Work in progress glossary of MBUI terms.

Joelle’s modifications are in red.
Gaelle’s modifications are in pink.
Paolo’s modifications are in green
Jean’s modifications directly applied in the text with comments

References.

- inital version was copied from the *CAMELEON Glossary*
- references to this glossary are implicit, others should be marked explicitly
  - **[NOTE]** - Introduction to Model-Based User Interface Design
    - this version is intended for final glossary
  - [BOU02] Bouillon, L., Vanderdonckt, J., *Retargeting Web Pages to other Computing Platforms with Vaquita*, Proc. of IEEE Working Conf. on Reverse Engineering WCRE’2002 (Richmond, 28 October-1 November 2002), A. van Deursen, E. Burd (eds.), IEEE Computer Society Press, Los Alamitos, 2002, pp. 339-348.
  - [COU95] Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J. Young, R., *Four Easy Pieces for Assessing the Usability of Multimodal Interaction: the CARE Properties*, Proc. of 5th IFIP TC 13 Int. Conf. on Human-Computer Interaction INTERACT’95 (Lillehammer, 27-29 June 1995), K. Nordbyn, P.H. Helmersen, D.J. Gilmore, S.A. Arnesen (eds.), Chapman & Hall, London, 1995,, pp. 115-120.
  - [KUR02] - Kurtev, I., Bézivin, J., Aksit, M., *Technological Spaces: An Initial Appraisal*, CoopIS-DOA’2002 Federated Conferences, Industrial Track, Irvine.
  - [NIG95] Nigay, L., Coutaz, J. A Generic Platform for Adressing the Multimodal Challenge. Proceedings of CHI’95, ACM Press, New York, 1995, pp. 98-105.
  - [URC3] - ISO/IEC 24752-3:2008 - Universal remote console -- Part 3: Presentation template
  - [Can10] Cantera Fonseca, J.M., González Calleros, J.M., Meixner, G., Paternò, F., Pullmann, J., Raggett, D., Schwabe, D., and Vanderdonckt, J. Model-Based User Interface Incubator Group, Final Report. 4 May 2010. Available at http://www.w3.org/2005/Incubator/model-based-ui/XGR-mbui/.
  - [Van93] Vanderdonckt, J., Bodart, F., *Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection*, Proc. of the ACM Conf. on Human Factors in Computing Systems INTERCHI'93 (Amsterdam, 24-29 April 1993), ACM Press, New York, 1993, pp. 424-429.
  - [Bod95] Bodart, F., Hennebert, A.-M., Leheureux, J.-M., Vanderdonckt, J., *Computer-Aided Window Identification in TRIDENT*, Proc. of 5th IFIP TC 13 Int. Conf. on Human-Computer Interaction INTERACT’95 (Lillehammer, 27-29 June 1995), K. Nordbyn, P.H. Helmersen, D.J. Gilmore, S.A. Arnesen (eds.), Chapman & Hall, London, 1995, pp. 331-336.

- Fabio Paternò, Carmen Santoro: A logical framework for multi-device user interfaces. EICS 2012: 45-50
- [MIN 05] Mens, T., Czarnecki, K., and Van Gorp, P. Taxonomy of Model Transformations, Dagstul xx to be completed by Jeanxx

Suggestions on content, wording and style.

In telcon on 24.5. we agreed to select the most relevant terms, adapt their definitions and extract these terms into the final glossary. Please use the first column to mark these terms.

