Coupling Interaction Resources: an Analytical Model

Joëlle Coutaz[(1)] *& Stanislaw Borkowski* [(2)] *& Nicolas Barralon* [(1)]

[(1)] Université Joseph Fourier, Lab. CLIPS-IMAG, BP 53, F-38041 Grenoble Cedex 9, France
{Joelle.Coutaz,Nicolas.Barralon}@imag.fr

[(2)] Project PRIMA, Lab. GRAVIR, 655 Av. de l'Europe - 38330 Montbonnot-St. Martin, France
Stan.Borkowski@inrialpes.fr

Abstract

This paper addresses the problem of coupling interaction resources. Coupling is the action of binding two entities so that they can operate together to provide a new set of functions. For example, coupling a piece of cardboard with a steerable camera-projector pair results in a portable interactive surface. Although coupling is not a new phenomenon, recent research demonstrates that coupling opens the way to unbounded forms of interaction. Because the risk of introducing complexity is high, we propose an analytical model that can serve two purposes: the model can be used to inform the design of novel user interfaces; the computational version of the model can be used at run time as a mechanism to evaluate candidate solutions when dynamic adaptation to the context of use must be undertaken. We illustrate the contribution of the model with three chosen running examples.

1. Introduction and related work

In their provocative paper, "Making Sense of Sensing Systems", Bellotti and her colleagues address the challenge of interaction for novel genres, beyond the familiar GUI paradigm [2]. They propose five issues as the beginning of a systematic approach to the design of innovative systems: address (directing communication to the system), attention (establishing that the system is attending), action (defining what is to be done with the system), alignment (monitoring system response) and accident (avoiding or recovering from errors). These issues, inspired from social science, cover the general problem of interaction with an emphasis on communication rather than on cognition. In this article, we address the problem of coupling interaction resources in novel user interfaces with an emphasis on system development.

Many examples of the state of the art illustrate that coupling interaction resources is an increasingly important problem of interaction. In Pebbles, PDA's can be coupled to a single shared display [15]. Similarly, two ConnecTables can be dynamically coupled by approaching them close to each other to enlarge the screen real estate [19]. Dynamo supports cooperation through a public interactive surface by allowing users to couple their personal devices based on the concept of "mobile interaction points" [12]. With the DataTiles system, users can obtain new services by configuring tagged transparent tiles on a flat panel display [17]. In these examples coupling of interaction resources is implemented ad hoc.

On the other hand, in [20] Ullmer and Ishii formalize coupling of physical objects with bits [8] in a framework inspired by the MVC model. Their approach, however, is strictly focused on software architecture. Icon [6] is a toolkit based on a data-flow model. Though it enables creation of multiple configurations of input devices, it only allows to define couplings during design; that cannot be modified at runtime. Generally, in the state of the art we notice the lack of the analysis of mechanisms governing coupling.

Coupling is not a new phenomenon, but in the GUI genre, most couplings are pre-packaged and immutable. Therefore, they are taken for granted by the HCI community. With the proliferation of new computing forms and functions, coupling becomes an important source for innovation at a risk of high complexity. For example, how do I know that my home keys can be coupled with my shoes so that I will never forget them? How do I know that I can couple them by shaking them together [11]? How do I know that they are now coupled? If so, how can I uncouple them?

To address these issues, we propose an analytical tool that models coupling as a finite state automaton. Each state is defined by a set of predicates whose manifestation at the user interface can be tested against a usability framework. Because we are system developers, we use the IFIP properties [9] to show how the automaton can be exploited at the design stage. Because these properties have the potential to be expressed as metrics, we hope, in our future work, to embed the automaton into a run time infrastructure to serve run time adaptation as coupling and uncoupling occur [1].

This article is structured as follows. First, we introduce the conceptual foundation of our work with the formal definition of coupling along with the concept of interaction resource. Then, in Section 3, we illustrate different forms of couplings with three systems implemented without the predictive support of our analytical model. Sections 4 and 5 present the two complementary facets of the model: 1) the formal decomposition of coupling as a finite state automaton (FSA) and 2) the verification, based on the FSA, of the system-centric IFIP usability properties. We use systems presented in Section 3 to show that a number of usability problems detected through experimental user studies could have been avoided using the model in a formative manner early in the design process.

2. Coupling interaction resources - definitions

Coupling is the act of binding two entities so that they can operate together to provide a new set of functions that cannot be provided individually by the entities. More formally, let;

- **E** be a non empty finite set of entities,
- **F**, a non empty finite set of all the possible functions that can result from the set **C** of all the possible couplings between the entities of **E**.

