Multiple Metaphor Environments: Issues for effective interaction design

C. Stephanidis and D. Akoumianakis

Institute of Computer Science, Foundation for Research and Technology-Hellas (FORTH)
Science and Technology Park of Crete
GR-71110, Heraklion, Crete, Greece
Tel.: +30-81-391741, Fax: +30 - 81 - 391740
emails: cs@ics.forth.gr, demosthe@ics.forth.gr

Abstract. This position paper presents the notion of multiple metaphor environment and discusses principles and techniques for constructing user interfaces as multiple metaphor environments. Though, multiple metaphor environments represent a generic concept, they are particularly relevant to novel application domains and technologies, such as Digital Libraries.

1. Introduction: Metaphors and the user interface

The notion of a metaphor in interface design is increasingly becoming a critical aspect in the attempt to provide more effective and higher quality interaction between humans and artefacts. Though several efforts have been devoted to the study of metaphors (e.g., Carroll et al., 1988; Henderson et al., 1986; Moll-Carrillo et al., 1995), very little is known as to how they can be systematically embedded into computer-based interactive software. At the same time, the number and diversity of application domains, in which the use of metaphors is critical, continuously increases (examples include, educational software, digital libraries, home-based interaction environments, virtual and augmented realities, health care records, electronic commerce, etc). A closer look into these application areas and the study of success of respective systems, reveals that, whereas in the past the use of metaphors was at the discretion of the designer, or in the best of the cases, bound to what the underlying development toolkit offers (i.e., trashbins, form filling), today and for certain non-traditional / non-business applications, embedding metaphors to interface design is compelling for the wide adoption and user acceptance of the application.

The use of metaphor may be studied at various levels, ranging from the overall interactive environment offered by an application, to the task level (i.e., how users engage and perform specific goal-oriented activities), as well as the physical level of interactions (i.e., icons used to convey intended meaning). Moreover, it is important that each of those levels may not involve the articulation of the same real world metaphor, but variants of different ones. Thus, at the level of the overall interactive environment, users may be exposed to a books-like (Moll-Carrillo, et al., 1995) or rooms-like (Henderson et al., 1986) metaphor, while in order to accomplish specific

tasks (such as for example, deletion of a file) alternative metaphors (i.e., deleting a file from a folder) may be recruited.

This leads to the conclusion that progressively interactive computer-based applications move towards a state which can be characterised as *multiple metaphor environments.* The notion of a multiple metaphor environment was firstly introduced and elaborated in the context of FRIEND21 (a major Japanese collaborative research and development project). There, it was claimed that interactive systems capable of mapping concepts from a source domain (i.e., database search operation) to different / multiple target domain functions (e.g., newspaper or HTML-based search), and vice versa, provide multiple metaphor environments. However, the main thrust of work in FRIEND21 was mainly conceptual and, as a result, did not deliver any detailed insight into how such systems may be specified, designed or implemented, other than guidelines for the human interface of the next century (Institute for Personalised Information Environment, 1995).

In this position paper, we revisit the notion of multiple metaphor environments from a slightly different view angle. In particular, we are interested to investigate the contributions of multiple metaphor environments to the design and development of user interfaces for different user groups, including people with disabilities (Stephanidis, 1997; Akoumianakis, et al., in press). Our objective is to draw upon recent experience, in the context of collaborative research and development projects, and shed light into the way in which (i) metaphors may become embedded into user interfaces, (ii) multiple metaphor environments may be specified, realised and implemented, and (iii) the above impact on the architectural abstractions of user interface software. The presentation of the relevant issues will be complemented by reference to example case studies in which multiple metaphor environments have been designed and implemented. In particular, examples, involving the fusion of conventional visual desktop interaction, augmented interaction elements (Savidis et al., 1997), such as scanning, and alternatives interaction environments, such as non-visual Rooms (Savidis et al., 1995), will be discussed with the view to reveal characteristic properties of such interactive software, development tool requirements, and prospective challenges (Stephanidis, 1997).

2. Multiple metaphor environments

In the context of the present work, metaphors are considered to have a two-fold purpose in interface design. They can either be embedded in the user interface, or characterise the overall interactive environment of an application. For example, the menu interaction object class, as commonly encountered in popular user interface development toolkits, follows the “restaurant” metaphor, and provides an example of embedding metaphor into a user interface. This is because it is commonly found as embedded element in systems conveying radically different interactive embodiments of the computer; examples are the visual desktop as in Windows95TM, rooms as in (Henderson et al., 1986), or book as in (Moll-Carrillo et al., 1995).

