands for *Norm Based Interface Configurator* and it is responsible for storing and managing norms. ICE means *Interface Configuration Environment*. This component is responsible for making inferences, using Jess inference machine, over norms stored by NBIC and creating action plans, according to the inferences made, describing how the interface should be customized. In other words, which interface elements should me modified and how it should be done or which elements should be removed or added.

A proxy is used in the communication between the framework and NBIC/ICE System in order to avoid a JavaScript-related security issue. Some web browsers don't execute JavaScript codes stored in a Server different from the Server that hosts the current web Page.

3.2 How it Works

Every time a page is loaded the Perception module creates a new context by making an asynchronous call to ICE. A context stores information such as the user id, the system name (or system id) and some information about the page that the user is visiting.

After the context is created or loaded, the norms related to the system are loaded in that context. After creating the context and loading the norms, the Perception module makes a request to Users' Facts Storage module to load all the facts related to the current user and current web page (node). If the Users' Facts Storage finds facts of the current user and node, this module will build a list with these facts, otherwise it will build an empty list. This list is, then, returned to the Perception module. If the list is not empty, this module will start the customization process described as follows, for each fact in order to adjust the interface so that it looks and behaves like in the last time the user visited or adjusted that page.

When an event on the interface (e.g. mouse click over a button) is captured by the Perception module, as shown in Figure 2, this module creates predicates related to the event and the actual context.

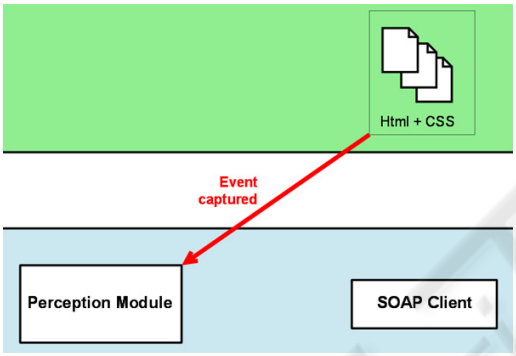

Figure 2: Event captured by Perception.

After creating predicates, the Perception module uses an ICE service, making an asynchronous call, via SOAP Client module, to generate a new fact based on the generated predicates, as shown on Figure 3. A fact is basically a collection of predicates. Also the Perception module requests Users' Facts Storage to store the predicates on a XML as a new fact so that the new adjustment may be saved.

After creating a new fact, ICE starts to make inferences over the norms, stored by NBIC and related to the context and fact, and then it builds an action plan, an XML file that stores interface objects IDs that should be modified and how the modifications should be done. The action plan is then sent to SOAP Client module, which calls the Action module to modify the interface according to the action plan generated by ICE. These steps are shown in Figure 4.

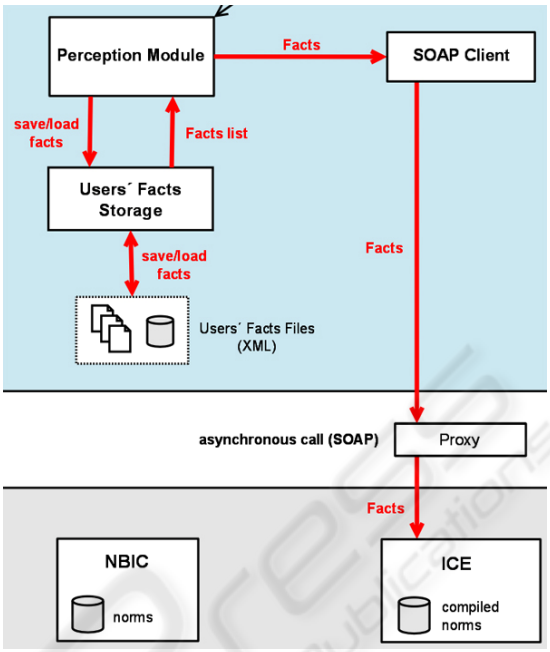

Figure 3: Perception making requests to Users' Facts Storage and ICE.

The Action module parses the XML file (action plan), as shown in Figure 5, to look for interface objects ids and attributes values or JavaScript code that should be executed. This module then accesses the found interface objects using DOM and modifies their attributes according to the new values found on the action plan. Also, this module executes any JavaScript code found in the Action Plan.

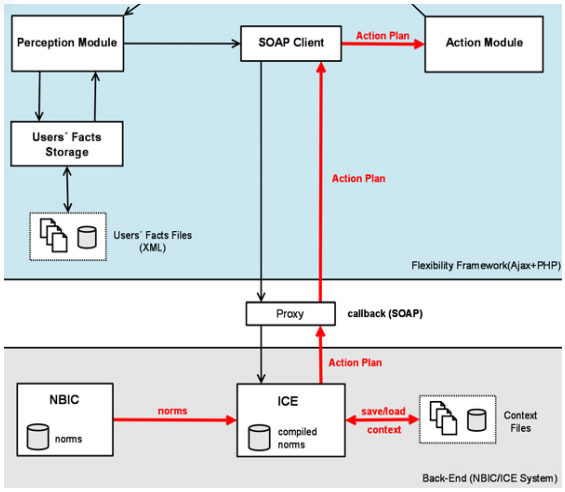

Figure 4: NBIC/ICE generating an Action Plan.