The glossary is intended for inclusion into the informal MBUI note. The assumed audience are decision makers, developers or interested public and not primarily researchers. The glossary definitions should therefore adhere to these suggestions:

- use simple illustrative language
- provide examples, comparisons (even from other domains)
- do not assume exhaustive pre-knowledge of the research work nor use references to it in order to define terms
- use / refer to terms consistently
    - **model**: *description vs. representation vs. specification vs. expression*
    - **context**: *target*
        - in further variations (context-aware/independent etc.)
- emphasize functional, rather than descriptive/structural perspective
    - what is the function / usage / impact of given item in MBUI context ?
    - vs. what is it composed of ?
- cover and differentiate modern terms like *plastic / nomadic / migratory UIs*
    - state explicitly their distinguishing features or synonymy
- apply consistent **capitalization rules** for terms: Domain **m**odel vs. Abstract **U**ser **I**nterface **M**odel

USE	**Terms**	**W3C recommended**	**Definitions**
	Backward recoverability	tbd	Ability of an interactive system to provide the user with an undo facility to return to a previous desired state [Amodeus 95].
	Decoration	tbd	Kind of information attached to a description element. A way to modify the interpretation of the description element without modifying

			the element per se.
	Design recovery	tbd	[dangling] Process of recovering UI design options and models from existing source code (for example, by code static analysis, by using dynamic analysis, by behavioral analysis, by program understanding), documentation analysis, trace examination, code instrumentation, etc. Effective design recovery implies a thorough knowledge of the domain of discourse, information external to the UI source code, and deductions from it.
	Directive decoration	tbd	Decoration used when it corresponds to rules that cannot be easily expressed in terms of general-purpose inference rules. For example, suppose the multi-target development environment includes the following generation rule: "any domain concept of type Integer must be represented as a Label in the Concrete UI". If the designer wants the temperature domain concept to be represented as a gauge, a directive decoration can be attached to that particular concept. Directive decorations are pro-active.
	Dialogue (of the User Interface)		*Ordered set of actions between the user and the interactive system.*
	Dialogue model		*[dangling] Abstract description of the actions, and their possible temporal relationships, which users and systems can perform at the user interface level during an interactive session [Paternò 99].*
	Elementary Abstract Interaction Unit	OK	AIU that cannot be decomposed any further.
	Enabled Task Set (ETS)		*Set of tasks that are enabled over the same period of time according to the constraints indicated in the task model [Paternò 02].*
	Factorization	tbd	Operation, which from a set of target-specific descriptions of class X, produces a new description of class X composed of a context-independent part (i.e., shared by all the targets) and of context-dependent parts specific to each target.
	Factorization decoration	tbd	Decoration that expresses exceptions to the nominal case used as the reference in the process of multi-targeting.
	Grouping	OK	Relationships among entities (e.g., CIU) indicating that they are logically connected.
	Honesty	tbd	Property that the presentation of the system renders its functional state appropriately (e.g., it does not distort the functional state) and in a way that is understood correctly by the user [Amodeus 95].
	Interaction capacity (of an interactor)		General-purpose interaction tasks (e.g., selection, deletion, navigation) that the interactor is able to support.

	Interaction language assignment		Relation between an interaction language over a state and a non empty subset of domain-dependent concepts of a system. An interaction language l is assigned in state s to a set of domain-dependent concepts C, if it does not exist any interaction language equivalent to l over s and c. Assignment is permanent if the relation holds for any state. Assignment is total if the relation holds for C equals to the set of domain-dependent concepts of the system [Amodeus 95].[COU95]
	Interaction language complementarity		Relation between a set of interaction languages over a state and a non-empty subset of domain-dependent concepts. Interaction languages of a set L are complementary over a state s and a non empty set C of domain-dependent concepts of the system, if C can be partitioned such that for each partition Cp of C, it exists a language l of L assigned over s and Cp. Complementarity is permanent if the relation holds for any state. Complementarity is total if the relation holds for C equals to the set of domain-dependent concepts of the system. Comment. Language complementarity is best illustrated by coreferential expressions. For example, natural language and direct manipulation are complementary over the conceptual unit "city" and any state where the specification of a city name is possible: "flights from this city" and selection of a city name through direct manipulation [Amodeus 95].[COU95]
	Interaction language equivalence		Relation between a set of interaction languages over a state and a non empty subset of domain-dependent concepts of a system. Interaction languages of a set L are equivalent over a state s and a non empty set C of domain-dependent concepts of the system, if all of the domain-dependent concepts in C can be represented using either one of the language in L. Equivalence is permanent if the relation holds for any state. Equivalence is total if the relation holds for C equals to the set of domain-dependent concepts of the system [Amodeus 95].[COU95]
	Interaction language redundancy		Relation between a set of interaction languages over a state and a domain-dependent concept of a system. Interaction languages of a set are used redundantly in some state s for a domain-dependent concepts c, if these languages are equivalent over s and c, and if they are used simultaneously to represent c. For example, a wall is represented redundantly by the system via a red line (graphics interaction language) and the message "mind the red wall!" (natural language) [Amodeus 95].[COU95]
	Interaction object		[syn] Interactor.
	Interactor Model		A description that makes explicit the properties of an interactor for a