If $e1$ and $e2$ are two entities ($e1, e2 \in \mathbf{E}$), and c, a particular coupling of $e1$ with $e2$ ($c \in \mathbf{C}$), then c is defined as the Cartesian product of **E** in **F**:

$$c : \mathbf{E} \times \mathbf{E} \rightarrow \mathbf{F}$$

and F ($F \subset \mathbf{F}$) is the set of functions that results from the coupling c of $e1$ with $e2$. We note this coupling by the triple:

$$(e1, c, e2).$$

In the context of this work, entities are everyday objects such as tables, walls, and pens, as well as interaction devices such as keyboards, mice and computer screens. To address this diversity, we draw on the ontology presented in [5] where real world entities involved in a human-computer interaction experience are unified into the notions of interaction resource and interaction role. Assignment of roles to entities is performed by *actors*: system actors and/or human actors. A physical entity is an *interaction resource* when it plays one, possibly multiple, interaction role(s). These are: the role of a surface, of an instrument, of an actuator and/or a sensor.

From a system perspective, a physical entity plays the role of a *surface* if the system maps digital information on it and makes the result observable to another actor (such as a user) by the way of an actuator. A physical entity plays the role of an *instrument* if it is sensed as a thing used by another actor (e.g., a user) to modify the state of digital information. When coupled with a system, an *actuator* is a physical entity capable of modifying the state of the world such as making observable digital information on a surface. Conversely, a *sensor* is a physical entity capable of observing the state of the world, such as observing mouse clicks or human fingers.

In the GUI genre, most roles and couplings are defined statically: screens are assigned the role of surface, mice and keyboards that of instrument; actuators are embedded within screens and these actuators and screens are coupled for ever. Similarly, by construction, sensors are coupled permanently with mice and keyboards. Pointer instruments and input text instruments are coupled providing the “input focus” function. The existence of this coupling is not observable to users. Coupling is reduced to pure convention that must be learnt. On the other hand, other couplings are observable. For example, the coupling between a mouse and a screen, which provides the “location selection” function, is observable as a cursor whose behavior is tightly coupled to the movements of the mouse.

In ubiquitous computing unpredictability is paramount, interaction roles and couplings are not necessarily decided by design. On the contrary, couplings and interaction roles are intended to emerge opportunistically. To evaluate and understand this risk, we have analyzed the notion of coupling in a systematic way. By looking carefully at the nature of coupling, we observe that it is not an instantaneous phenomenon whose existence is “yes” or “no”.

3. Coupling examples

Our first example illustrates coupling between workstations where the conventions of the GUI paradigm are preserved. We then present two examples in which the boundaries of the interactive space are not so well delineated: the computers no longer sit on the desk, but have disappeared into the background. In the second example, the user interface is distributed across the surfaces of an interactive environment, and in the third example, the user interface can dynamically migrate to a piece of cardboard transformed into a Portable Display Surface (PDS).

3.1. Coupling Conventional Workstations

In this example, the interactive space is populated with workstations running a mix of MacOS X and Windows. Here, the native window managers have been replaced with a middleware (I-AM for Interaction-Abstract Machine) that supports dynamic coupling of screens, keyboards and mice, to form a unified interactive space. A detailed system implementation of I-AM can be found in [14]. In such a space, interaction resources seem to be handled by a single computer even when being connected to different machines. I-AM is similar in spirit to the roomware developed for iRoom and i-LAND. Although iRoom supports heterogeneous workstations, windows in iRoom cannot cross screen boundaries. In i-LAND, windows can cross screens boundaries but the underlying workstations run the same operating system.

For interaction roles and couplings, the I-AM middleware preserves conventions defined in the GUI genre. For example, the pointing instrument of a machine (e.g., a mouse) is automatically coupled with the input text instrument (e.g., a keyboard). As for screens, windows can sit between two coupled screens although these screens may be connected to different workstations and may differ in resolution and orientation.

Figure 1 shows an example of the interaction technique for configuring an interactive space populated with two workstations running I-AM on top of their native operating systems, Windows XP and MacOs X respectively.