Alternatively, a metaphor may characterise the properties and the attitude of the overall interaction environment. For instance, the visual embodiment of the desktop metaphor in Windows95TM presents the user with an interaction environment based on high level containers, such as sheets of paper called windows, folders, etc., which characterise the overall interactive embodiment of the computer. Systems, such as those in (Henderson et al., 1986) or (Moll-Carrillo et al., 1995), are examples of alternative embodiments of real world metaphors into a user interface. It should be noted that a particular real world metaphor may have different interactive instantiations. Thus, for example, OSF/MotifTM and Windows95^{TM} support variations (mainly in the look and feel) of the visual embodiment of the desktop metaphor. From the above, it follows that the interactive environment of a metaphor is realised by specific user interface development toolkits.

Different interaction metaphors may be facilitated either through the enhancement, or augmentation of existing development toolkits, or by developing new toolkits. For instance, an enhancement of the interactive environment of a metaphor may be facilitated by introducing new composite object classes, such as the note cards in prevailing Windows-like systems, or by embedding in the toolkit additional interaction techniques, such as automatic scanning facilities for interaction object classes (Savidis et al., 1997). What is important to note about enhancement, or augmentation is that it rarely alters the overall interactive environment of the metaphor. This is because, the scope of the enhancement, or augmentation does not account for top-level container object classes (such as a window in Windows95TM, the room in (Henderson et al., 1986) or the book in (Moll-Carrillo et al., 1995). Instead, through sub-classing, augmentation extends the range of simple or composite interaction elements that may be supported in a toolkit's object hierarchy.

In case that an alternative interaction metaphor needs to be supported, then it may be realised through new toolkits. An example of the latter case is reported in (Savidis et al., 1995; Savidis et al., in press) where Commonkit is used to support user interaction based on a non-visual embodiment of the Rooms metaphor through speech and / or Braille output and keyboard input. COMMONKIT offers the full range of programming features encountered in currently available GUI toolkits, such as hierarchical object composition, dynamic instantiation, call-back registration and event-handling. In its current version, COMMONKIT implements only one container, namely Room, and several object classes (e.g., floor, ceiling, front / back / left / right wall), in addition to conventional objects, such as menu, toggle (represented as on / off switch), button, text reviewer, etc. A more elaborate account of the object library of COMMONKIT, as well as applications built with it, can be found in (Savidis et al., in press).

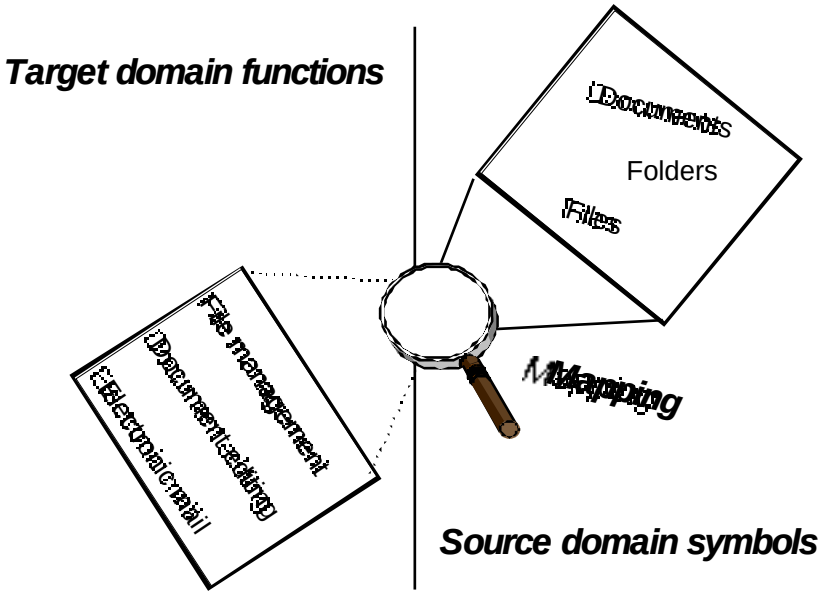

Figure 1: Concept of source and target domains

Following the above, the notion of a multiple metaphor environment implies a particular computer-based embodiment of an integrated system, capable of performing context-sensitive mapping between functions in a target domain (e.g., functions of a computer environment) to symbols in a source, or presentation domain (e.g., the desktop interactive embodiment), and vice-versa (Figure 1). Alternatively, it may be conceived as an integrated multiple toolkit platform, capable of context-sensitive mapping. For example, consider typical functions such as file management, electronic mail and editing, as performed in a computer environment (target domain). Such functions in the target domain are mapped onto user operations on objects (i.e., folders, documents, drawers) of the source domain, namely the desktop.