			specific purpose. For example, for the purpose of multi-targeting, this description includes the representational capacity, the interaction capacity and the usage cost of the interactor.
	Interface model		An interface model represents all the relevant aspects of a user interface in some type of interface modelling language. Components typically included in a comprehensive interface model are user tasks, domain elements, users, presentation items, and dialog structures. The elements of an interface model are grouped into model components [Puerta 99]. [see also] Model component.
	Introspection (of an entity)		The ability of the entity (e.g., an interactor, a software component) to export its properties and behaviour on request.
	Inverse functions		Function defined by inversing domain and co-domain of a previously existing function. Forward and reverse engineering can be seen as two inverse functions since the four reification steps (used for the UI production) can be recovered by their corresponding abstraction processes [Bouillon 02c]. The inverse function of a function f is denoted f -1. In this case, f is said to be inversible.
	Inversible function		See inverse functions.
	Logical interaction object		[syn] Logical interactor, abstract interaction object.
	Logical interactor		[syn] Logical interaction object, abstract interaction object.
	Logical Presentation Component (LPC)		In the Arch software architecture reference model, software component that insulates the rendering of domain objects from the actual interaction toolkit of the target platform. It is expressed in terms of logical interactors. [Arch 92]
	Logical window (LW)		A composite AIO or a physical window, or a dialog box, or a panel. [syn] Elementary workspace.
	Meta-User Interface (Meta-UI)	NOTE	Interactive system whose set of functions is necessary and sufficient to control and evaluate the state of an interactive ambient space. This set is meta- because it serves as an umbrella beyond the domain-dependent services that support human activities in this space. It is UI-oriented because its role is to allow users to control and evaluate the state of the ambient interactive space. It is an over-arching interactive system whose role is to ambient computing what desktops and shells are to conventional workstations. [Coutaz 06].
			A Meta-UI is an interactive system providing facilities to users of an underlying interactive environment to control, evaluate, and adjust the state and behaviour of that same underlying environment.

	Migrability (of a User Interface)		Ability of a UI to support migration.
	Migratable (UI).		A UI capable of migration.
	Mono-target UI		<ul><li>mentioned in the Cameleon D1.1</li><li>opposite of Multi-target UI</li><li>features ?<ul><li>single user</li><li>single platform (PC)</li><li>not environment aware</li></ul></li></ul>
	Multi-device application		An application that can be accessed through multiple devices
	Multi-device user interface		Multi-device user interfaces can be accessed through multiple devices (comment: this is a general category, which includes various subcases, such as distributed UIs, migratory UIs, and still others).
	Multi-environment targeting		Process of supporting multiple classes of environments.
	Multi-environment UI		Multi-target UI sensitive to environments variations: It is adaptable and/or adaptive to multiple classes of environments. The users and platform classes, either are modeled as archetypes, or are implicitly represented in the system.
	Multi-lingual UI		UI able to accommodate variation of the natural language according to what is needed by the user. For example, the user can switch from one language to another by selecting it from a UI menu or the system can automatically set it according to a preference stated in a profile.
	Multi-platform targeting		Process of supporting multiple classes of platforms.
	Multi-platform UI		Multi-target UI sensitive to platforms variations. It is adaptable and/or adaptive to multiple classes of platforms. The environment and user classes, either are modeled as archetypes, or are implicitly represented in the system, or are not represented in the system.
	Multi-targeting		Process of supporting multiple targets.
	Multi-user targeting		Process of adapting to multiple archetypes of users.
	Multi-user UI		Multi-target UI sensitive to users variations. It is adaptable and/or adaptive to multiple archetypes (i.e., classes) of users. The environment and the platform, either are modeled as archetypes, or are implicitly represented in the system.
	Nomadic task model		A task model is "nomadic" when it encompasses the specification of activities for all the platforms that need to be addressed. By means of filtering this "nomadic" task model according to a specific target