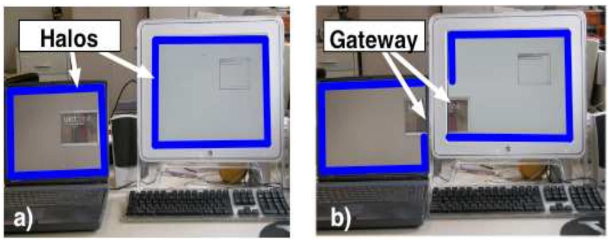

Figure 1. The PC and the Macintosh screens are decoupled and run two applications (left). The two screens are coupled to form a single display area (right). Outlines have been artificially enhanced on the pictures to increase readability.

In Figure 1-a, two applications are running on two independent workstations. The closed blue halo that outlines each screen denotes the possibility for currently uncoupled screens to be coupled. In Figure 1-b, the screens are coupled by bringing them in close contact. The result is the “single display area” function. A blue border outlines the display area and a gateway shows where windows can transit between the screens.

Coupling by close contact is similar to Hinckley’s synchronous gestures where devices are bumped against each other [10]. Whereas Hinckley uses accelerometers, I-AM uses infrared-based proximity sensors. Alternative interaction technique, inspired from SyncTap [18], called “Click’n Couple”, consists in bringing face to face the cursors of the mice managed by different workstations, and then click the mouse buttons simultaneously.

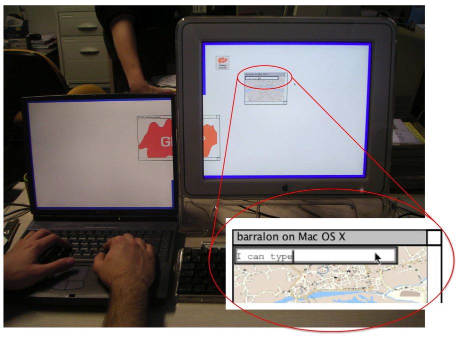

Figure 2. Entering characters in a text field located on a Macintosh screen using a PC keyboard: to do so, the user has selected the text field with the PC touchpad.

Once two screens are coupled, users can couple any instrument to any interactor of the unified surface. For example, in the configuration of Figure 2, a PC mouse can be used to move a window created on the Macintosh and migrate this window to the PC. This facility is similar to that provided by PointRight [13] except that in I-AM, multiple mice cursors can be used simultaneously.

The multiplicity of instruments raises interesting questions. For example, Figure 2 shows the situation where the user has selected a text field with the touchpad of the PC. Because the touchpad is automatically coupled with the keyboard of the PC, the text field is now the input focus for the PC keyboard. Thus, the user can enter text with the PC keyboard (in the example, "I can type text"). Then, the user can select the same text field with the mouse of the Macintosh. Because the mouse of the Macintosh is automatically coupled with the keyboard of the Macintosh, the text field now becomes the input focus for the Macintosh keyboard as well. Since the text field is the input focus for two keyboards, input can be provided simultaneously from the two keyboards!

As these examples show, coupling familiar entities such as the screens of our everyday life workstations, opens the way to unusual situations. The FAME meeting room presented next goes a step further with less familiar forms of coupling.

3.2. The FAME Meeting Room

The FAME meeting room [7] is an interactive space that supports collaborative exploration of information spaces based on mixed reality and multimodal interaction. Figure 3 shows the overall setting. A table and two walls play the role of surfaces. Each surface is coupled to its own video-projector. In addition, the table is sensed by a camera that tracks colored, 4 cm wide round shape tokens made of plastic. Tokens play the role of instruments: users can couple them to the table to retrieve information (see Figure 3 and Figure 4). By dropping a token on a selectable digital item a flower menu pops up, from which users can select information by moving the token to the appropriate petal (see Figure 4 for details). Multiple tokens can be coupled simultaneously to the table: the system is able to track more than 12 tokens simultaneously while maintaining latency of 80ms on average on a dual PowerPC 7400 (G4) 1.4 Ghz machine.

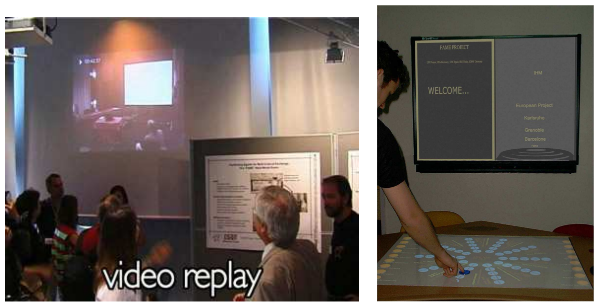

Figure 3. Two settings of the FAME room. The setting at the Barcelona Cultural Forum (left). The setting in the lab (right). Here, a user is coupling a blue token to the digital information projected on the table. Results are projected on the wall.