The visual desktop embodiment in current computer systems performs precisely such mappings between symbols from a target domain to symbols in the designated source domain. However, the visual desktop, as embedded in currently available user interface development environments, does not satisfy the conditions of multiple metaphor environment, since it does not perform any context-sensitive processing to map functions from the target domain to corresponding symbols in the source domain. This is because the source domain is fixed and unique (i.e., the desktop of an office). In other words, there is no possibility to map a file management function onto a book operation, and vice versa. Consequently, the construction of multiple metaphor environments reflects two important properties, namely the explicit embodiment of alternative metaphors (i.e., desktop, book, library) into the user interface, as well as their fusion into an integrated environment (i.e., context-sensitive mapping).

To demonstrate the principles underpinning the design and development of multiple metaphor environments, let us assume three users, namely a sighted user, a child and a blind user. All three are tasked to carry out a file management operation, namely delete a file. Since the capabilities of the users differ (e.g., with regards to the modalities that may be employed to facilitate the interactive task), the interface should ideally

undertake the required transformation so as to present an appropriate (i.e., accessible and usable) instantiation, suitable for each user.

Figure 2 depicts indicative examples of plausible alternatives which can be realised in a programming-intensive manner, by providing separate interface implementations for each user. Alternatively, the same interface could be built, in such a way, so that it is capable of context-sensitive processing leading automatically to the undertaking of suitable transformations to map the file management operation onto appropriate interactive environments, such as those depicted in Figure 2.