			platform it is possible to obtain a task model for a particular platform
	Ontological model		Within the multi-target reference framework, meta-model that makes explicit key-dimensions for addressing multi-targeting. Is independent from any domain and interactive system but is intended to be conveyed in the tools used for developing and running multi-target user interfaces. When instantiated (possibly, with tool support), ontological models give rise to archetypal models and observed models which, in turn, are specific to a particular domain and interactive system. [see also] Archetypal model, Observed model.
	Ordering (of interactors)		Relationship among interactors that indicates the existence of some ordering (e.g., temporal ordering) among them.
	Peripheral assignment		Relation between a peripheral over a state of an interactive system and a non empty subset of expressions of an interaction language. A peripheral p is assigned in state s to a set E of expressions of a language l, if it does not exist any peripheral equivalent to p over s and E. Assignment is permanent if the relation holds for any state. Assignment is total if the relation holds for E equals to the set of expressions that define l. For example, a mouse is permanently assigned to the expression of window resizing in the direct manipulation interaction language. [Amodeus 95].
	Peripheral equivalence		Relation between a non empty set of peripherals over a state of an interactive system and a non empty set of expressions in an interaction language. Peripherals in a set P are equivalent over a state s and a non empty set E of expressions in an interaction language L, if all of the expressions of E can be elaborated using either one of the peripherals in P. Equivalence is permanent if the relation holds for any state. Equivalence is total if the relation holds for E equals to the set of expressions that define L. For example, keyboard and microphone can be totally and permanently equivalent over natural language. [Amodeus 95].
	Peripheral redundancy		Relation between a set of peripherals over a state of an interactive system and an expression of an interaction language. Peripherals of a set P are used redundantly in some state s for an expression e of a language l, if these peripherals are equivalent over s and e, and if they are used simultaneously to express e. For example, the user can spell a character using the microphone and type in the same character. [Amodeus 95].
	Peripheral complementarity		Relation between a set of peripherals over a state of an interactive system and a non empty subset of expressions of an interaction language. Peripherals of a set P are complementary over a state s and a non empty set E of expressions of a language l, if E can be

			partitioned such that for each partition Ep of E, it exists peripheral p of D assigned over s and Ep. Complementarity is permanent if the relation holds for any state. Complementarity is total if the relation holds for E equals to the set of expressions of I. Language complementarity is best illustrated by spoken natural languages where concept names must be typed in. For example, in Munix, a multimodal user interface for Unix, commands that involve a file name such as remove, can be expressed using the microphone for the command name and options while file names must be elaborated with the keyboard. [Amodeus 95].
	Peripheral domain concept (for a task)		Domain concept that is not central for the task but that may have an impact on it.[see also] Central domain concept.
	Physical action		Action performed either by the user or by the system on a device.
	Physical Presentation Component (PPC)		In the Arch software architecture reference model, software component that renders the domain concepts and functions in terms of physical interactors [Arch 92].
	Plasticity domain (of a UI).		Set of contexts of use that the plastic UI covers.
	Plasticity threshold (of a UI)		Boundary of a plasticity domain.
	Pre-computed multi-target User Interface (pre-computed MUI)		Results from adaptation performed during the design, implementation or installation phases of the development process of the UI: given a functional core, a specific user interface is generated for every known target
	Predicate		Boolean-valued function of the state, behaviour, or trace of a system. A predicate may represent a property [Amodeus 95].
	Presentation Abstract Interaction Object		An AIO whose role is to present information without allowing any user interaction.
	Presentation Unit (PU)	NOTE	W3C recommended term Abstract Interaction Unit.
	Probe (for detecting context changes)		Software mechanism that monitors and detects context changes.
	Prologue		(a) Opening functional portion of the execution of a reaction to context change. It prepares the reaction: the current task is completed, suspended, or aborted; the execution context is saved; if not ready for use, the new version of the user interface is produced on the fly (e.g., a new presentation, a new dialogue sequence, etc.). (b) In ARTStudio, reference to a function of the Functional Core before the execution of a task.
	Property		Observable characteristic of a system that can be measured and