The originality of the FAME augmented meeting room lies in the diversity of the modalities available as well as in the presence of both explicit and implicit interaction. Modalities include: speech recognition of explicit commands such as "is any information about computer vision available?", direct manipulation of physical tokens, graphical and sonic renderings. Users chatting together perform implicit interaction from which the system can recognize topics of interest.

As far as coupling is concerned, the FAME meeting room includes pre-wired couplings between the surfaces and the actuators (video-projectors) and sensors (camera and microphones). Similarly, the table and the information wall are pre-coupled based on their spatial relationships: when a petal is selected, an animated icon moves from the petal to the wall so that users' focus of attention is driven appropriately. The FAME room offers dynamic coupling as well: those between the table and the tokens.

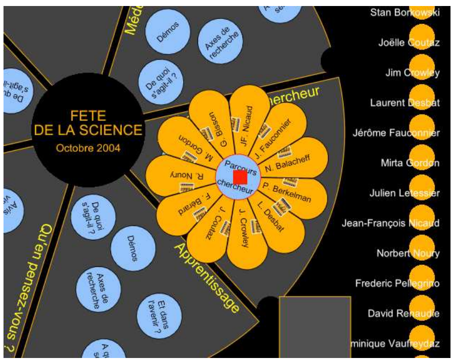

Figure 4. Partial view of the FAME UI. Selectable digital information is rendered as round shape items that match the shape of the tokens. A flower menu is obtained by placing a token on a round-shape item.

Whereas in FAME, the surfaces are stationary, the PDS can be hold in the hand and moved around.

3.3. The Portable Projection Screen (PDS)

The PDS results from coupling a hand-held planar physical entity with the Steerable Camera-Projector pair shown in Figure 5 [3][4]. The surface of this entity should be uniform, light-colored and delineated by a clearly marked border. In practice, the entity can be made of a white cardboard with black borders (Figure 6, left). The Steerable Camera-Projector (SCP) is an actuator/sensor device that binds a video-projector and a camera to the same mechanical pan-tilt motorized assembly. The camera and the video-projector are coupled by construction in such a way that the system can observe the object on which it acts. Similar coupling of a motorized camera and a projector was first introduced by Pinhanez with the Everywhere Display [16].

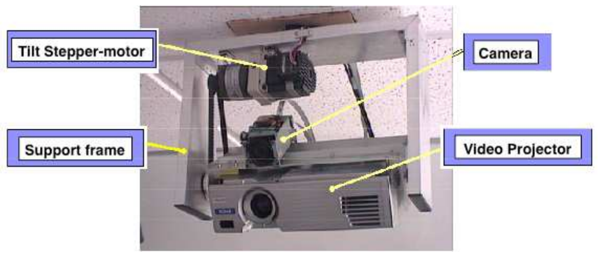

Figure 5. The Steerable Camera-Projector (SCP) pair.

Coupling the piece of cardboard with the SCP is performed by introducing the cardboard over an image projected by the SCP on a wide surface such as a table. The piece of cardboard is couplable with the SCP because it has the right shape and size, the right background color and has clearly delineated borders. When the cardboard enters the field of view of the SCP, the SCP is decoupled from the wide surface and coupled to the piece of cardboard. The resulting function is the PDS: the image of the source surface is rescaled, rectified, and transferred to the cardboard and sticks to it even when the cardboard is moved. The piece of cardboard has now the role of a surface as well as that of an instrument: because it is held in hands, its location in space, panning and tilting, can be used to express commands to the system [21].

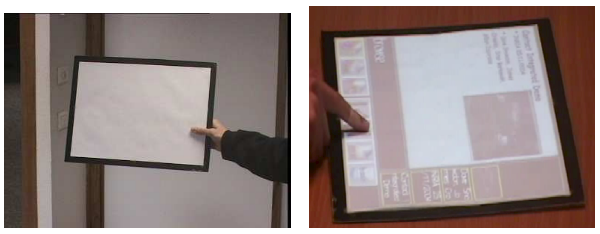
Figure 6. A piece of white cardboard with black borders as a potential interactive surface (left). The cardboard transformed into an interactive surface when coupled with the SCP (right).