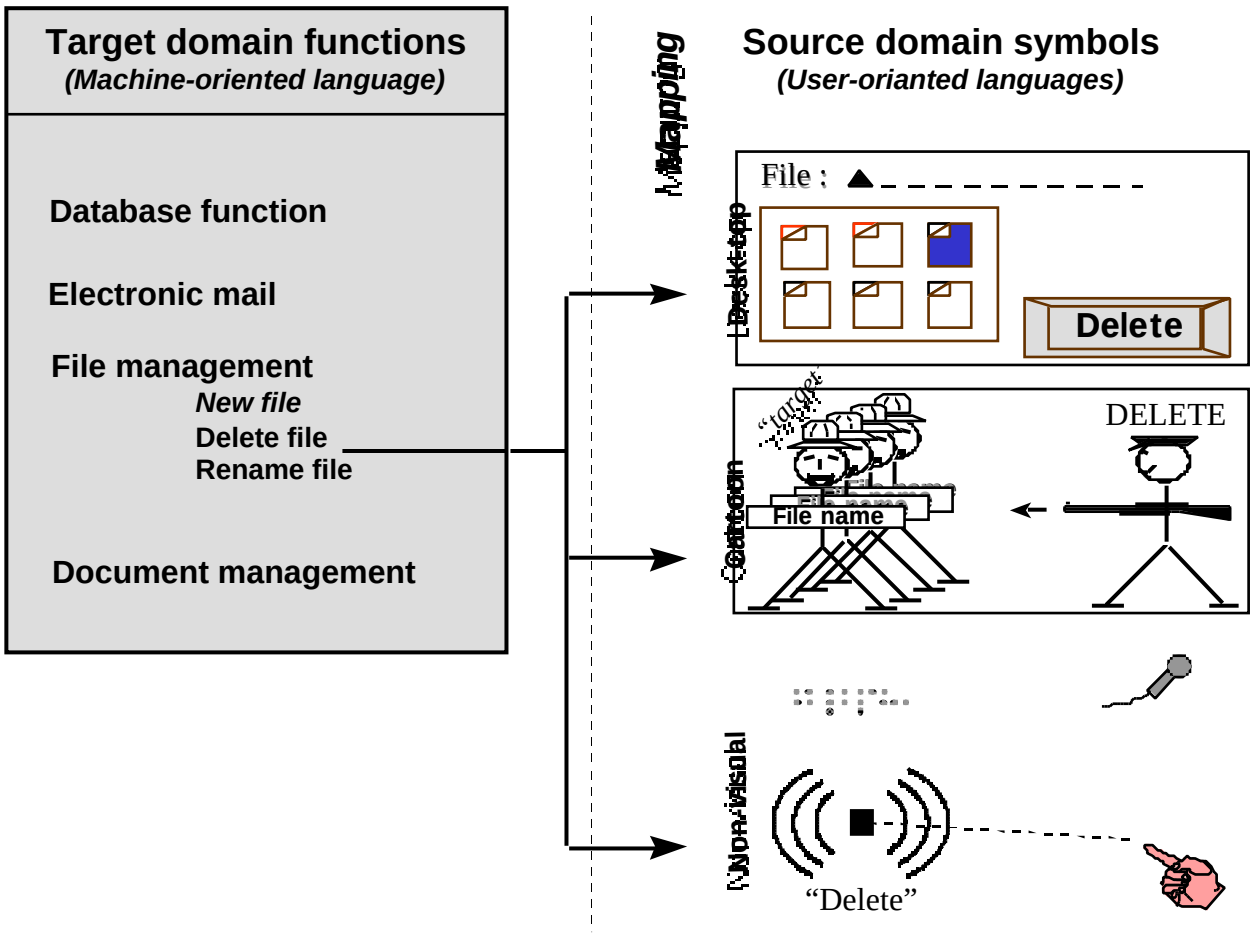

Figure 2: Mapping target domain functions to source (presentation) domain symbols

3. Conclusions

From the above, it follows that multiple metaphor environments are necessitated from the diversity of users (i.e., diverse requirements of different target user groups), the diversity of contexts of use (i.e., the variety of contexts in which artefacts may be encountered) and the diversity of interaction platforms (i.e., proliferation of different interaction toolkits), all of which may necessitate sometimes radical changes in the design. As a result, the important features characterising such environments are that: (a) there is a clear separation between knowledge and presentation; (b) the system integrates components (i.e., toolkits) implementing alternative interactive embodiments of a particular artefact; (c) the system is capable of performing context-sensitive processing and selection of suitable symbols to interact with the user, based on information provided by a dedicated tool usually referred to as user modelling component, or user information manager, offering information, both general and task specific, on the current user; (d) multi-modality is preserved through the fusion of metaphors into an integrated environment.

References

Akoumianakis, D., Savidis, A., Stephanidis, C., in press. *Encapsulating intelligent interactive behaviour in unified user interface artefacts*. To appear in the International Journal on Interacting with Computers, Special Issue on The Realities of Intelligent Interface Technology.

Carroll., J., Mack, R., Kellog, W., 1988. *Interface Metaphors and User Interface Design*. In Handbook of Human-Computer Interaction, M. Helander (Ed.), North-Holland, pp. 67-82.

Henderson Jr., A., Card, S., 1986. *Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface*. ACM Transactions on Graphics, vol. 5(3), pp. 211-243.

Institute for Personalised Information Environment, 1995. *FRIEND21 Human Interface Architecture Guidelines*. Tokyo: Institute for Personalised Information Environment.

Moll-Carrillo, Salomon G., March, M., Fulton Suri, J., Spreenber, P., 1995. *Articulating a Metaphor Through User-Centred Design*. In the Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'95), Denver, Colorado, New York: ACM Press, 7-11 May, pp. 566-572.

Savidis, A., Stephanidis, C., 1995. *Building Non-Visual Interaction through the development of the Rooms metaphor*. Companion Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '95), Denver, Colorado, New York: ACM Press, 7-11 May, pp. 244-245.

Savidis, A., Stephanidis, C., in press. *The HOMER UIMS for Dual User Interface Development: Fusing Visual and Non-visual Interactions*. To appear in the International Journal of Interacting with Computers, 47 pages.

Savidis, A., Vernardos, G., Stephanidis, C., 1997. *Embedding Scanning Techniques Accessible to Motor-Impaired Users in the Windows Object Library*. In the Proceedings of 7th International Conference on Human-Computer Interaction (HCI International '97), San Francisco, California, USA, 24-29 August, pp. 429-432.

Stephanidis, C., 1997. *Towards the Next Generation of UIST: Developing for all users*. In the Proceedings of 7th International Conference on Human-Computer Interaction (HCI International '97), San Francisco, California, USA, 24-29 August, pp. 473-476.