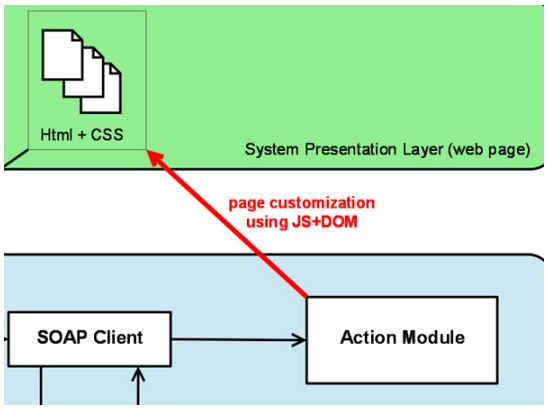

Figure 5: Action module customizes the interface according to the Action Plan.

4 APPLYING THE FRAMEWORK

The framework is being used and tested in the Vilanarede system. Vilanarede is a social network system developed in the inclusive context of e-Cidadania project.

The framework for flexibility we propose should help Vilanarede system to achieve its objectives because it may give each user the possibility to adjust the system interface according to his or her needs or interests. As an example, the framework may enable a user to change the type of the main menu, changing the way he or she will see it. Using inference the framework may discover that a user has some kind of sight problem and them the framework may change the size of interface elements like images and texts, making them bigger.

The framework was tested for the first time with users on a participatory activity organized by the e-Cidadania project with Vilanarede users. During the activities, 7 users were asked to do some tasks at Vilanarede system or tasks related to the system. After each participatory activity researchers analyzed the data collected.

Further analysis of users' facts files showed that at least 21 of approximately 270 users of Vilanarede system already made at least one adjustment in the interface, using the framework.

4.1 Tests and Results

On the activity organized to test the framework, practitioners used a notebook station with Vilanarede system already loaded, as shown in Figure 6, and they were asked to fill their profile at Vilanarede with some personal information such as their level of education, level of experience with web navigation and how frequent they change the font size in the system. If a user selected that he or she makes the page font bigger most of the time or always, the framework would infer that it would be useful to make the font bigger for that user, so the font would be bigger on every page as soon as the user saved his profile.

After filling profile information, users were asked to change the type of the page's main menu from a default list of icons to a circular menu. In order to do this, users should first click on a special button in the page so that adjustment buttons would be shown on the page. These buttons would enable the user to modify some interface elements. After that, users should find and click on the correct button to change the menu type from a linear list to a circular set of options. Figure 7 shows the menu types and the buttons to change from one type to the other.

Figure 6: Station set used by Vilanarede users to test the framework.

Everything the users said and their emotions during the activity were recorded by a digital camera and a webcam. All the things they did on the interface (mouse movements, clicks etc) were recorded with Camtasia Studio, a desktop screen recorder. Also, e-Cidadania researchers observed the users during the activity and they wrote down on paper forms some important information regarding their interaction with the system. They wrote information such as whether a user noticed the font becoming bigger (automated adjustment) depending on the information he or she filled in his or her profile and how difficult it was for each user to locate the special button that enables adjustment buttons and how difficult it was for them to change the main menu type and the steps the users took to complete the task. Also, researchers observed users' suggestions to change the appearance and position of that special button and which new adjustments they would like to be able to make on the interface.

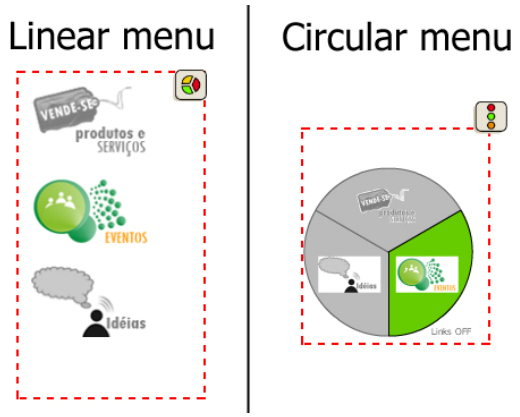

Figure 7: Alternative ways to see the Vilanarede's main menu – a linear way and a circular one.

The data collected by the researchers, the cameras and the screen recorder showed that almost all the users who tested the adjustments provided by the framework enjoyed being able to adjust the interface according to some of their interests and needs, as one of the users said "it's very good to have the possibility to change the interface". Also, the data showed that the majority of the users preferred the new type of menu offered by the framework.

Despite the advantages of using the framework and the good and interesting data collected, before and during the activity it was possible to identify some problems, some drawbacks and the users pointed out some aspects of the adjustment process that could be improved. Many users didn't like the appearance and position of the button they needed to click on in order to make adjustment buttons visible. In general, they would like to see a bigger button in a different position so that it would be easier to find, indentify and click on it.

During the activity it was possible to notice that one of the drawbacks of the framework is performance. The framework may take a few seconds to adjust a system's interface if the system has many complex norms or if the NBIC/ICE server is not so fast. The framework may also have problems with performance especially if the user (client) has a very poor internet connection as the framework may suffer from connection timeouts if packets got lost or corrupted or if they take so much time in their way from the client to the server and back.