As the cardboard is moved, its corners are tracked to allow the projector to be steered in order to maintain the image on the PDS. Interactive widgets can be projected onto the PDS so that a user could select and manipulate information using his fingers as shown in Figure 6.
To uncouple the SCP and the cardboard, the user places the cardboard on a wide surface known to the system: the SCP is decoupled and re-coupled to the wide surface, and the digital information is projected back to the wide surface. The PDS function ends and the cardboard looses its status of interaction resource.

4. Coupling as a Finite State Automaton

A coupling, c, between two entities, e1, and e2, denoted as (e1, c, e2), has a life cycle. As shown in Figure 7, this life cycle includes eight states.
A state of (e1, c, e2) is defined by the conjunction of the following set of predicates:

- *Coupled (e1, c, e2)* = TRUE if and only if **F**≠∅ where **F** is the set of functions that results from (e1, c, e2). If **F**=∅, then Coupled (e1, c, e2) = FALSE and NotCoupled (e1, c, e2) =TRUE.
- *Locked (e1, c, e2)* = TRUE if the state of e1 does not permit to modify the state of (e1, c, e2). This predicate can be used to express that e1 is not “socially” or “technically” available to enter or exit the coupling c with e2. For example, a user does not want to connect his private PDA to a public screen. The state of (e1, c, e2) is kept unchanged until Locked (e1, c, e2) = FALSE or NotLocked (e1, c, e2) = TRUE.
- *Couplable (e1, c, e2)* is an expression of predicates **P**, where **P**≠ Coupled (e1, c, e2) and **P**≠Locked (e1, c, e2). This expression specifies the conditions (different from Coupled (e1, c, e2) and Locked (e1, c, e2)) that are necessary for (e1, c, e2) to happen. For example, the compatibility of shapes or roles between two entities can be expressed with Couplable. Symmetrically, Uncouplable expresses the conditions (different from Coupled (e1, c, e2) and Locked (e1, c, e2)) that are necessary for (e1, c, e2) to end.

The automaton shown in Figure 7 corresponds to the coupling (e1, c, e2). A similar automaton models (e2, c, e1). It is comprised of two sub-automata: one that includes the states 1, 2, 3, 4 where *Coupled (e1, c, e2)* is TRUE, the other one that covers the states 5, 6, 7, 8 where *Coupled (e1, c, e2)* is FALSE. States 4 and 6 serve as gateways between the two sub-automata. State 4 corresponds to the situation where all the conditions for realizing (e1, c, e2) are satisfied. Only a coupling request event is missing to enter state 6.

By definition, coupling occurs between two entities. But by transitivity, multiple entities are bound together to form an interactive space whose functionalities depend on the set of functions that each coupling delivers. Do these functions, all together, form a “consistent story” for the user? Since the management of the interactive space corresponds to the interplay of multiple automata, how many of them does the system (and the user) can reasonably handle at a time? As one can see, modeling coupling as an automaton reveals the problem of scalability right away. In addition, by applying a usability framework (such as the IFIP properties, Bellotti’s et al., the Cognitive Walkthrough, Nielsen’s or Bastien-Scapin’s criteria) to every state of the automaton can provide useful design insights. As discussed in the introduction, we have selected the IFIP properties for their system-centric perspective and their potentiality for serving as run time metrics.