			described by a predicate [Amodeus 95].
	Reaction (to context change)		Three-step process that permits adaptation to context changes: situation recognition, reaction computation, and reaction execution.
	Reaction computation		Identification of candidate reactions to context change, then selection of the reaction that best fits the situation.
	Reaction execution		Three-step process that consists of a prologue, the commutation to the new UI, and an epilogue.
	Recoding (of a UI)		[dangling] Any functionally equivalent transformation of the source code of a final UI. Reformatting and Refactoring are particular cases of recoding.
	Reconfigurability	NOTE	[dangling] Ability of a UI to support multiple targets simultaneously by offering multiple UI configurations while not necessarily preserving usability.
	Recovery		Performance of actions that take a system from some `unsafe' or undesired state to one satisfying some safety property [Amodeus 95].
	Recoverability		Property that the system provides the user with means to undo the effect of some action. [Amodeus 95].
	Redesigning (a UI)		Changes to design characteristics. Possible changes include restructuring design architecture, altering the domain model, etc. Such changes may derive from a change of context of use.
	Redocumenting (a UI)		From the UI source code, process of deriving another form of UI documentation such as, but not limited to, data structure, data flow diagram, I/O analysis. [see also] Reformatting.
	Reengineering (a UI)		Examination and alteration of a subject interactive system to reconstitute it in a new form and the subsequent implementation of the new form. This process encompasses a combination of sub-processes such as reverse engineering, restructuring, redocumentation, forward engineering, and retargeting [STSC]. [syn] Renovation, reclamation.
	Refactoring (of a UI)		[dangling] Functionally equivalent transformation of the source code of a final UI to improve its efficiency, its performance. [see also] Recoding.
	Reflexivity		Reflexive functions are functions mapping an existing UI representation at a given level of abstraction to another UI representation at the same level of abstraction for the same context of use [Bouillon 02b]. In ArtStudio, reflexive functions are performed

			manually [Thevenin 01].
	Reformatting (a UI)		Functionally equivalent transformation of a source code that changes the structure of the code to improve readability.
	Regenerating (a UI)		Composition of the two reification steps from the abstract UI to the final UI. It is used to complete the process of retargeting. To obtain another UI for any other computing platform, regenerating can be defined similarly by the composition regenf = reic o reia so as to represent the complete process by regenf o retarga [Bouillon 02b]. [see also] Retargeting.
		NOTE	Transformation of a model to a more concrete level, the opposite of abstraction. For example the reification of a concrete user interface model involves generation of the final user interface code.
	Representational capacity (of an interactor)		Types of domain concepts the interactor is able to represent (e.g., a table, an integer).
	Restriction		Function obtained by restraining the domain of the initial function to those elements of the domain that satisfy a given predicate. In VAQUITA [Bouillon 02a, c, Vanderdonckt 01], constraints attached to retargeting/translation functions are defined as restriction of translations and abstractions. See http://www.isys.ucl.ac.be/bchi/research/vaquita.htm. For a given function to be applied in a specific computing platform, there is a need to define a condition to be satisfied when applying this function. For example, a constraint may be imposed when a translation between two computing platforms occurs, such as: "the target computing platform does not allow hierarchy of presentation elements deeper than a certain threshold". The WML language instantiates this constraint to 9 (not more than 9 presentation levels in decks and cards), while certain versions of cHTML instantiates this contraint to 4 (not more than 4 levels of cHTML tags). The restriction of function is therefore required. [syn] Selection. [see also] Retargeting.
	Restructuring (a UI)		Transformation from one presentation form to another at the same level of abstraction while preserving the subject's system external behavior (functionality and semantics) [IEEE Terminology]. The task, the functions (application model) and the domain models should remain identical. Moreover, the dialog model, which is left unchanged, also represents the external behaviour.
	Retargeting (a UI)		Process allowing the production of an abstract UI tailored for a particular computing platform from a final UI. Retargeting is done at design time. It is the composition of three functions: two successive abstractions followed by a translation for another platform (retarga=