Another drawback is that it is not easy to synchronize the requests made by the perception module to ICE/NBIC via SOAP Client so that ICE/NBIC returns the correct and expected data. Also, the framework enables developers to make almost all elements in the interface adjustable, using simple norms, but developing JavaScript code to implement the adjustable behavior of certain elements may be not that simple.

5 CONCLUSIONS AND FUTURE WORK

Different users have different needs and it is essential that user interfaces meet these different needs. In order to deal with this diversity, interfaces should be adjustable. To help the design and development of flexible, adjustable interfaces we proposed a framework, developed using Ajax and PHP, grounded in the concept of norms. This framework was successfully used and tested in Vilanarede system during a participatory activity organized by e-Cidadania project. The test helped us to identify the advantages and drawbacks of using the framework and the users gave us important feedback to help improving the framework.

As future work a norm modeler using web technologies is being developed in order to make it easier and more accessible for developers to include new norms into NBIC or changing the existing ones. In order to enhance the performance of the process of changing the interface according to users' needs, we could look for a more powerful and faster server with a faster broadband internet connection to run NBIC/ICE system as it is running, at the moment, in an old Pentium4 server with a limited broadband connection.

One powerful idea we consider as future work is to integrate the framework with semantic web. Instead of storing norms and facts inside files or databases at NBIC/ICE system we may use technologies like RDFS or SWRL (Semantic Web Rule Language) to store norms and facts on the web. Queries over norms may be done using SparQL, a RDF query language. Instead of using NBIC/ICE to make inferences over norms we may use RDF/RDFS, OWL and SWRL to do it. That way, users may store information about their needs (facts) on RDF files and store them on the web. The framework may use the facts stored on these files to make inferences and adjust a system's interface according to users' needs.

ACKNOWLEDGEMENTS

This work is funded by Microsoft Research - FAPESP Institute for IT Research (proc. n. 2007/54564-1 and n. 2008/52261-4).

REFERENCES

Baranauskas, M.C.C.: *e-Cidadania: Systems and Methods for the Constitution of a Culture mediated by Information and Communication Technology*. Proposal for the Microsoft Research-FAPESP Institute, (2007).

Bonacin, R. ; Baranauskas, M. C. C. ; Liu, K. ; Sun, L. . *Norms-Based Simulation for Personalized Service Provision*. Semiotica (Berlin), v. 2009, p. 403-428, 2009.

Dourish, P. (1996) *Open implementation and flexibility in CSCW toolkits.* Ph.D. Thesis, University College London, ftp://cs.ucl.ac.uk/darpa/jpd/dourish-thesis.ps.gz, last access October 17, 2007.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A.G., and Mehandjiev, N. (2004) Meta-Design: a Manifesto for End-User Development. *Communications of the ACM*, vol. 47, Issue 9.

Garrett, J.J. *Ajax: A New Approach to Web Applications*, Adaptive Path, February 18, 2005. http://www.adaptivepath.com/ideas/essays/archives/000385.php, last access December, 21, 2009.

Institute of Electrical and Electronics Engineers. (1990) *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. New York, NY: 1990

Kahler, H, Morch, A., Stiemerling, O. and Wulf, V. (2000) Special Issue on Tailorable Systems and Cooperative Work, *Computer Supported Cooperative Work*, 1, 9, Kluwer Academic Publishers.

Liu K. (2000) *Semiotics in information systems engineering*, Cambridge University Press.

Moran, T. and Dourish, P. (2001) Introduction to the special issue on Context-Aware Computing. *Human-Computer Interaction*, 16 (2), 2-3, Lawrence Erlbaum Associates.

Morch A. e Mehandjiev N. (2000) Tailoring as Collaboration: the Mediating Role of Multiple Representations and Application Units. Kahler, H, Morch, A., Stiemerling, O., Wulf, V. (Eds), Special Issue on Tailorable Systems and Cooperative Work, *Computer Supported Cooperative Work*, v. 9, Issue 1, Kluwer.

Stamper, R. K. (1973) *Information in Business and Administrative Systems*, NY, USA: John Wiley & Sons, Inc.

Stamper, R., 1993. Social Norms in Requirements Analysis – an outline of MEASUR. In Jirotka, M., Goguen, J. and Bickerton, M. (eds.), *Requirements Engineering, Technical and Social Aspects*. Academic Press. New York.

Stamper, R., Liu, K., Hafkamp, M. and Ades, Y., (2000) Understanding the roles of signs and norms in organizations – a semiotic approach to information system design, *Behaviour & Information Technology*, 19, 1, 15-27, Taylor & Francis.

Teege, G., Kahler, H. and Stiemerling, O. (1999) Implementing Tailorability in Groupware, In *SIGGROUP Bulletin*, 20, 2, 57-59, Association for Computing Machinery.

W3C, XMLHttpRequest, *W3C Working Draft*, November, 19, 2009, http://www.w3.org/TR/XMLHttpRequest/, last access December, 21, 2009.