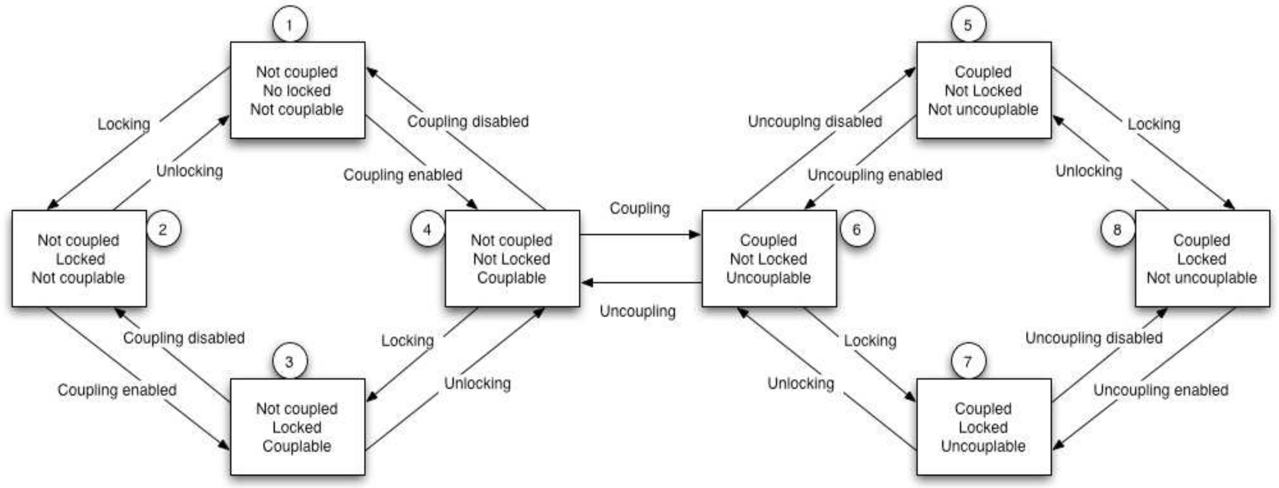

Figure 7. Coupling (e1, c, e2) as a Finite State Automaton. For the sake of readability, the transitions between states 1 and 3, 2 and 4, 5 and 7, 6 and 8 are not represented.

5. Coupling and usability properties

Given a particular context of use identified as a reference, given a coupling (e1, c, e2) identified as central for this context, and given a set of relevant properties for this context, the satisfaction of these properties must be evaluated for each state of the life cycle of (e1, c, e2). This evaluation can be performed in a formative and predictive way during the design stage of the system, as well as by the system itself at run time to choose between multiple candidate solutions when adaptation to the context of use must be performed. We illustrate this process with a subset of the IFIP properties applied to our running examples.

Reachability is the ability of the system to permit users to reach a desired state. Applied to our problem, reachability means two things: (a) the automaton of the (e1, c, e2) must cover the life cycle depicted in Figure 7, and (b) the system provides users with the interaction techniques to reach these states. None of the three systems depicted above fully covers the life cycle of Figure 7. The extreme example is the PDS: We need to consider two couplings: (SCP, c1, table) and (SCP, c2, cardboard) where *Coupled (SCP, c1, table) = TRUE* implies that *Coupled (SCP, c2, cardboard) = FALSE* (i.e. the SCP cannot be simultaneously coupled to two entities). Initially, (SCP, c1, table) is in state 6, and (SCP, c2, cardboard) is in state 4. By bringing the cardboard in the field of view of the SCP, (SCP, c1, table) enters state 4 and (SCP, c1, table) enters state 6. Therefore, c1 and c2 alternate between states 4 and 6 thus covering the life cycle of Figure 7 in a very limited way.

So, the general design question becomes: for a particular system under consideration, is partial reachability good enough?

Non-preemptiveness denotes the ability of the system to permit users to reach a desired state directly from any state (i.e. the length of the interaction trajectory is equal to 1). Applied to our problem, this means that the automaton of (e1, c, e2) is completely connected (transitive closure). The surfaces in I-AM, can always be coupled or decoupled in a single gesture. On the other hand, in the FAME room, a token cannot be directly coupled to a petal when this petal is already coupled with another token. So, the general design question becomes: for a particular system under consideration, is preemptive coupling acceptable or even desirable?

Multithreading: Ability of the system to support user interaction pertaining to more than one task at a time. Applied to coupling, this property translates into the capacity for the system to support multiple couplings simultaneously. This capacity is supported in the FAME augmented meeting room since multiple tokens can be coupled to the table to express multiple requests in parallel. On the contrary, it is not supported by the PDS where the SCP cannot be shared. So, the general design question becomes: for a particular system under consideration, is partial multithreading desirable and if so, at what scale?

Task Migratability corresponds to the ability to pass control for the execution of a given task so that it becomes either executed by the user or by the system or shared between them. In the context of this study, this means that coupling and uncoupling tasks can be dynamically performed either by the system or the user or as cooperation between them. This raises the interesting question of the balance between explicit and implicit interaction. For example, in I-AM by using proximity sensors, two screens that are close enough are coupled automatically by the system. On the other hand, with the "Click'n Couple" interaction technique, coupling is explicitly performed by the user.

Adaptability denotes the ability of the user to modify the user interface. So far, the state of the art (including our own examples) does not show any example of interaction techniques related to coupling that can be customized by users: coupling/uncoupling, locking/unlocking, etc. are conventional. They must be learnt. Here, we spot an interesting area for future research: what are appropriate expressions of coupling and uncoupling given that the interaction resources are potentially large?