			transa o absa o absc) [Bouillon 02c] The retargeting / translation function can be subject to restrictions due to constraints imposed by the target platform. Rather, the double abstraction up to the abstract UI level and a translation to a new context of use is independent of any computing platform.
	Retasking (a UI)		The change of the task model to fit a different context of use.
	Revamping (a UI)		Change of the user interface without modifying the functional core. Revamping makes possible to considerably modify the look and feel of a user interface. Not only the visual presentation of screens can be changed, but also the phrasing can be redefined, multimedia features can be added, or on-line documentation can be created. However, revamping doesn't imply a change in the requirements, nor re-specification. Revamping is a useful reengineering strategy when an organisation wishes to adopt graphical user interfaces (GUIs) by using middleware products, which sit between the legacy system and the UI. The new UI is created (manually [Csaba 97] or automatically [Stroulia 00]) at design time. Revamping is frequently performed by designers to beautify a presentation according to users' needs.
	Selection		[syn] Restriction. [comment] Selection is the same as a restriction (mathematical term), but is more frequently used in the domain of database engineering.
	Situation recognition		Identification of the current context of use.
	State		Assignment of values to names representing the observables of an interactive system [Amodeus 95].
	State vector (of a component, of a system)		Names that define the state of a component, of a system [Amodeus 95].
	Target change		Change of at least one element in the triple "e, p, u".
	Target environment		Archetypal set of environments envisioned for the interactive system.
	Target aware (UI)		[syn] Context aware UI.
	Target sensitive (UI)		[syn] Context sensitive UI.
	Target user		The archetypal end-user envisioned for the interactive system.
	Task domain concept		Concept identified by task analysis as relevant to the user to accomplish tasks in that domain. [syn] Domain concept.
	Task presentation set		Set of tasks supported by one presentation.

	Transient description		[syn] Transient model.
	Transient model		Within a multi-target development environment, intermediate description used in the process of producing multi-target UI's.
	Transition task		When performed, a task that triggers a new presentation.
	UI dialogue		See Dialogue of the UI.
	UI distribution		See Distribution of the UI.
	UI migration		See Migration of the UI.
	Task migration		Dynamic transfer of task performance between agents (whether these agents be humans or computational).
	Transition UI		Feedback provided to the user during the adaptation of the UI to changes of context of use so that the user can follow and evaluate the evolution of the adaptation process.

C References

C.1 Normative references

[RFC2119]
"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997.
[SOAP12]
"SOAP Version 1.2 Part 1: Messaging Framework", M. Gudgin, M. Hadley, J. Moreau, H. Nielsen, December 2001.
[XML10]
"Extensible Markup Language (XML) 1.0 (Second Edition)", T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, October 2000.
[XMLSCHEMA1]
"XML Schema Part 1: Structures", H. Thompson, D. Beech, M. Maloney, N. Mendelsohn, May 2001.
[XPATH10]
"XML Path Language (XPath) Version 1.0", J. Clark, S. DeRose, November 1999.

C.2 Informative references