Recoverability: Ability of the user to take a corrective action once an error has been recognized (i.e. an undesired state has been reached). Technically, this property expresses for every state, the existence of a reverse transition, and for the user the existence of an interaction technique that triggers this transition. This property is satisfied for the tokens of the FAME table: Uncoupling a token from the table can be done at any time by hiding the token with the hand. Conversely, to re-couple, the user can drop the token back at the appropriate location on the table.

Observability: Ability for the user to evaluate the internal state of the system from its perceivable representation. When applied to the life cycle of coupling, this property requires that every state of the automaton be made observable to users. As a counter-example, let's consider the coupling of tokens with the FAME table. Let t1 and t2 be two tokens, and i1, a selectable item projected on the table. At the beginning, the user is holding the tokens in his hands, and i1 is rendered as a round shape graphics. Thus, (t1, c, i1) is in state 4. By dropping t1 on i1, one couples t1 with i1 making the select function available: the automaton for (t1, c, i1) enters state 7. To make this state observable, i1 opens itself as a flower where each petal is couplable to t1. i1 is now locked for tokens different from t1. As a result, dropping t2 on any petal of i1 will have no effect (although dropping t2 on another selectable item i2 would work correctly as it is unlocked). Two informal user studies with 30 subjects unfamiliar with the FAME room showed that some people selected the petals using additional tokens instead of traversing the flower menu with the coupled token. If FAME designers had our analytical model at the time of the development of FAME, they would have been able to spot this problem and take corrective actions such as making the *Locked* state observable or allowing coupling a flower menu with multiple tokens.

Predictability: Support for the user to determine the effect of future action based on past interaction history. Here, the user should be able to anticipate the set of functions F that results from coupling. I-AM provides an example showing the importance of predictability: Let S1 be a surface coupled by construction (i.e. GUI conventions legacy) to a PC workstation and its pointing instrument I (i.e. mouse). Let S2 be a surface connected to a second computer with no instruments. S1 can be now coupled to S2 using one of the interaction techniques presented in 3.1. According to the I-AM model, the instrument I can get coupled to S2 as well: it can be used to modify the information space mapped on S2. Thus the cursor of I can be mapped on S2. Can the user predict what will happen if S1 is uncoupled from S2 while I is mapped on S2? Will the instrument be uncoupled from S1 and stay coupled with S2? Or, alternatively, will it follow its home surface? If so, where will the cursor re-appear on S1? This type of problem was spotted by the PointRight [13] developers where they stated that "a free-space device [such as a wireless mouse] needs an explicit starting screen".

Translated into our framework, this means that when a wireless mouse is dynamically coupled to the interactive space, its associated cursor must be mapped onto a predefined home screen in order to support predictability.
In summary, applying a usability framework such as the IFIP properties to the states of a set of carefully selected couplings can provide useful insights before the implementation of the final UI can take place.

6. Conclusions

In the conventional GUI paradigm, coupling is limited and pre-packaged. Therefore, it is taken for granted. With the proliferation of new computing forms and functions, coupling becomes an important source for innovation at a risk of high complexity. To address this risk, we have presented an analytical model that structures the coupling of interaction resources as a finite state automaton where each state is defined by a set of predicates and tested against the properties of a usability framework.
Our analytical model can be used to inform design. For example, in the case of the FAME room, the menu traversal problem could have been detected before implementation started.
Our model can be used to inform future research as well. For example, we have not seen any example in the literature where coupling covers all the states of our automaton. Associated to these states, new interaction techniques should be devised to allow users to move between states.
In the long run, our analytical model can be used for system self-evaluation: by essence, the finite state automaton can be translated into code, and the properties can be mapped into observables. These would then be integrated into a run time infrastructure such as [1] whose role includes the computation of new forms of couplings to adapt to the context of use while preserving usability.

Acknowledgements

This work has been partly supported by the European IST-2000-26070 GLOSS project and the IST-2000-28323 FAME project. We wish to express our sincere thanks to J. Crowley for his thoughtful recommendations, to C. Lachenal for the design and implementation of I-AM, to F. Bérard, B. Cohen and S. Rouillard for the design and implementation of the FAME room.

References

[1] Balme, L., Demeure, A., Barralon, N., Coutaz, J., Calvary, G. CAMELEON-RT: a Software Architecture Reference Model for Distributed, Migratable, and Plastic User Interfaces. Second Euro-pean Symposium on Ambient Intelligence, EUSAI 04, LNCS 3295, Markopoulos et al., 2004, pp. 291-302
[2] Bellotti V., Back M., Edwards K., Grinter R., Henderson A., Lopes C.: making Sense of Sensing Systems: Five Questions for Designers and Researchers. In: CHI 2002 Conf. Proceedings, Conference on Human Factors in Computing Systems, Vol. 4(1), ACM New York (2002) 415–422
[3] Borkowski, S., Riff, O. and Crowley, J.L. Projecting rectified images in an augmented environment. In Proc. ProCams Workshop. International Conference on Computer Vi-sion, ICCV 2003 (October, Nice,France) IEEE Computer Society Press, 2003
[4] Borkowski, S., Letessier, J., Crowley, J. L. Spatial Control of Interactive Surfaces in an Augmented Environment. Engineering HCI-DSV-IS 2004, The 9th IFIP Working Conference on Engineering for Human-Computer Interaction, July 2004.
[5] Coutaz, J., Lachenal, C. and Dupuy-Chessa, S. Ontology for Multi-surface Interaction. In Proc. Interact'2003, M. Rauterberg et al. Eds., IFIP, 2003, pp. 447-45
[6] Dragicevic, P. Un modèle de configurations d'entrée pour des systèmes interactifs multi-dispositifs hautement configurables, Thesis of Université de Nantes, 2004.
[7] IST-2000-28323 FAME European project. *http://isl.ira.uka.de/fame/*
[8] Ishii, H. and Ullmer, B. Tangible Bits : Towards Seamless Interfaces Between People, Bits, and Atoms. In Proc. of CHI'97, pp. 234-241, 1997.
[9] Gram, Ch., Cockton, G. (Eds.). Design Principles for Interactive Software. Chapman & Hall, London, 1996.
[10] Hinckley, K. Synchronous gestures for multiple persons and computers. In Proc. UIST03, ACM, 2003, pp. 149-158.
[11] Holmquist, L.E., Mattern, F., Schiele, B., Alahuhta, P., Beigl M., Gellersen, H.W. Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. In Ubicomp 2001, Atlanta, 2001, Pages 116-122.
[12] Izadi S., Brignull, H. Rodden, T. Rogers , Y. Underwood, M. Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media. The 16th annual ACM symposium on User interface software and technology, UIST 2003. Vancouver, Canada, 2003.
[13] Johanson, B., Hutchins, G., Winograd, T., Stone, M. PointRight: experience with flexible input redirection in interactive workspaces. In Proc. of the 15th annual ACM symposium on User interface software and technology UIST 2002. Paris, France. Pages 227-234.
[14] Lachenal, C. Modèle et Infrastructure Logicielle pour l'Interaction multi-instrument, multisurface. PhD Thesis, Université Joseph Fourier, December 2004.
[15] Myers B., Stiel H., Gargiulo R. Collaboration Using Multiple PDAs Connected to a PC. Proceedings Computer Supported Collaborative Work, CSCW98, Seattle, WA. Pages 285-294. 1998.
[16] Pinhanez C., The everywhere displays projector: A device to create ubiquitous graphical interfaces. In Proceedings of Ubiquitous Computing 2001 Conference, September 2001.
[17] Rekimoto, J., Ullmer, B., Oba, H. DataTiles: A Modular Platform for Mixed Physical and Graphical Interactions. In Proceedings of ACM CHI'01 Conference on Human Factors in Computing Systems, Seattle, Washington, 2001, Pages 269-276.
[18] Rekimoto, J., Ayatsuka, Y. and Kohno, M. SyncTap: an Interaction Technique for Mobile Networking. In Proc. Mobile HCI 2003, L. Chittaro Ed., Springer Publ., LNCS 2795, 2003, pp. 104-115
[19] Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N., Steinmetz, R. ConnecTables: Dynamic Coupling of Displays for the Flexible Creation of Shared Workspaces. In Proceedings of the ACM UIST'01, Orlando, Florida, 2001, pp.11-20
[20] Ullmer, B. and Ishii, I. Emerging frameworks for tangible user interfaces, IBM Systems Journal,Volume 39, Issue 3-4, July 2000, pp. 915- 931
[21] Zhang Z., Wu Y., Shan Y., Shafer S., Visual Panel: Virtual Mouse, Keyboard and 3D Controller with an Ordinary Piece of Paper. Proceedings of the 2001 workshop on Perceptive user interfaces, Orlando, Florida, 2001, p. 1-8.